Motif Discovery on Promotor Sequences

Maximilian Häussler, Jacques Nicolas

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

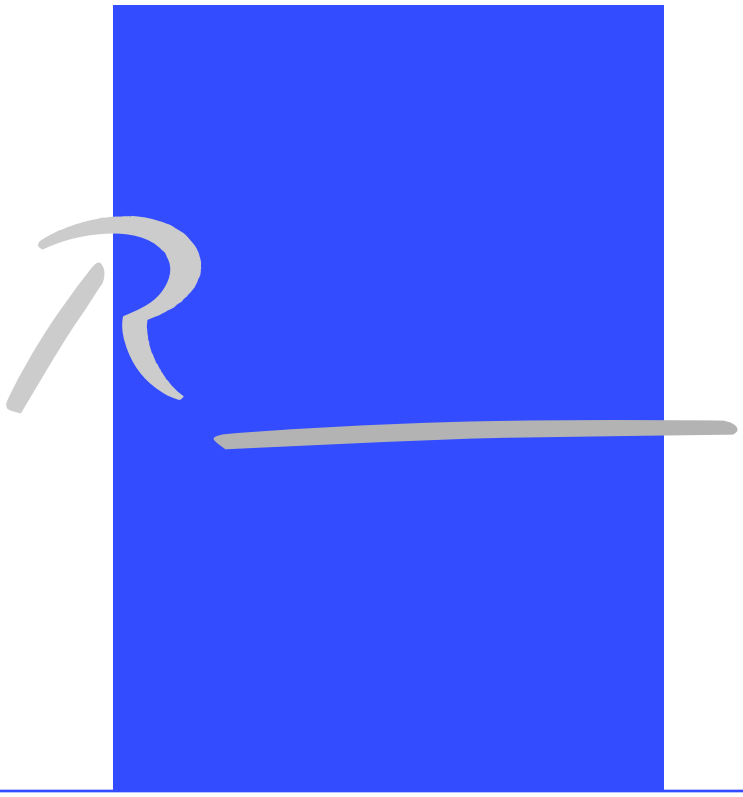
Motif Discovery on Promotor Sequences

Maximilian Häußler and Jacques Nicolas

N° 5714

Octobre 2005

Thème BIO



INRIA
RENNES

Motif Discovery on Promotor Sequences

Maximilian Häußler * and Jacques Nicolas †

Thème BIO — Systèmes biologiques
Projet SYMBIOSE

Rapport de recherche n° 5714 — Octobre 2005 — 136 pages

Abstract: Transcriptional regulation is the mechanism in the cell that controls when and how genes are expressed into proteins. This document gives an overview over current computational approaches that try to predict "motifs" that control the process. Motifs are short degenerate words or patterns within promotor sequences, putative binding sites, which are commonly usually upstream of a gene. The two common approaches in bioinformatics (matching against known representatives and *ab initio* prediction) are presented. For the latter, we describe algorithmical details of most existing implementations and introduce a tool that could simplify everyday work with these programs.

Key-words: promotor analysis, promoter analysis, motif discovery, motif matching, pattern discovery, binding site prediction, transcriptional regulation, bioinformatics, machine learning

En collaboration avec Torsten Schaub, Institut für Informatik, Universität Potsdam, Allemagne

* maximilian@bioinformatics.org
† jacques.nicolas@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex (France)
Téléphone : +33 2 99 84 71 00 — Télécopie : +33 2 99 84 71 71

Découverte de motifs dans les séquences promotrices

Résumé : La régulation de la transcription est le mécanisme principal dans la cellule dirigeant quand et comment les gènes sont exprimés en protéines. Ce document donne une vue globale sur des approches in silico essayant de prédire les motifs qui déclenchent et guident le processus d expression. Ces motifs sont des mots dégénérés, assez courts, parfois appelés patterns , représentant des sites de liaison putatifs dans les séquences promotrices en amont des gènes. Les deux approches courant en bioinformatique (comparaison avec des bases de données et prédiction ab initio) sont présentées. Nous développons plus particulièrement le cas de la prédiction ab initio à partir de séquences, en expliquant de manière détaillée l algorithmique de la plupart des implémentations et présentons un outil qui pourrait faciliter le travail quotidien avec ces logiciels.

Mots-clés : régulation de la transcription, analyse des promoteurs, découverte de motifs, prédiction de sites de liaison, apprentissage automatique, bioinformatique.

Contents

List of Figures

List of Tables

Acknowledgements

I am indebted to my parents who financed and supported this seemingly endless work. Without their understanding, the results would be rather disappointing and the reader would find here a chaotic assembly of 50 algorithms and no software at all.

Apart from them I would like to thank Jacques Nicolas, research group leader of the Symbiose team at IRISA Rennes, who gave me the topic, placed me in his very own office to finish it and invested a lot of time to correct it. All would have been never possible without Torsten Schaub from Potsdam University, who brought me to Rennes. Daniel Fredouille and Francois Coste supported me a lot at IRISA. Nathalie Theret-Bdioui from INSERM kept me motivated through challenging topics.

Thanks to Ute. She followed me around the world and found more than several hundreds grammatical and expression errors in this text.

Mr. Yong-Zhong Horng accepted me in his bioinformatics lab at National-Central University in Taiwan and Sun-Yi Ming, Lin-Feng Mao, and Zhang Zi-Hao had to answer many questions on any topic imaginable. They were the ones that spent a lot of their free time on helping me everyday. Thanks also to all the people from German Academic Exchange Service, the National Science Council of Taiwan and the Coastal Ocean Monitoring Center in Tainan City for the wonderful stay there.

Mr. Wingender from Biobase (Transfac), Klaus May from Genomatix and especially Herbert Mayer from Vienna University repeatedly sent me very detailed answers to my questions via email, which kept me focused to the biological, real-world track while digging through the many algorithmical articles. Bertrand Gakière from the Max-Planck Institute in Golm supplied a training data set of O2-regulated promotors.

This text is accompanied by additional material on the internet:

`http://www.stud.uni-potsdam.de/~haussler/master/` includes the text itself in html and pdf format and a big zip-file containing 150 — and therefore most – articles on motif discovery that were available for free on the web. The reader will also find there a database with the 60+ algorithms reviewed in this work, together with short comments, pdfs, links to implementations and downloadable programs. This is an online-version of Table 2.1 on page 32, but easier to use and to update. All data there can be changed and restored by anyone, similar to a Wiki-Web.

`http://www.bioinformatics.org/jannotatix` is the website for the open-source software Jannotatix as presented in chapter 4, together with instructions on how to run it on different operating systems, how to download source via CVS and a description of its file format.
This text was set with the KomaScript-classes for LaTeX.

1. Introduction

The intention of this work is to give an overview of motif discovery algorithms. This is an area where many articles have been published recently[1] that all resemble a lot. Like many fields of bioinformatics research, while getting more and more mature, motif discovery is not very fruitful for biological applications if it does not incorporate specific knowledge of the domain. A good idea, a new approach and an efficient algorithm is not enough: If the method wants to be actually used (and cited), it has to take into account biological knowledge, which usually means a lot of tedious reading in biology or bioinformatics, and communicating a lot with biologists.

Therefore, we tried to include as much biological background as possible. On the one hand, to make it possible for computer scientists to follow the assumptions that some newer algorithms make, on the other hand, to derive improvements how to continue from there. So we arrived at adding a lot about motif scanning, since it is older, closer to biology and includes many good ideas that could be incorporated into motif discovery. This enlarged the list of relevant reading even more but we find the result very consistent and easier to follow than the opposite, a context-less start into motif discovery, which would present it like a string-search problem. The reader will see that there are a lot more properties of the strings in the DNA than in a normal text and the "longest common substring problem" is only distantly related to motif discovery.

The thesis includes the most complete list of articles, algorithms and programs in motif discovery covering the last 10 years that is known to the author. Its structure, by coincidence, is almost identical to a recent review, published in late september last year (Pavesi *et al.*, 2004*a*)[2], but remains much more comprehensive, covering many more algorithms and explaining a lot more scoring schemes. At the end of the thesis, conclusions are derived that could guide the design of new discovery algorithms. In addition, we developed a program that, being an interface, is closer to down-to-earth programming than complicated research but can dramati-

[1] The bibliography of this text lists more than 250 references of which more than 100 were published in 2003/2004. Admittedly, quite a few are related to general promoter analysis or motif scanning.
[2] The text of this study was written between July and December 2004

cally simplify working with and comparing different motif discovery and scanning algorithms.

If you are a biologist that already knows a lot about binding sites, then you might consider skipping the complete chapter 1, whereas its tables are still interesting in our opinion. Even without a solid background in transcriptional regulation, you can safely ignore the first three sections and go to page 9. In the case that you are not working with micro-array data or don't have many similar promoters at hand, don't worry about chapter 3 with its formulas, you won't need both anyways.

For computer scientists without a profound understanding of biology, the first sections try to start as gently as possible. For the general biological introduction, we used the well-known *Molecular Biology of the Cell* (Alberts *et al.*, 1994) but any similar molecular cell biology textbook should serve the purpose.

Depending on their preferences, computer scientists might be surprised to find few formulas in the following chapters. This is simply because the text is a review, that is trying to extract the main ideas and not a new piece of research. (Liu *et al.*, 2004*b*) is a Masters Thesis with the same topic as this one, written at the same time. However, it opted to design yet another algorithm. The result has a worse performance on artificial sequences and is only compared to one other program on real examples.

In the context of bioinformatics, we do not consider algorithm design an art in its own right, but rather would like to see every step guided by previous research or biological examples. As mentioned before, this means a lot of literature mining and reading, a task that has, as we hope, become easier with this text. At least for the following months...

1.1. Biology of gene expression regulation

1.1.1. RNA polymerase transcribes genes

According to the central dogma of molecular biology, DNA first gets *transcribed* into RNA, which is subsequently *translated* into proteins. This is called *gene expression*, a process which is well understood today – we know pretty well how DNA codes for proteins. However, the central dogma doesn't say anything about *why* and *when* a gene is transcribed, we know little about the control of expression.

The situation is familiar to a programmer who has been working with a piece of uncommented source code (possibly his own): Although he knows very well to which result a command leads, it is very difficult to tell when exactly it is executed. This depends on special kinds of keywords in the source code: Control statements

like if...then, repeat...until, etc. They all influence the order in which commands are executed. Trying to figure out the final result for a given input can be more complex than the task the program has to solve. Actually, complexity is more a question of the control statements' structure than the number of commands.

In gene expression, the most important mechanism for determining whether or not a protein will be produced is control of transcription initiation (Lodish *et al.*, 2000). Here, the production of RNA is regulated. Only when certain conditions are met (not too cold, not too hot, only if male, only if muscle-cell, etc.) should a gene be decoded. Of course, the final decision can be made at any time during expression: Even after translation, the cell could still decide to throw away the current product. However, this is not a very efficient way to control the production of proteins. Therefore, the most reasonable and important stage where expression is regulated is at the level of transcription initiation, the very first step of protein synthesis.

To guide the process, we find, attached to every gene on the DNA, a detailed plan *under what conditions* and *how strongly* to transcribe it into RNA. In higher organisms, with their many cell types and complex reactions to (rare) external events, this plan is much more difficult to read and understand than the control structure of a computer program, because there is not a handful but quite probably thousands of different "control statements". Worse, most of them remain still unknown today.

1.1.2. Transcription factors direct RNA polymerase and trigger transcription initiation

RNA polymerase is the protein that binds to the DNA, opens the two strands and – starting from a location called *transcription start site* (TSS) – walks along the sequence and copies it into RNA. Regulation of transcription thus is the control whether or not RNA polymerase is able to bind to a sequence around the TSS. It turns out that the polymerase can not bind to DNA on its own; instead, it relies on a set of many *transcription factors* (TF), special proteins that recognize certain short sequences on the DNA. The polymerase then (often indirectly) attaches to the transcription factors and the whole complex together is affine enough to guarantee binding. See Figure 1.1 for the most well-known example from the bacteria *E. coli*, the factor called cAMP-CAP and the polymerase, taken from (Lodish *et al.*, 2000). The recognizing proteins are also called "trans-acting" logic, in contrast to the short sequences they bind to, which are called "cis-acting" logic, commonly referred to as

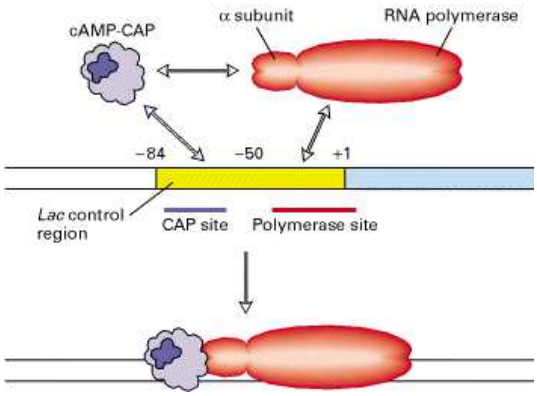


Figure 1.1.: Cooperative binding of RNA polymerase and cAMP-CAP

binding sites. Other terms - mainly used in bacteria - are *activators* or *repressors* for the trans-acting proteins[3] and *operators*, for binding sites.

The cAMP-CAP-activator recognizes a particular operator/binding site, which is roughly TTTACAC+17bp. of any nucleic acid+TATGTT. The whole signaling process is now easy to imagine: When the situation of the cell necessitates the transcription of a certain set of genes (which are all marked with a particular sequence), it releases the cAMP-CAP complex, which is transported to the nucleus, recognizes its sequence next to the target genes, attaches to it and by doing so directs the RNA polymerase to the right place on the DNA.

Of course, for higher organisms, this example is too simple, and one transcription factor will not be sufficient to encode a complex logic of "control statements". Eukaryotic cells do rarely react to changes in the environment; instead, their genetic program has to deal a lot with development, they specialized to become a certain type of cell and then produce only a few, very specific proteins until cell death. Imagine a gene that is only transcribed, if the conditions "cell is part of head"+"not in bone or flesh'+"no neighboring cell is a hair-producing cell' are met. Here at least three factors have to be involved, as a "signal" which genes are to be transcribed and which aren't.

For pro- and eukaryotes equally there are also ubiquitous transcription factors which always bind to the RNA polymerase and to almost every gene. For this study they are not of much interest, it should just be noted that they are called

[3] In eucaryotes, TFs that do not bind to the DNA directly, but are layered on top of other factors, are called mediators or co-activators (Zvonimir Marelja, personal communication)

Model Organism	Estimated number of genes	Estimated number of TFs	Genes per TF	Source
E. coli	4300	240	17	(Robison *et al.*, 1998*b*)
S. cerevisiae	6100	203	29	(Harbison *et al.*, 2004)
A. thaliana	28000	1500	18	(Riechmann *et al.*, 2000)
D. melanogaster	14000	1000	14	(Levine & Tjian, 2003)
H. sapiens	25000	3000	8	(Levine & Tjian, 2003)

Table 1.1.: Number of genes and transcription factors and their relation

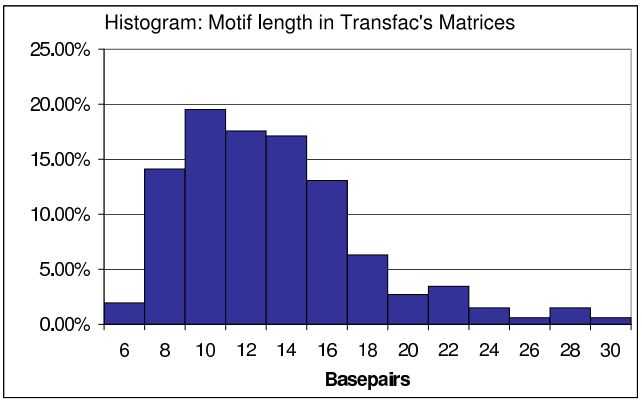


Figure 1.2.: Length of a major part of known binding sites
For this graph, we parsed all 690 binding site matrices included in Transfac 7.4 Professional, representing about 14,800 individual binding sites, with BioPerl and the TFBS Perl modules from (Lenhard & Wasserman, 2002).

general/basal transcription factors (bacteria: σ-factors) and will be needed all the time but therefore do not seem to fulfill very specialized regulation functions. Our focus here lies more on *inducible* or *upstream* factors.

1.1.3. Binding sites composition regulates the strength of transcription

As noted before, the role of TFs is to recognize 5 to 20bp-long short sequences (Rombauts *et al.*, 2003) on the DNA, called *binding site*, *motif* or *box*. Yet the way *how* to bind to the double-helix must be different from those that might already be known to the reader. Restriction enzymes, for instance, bind to DNA as well, but

always to a unique and exactly defined sequence that is used to direct their activity very specificially. On the other hand, the very long and redundant protein-protein binding domains often even continue to function although some nucleotides are inserted or completely changed. The small binding sites on the DNA are a mixture of both, they do not allow spacers but can tolerate certain kinds of substitutions[4]. A given site, TATAAT, for example, might still be fully functional if changed to TATTAT, but not anymore if changed to TAATAAT.

If some letters are changed, the site will be recognized less easily by the transcription factor and the *strength* of expression will be lower. (Stormo, 2000) This is a consequence of the way factors bind to the DNA and recognize their motifs, since due to the double-helical-shape of the gene, many TFs can not get close enough to every base-pair to recognize it. See Figure 1.3 (taken from (Purves *et al.*, 1998) for a well-known example of how a protein binds to the major grooves of the DNA. Resembling a clothes-peg[5], its two helical parts almost "touch" three to four nucleotides on each side. However, they don't get close enough to some of the nucleotides in between. Even when arbitrarily changed, they would not influence binding of the transcription factor at all. If we change some of the more important positions towards the left or right, then the factor might not recognize the binding site any more. But it could be also possible that it recognizes the site *better*, leading to a stronger expression of the gene or *worse*, leading to a weaker expression of the gene. So this "blurred" binding specificity, which complicates the search and analysis of promoters, is really a biological necessity: Not all binding sites are to be recognized with the same probability as not all genes should be transcribed at the same rate. If evolution is allowed to mutate a base-pair without loss of function then sooner or later this base-pair will change. But evolution will also find the optimal binding strength by mutating important nucleotides until the gene is expressed at an intensity that helps the organism to survive.

Not all TFs look like this example: Different protein classes exist, called zinc-finger, homeo-domains, leucine-zippers, etc.. Yet it is visible from illustration 1.3 that the transcription factor's shape leads to a certain set of possible binding site, as some part of the TF are closer to the DNA than others.

---

[4] This is in fact not completely true. There are examples with flexible spacers. (Owen & Zelent, 2000) and some motif databases allow gaps at least in their scoring model (Quandt *et al.*, 1995). However, to the best of the authors knowledge, no recent motif scoring or discovery model includes flexible gaps. Quoting (Sinha & Tompa, 2002):" Insertions and deletions among binding sites are uncommon, again because of the fixed structure of the factors' DNA-binding domain." (Cliften *et al.*, 2003) claim that most gapped motifs wrongly associate ubiquitous sequences with binding sites.

[5] clothes-pin = clothes-peg = Wäscheklammer = pince à linge

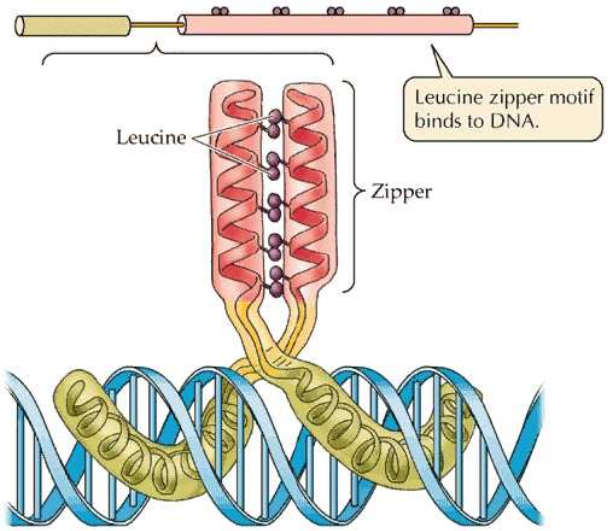


Figure 1.3.: A leucine-zipper transcription factor bound to DNA

1.1.4. A promoter is a set of binding sites upstream of TSS

As the transcription factors help to initiate transcription, it is not surprising that they bind to a region that is located close to the start of the gene. This region is called *promotor*[6] and its function "is to mediate and control initiation of transcription of that part of a gene that is located immediately downstream of the promotor" (Werner, 1999). So promotors are the place on the genome where binding sites for transcription factors are located. Like genes, their delineation becomes difficult, since in higher organisms sites can also be located downstream and up to 100 kilobases upstream of the start of transcription, whereas in bacteria or yeast they are more adjacent and compact. (Levine & Tjian, 2003)

Therefore, we differentiate between a *core promoter* and *enhancers*: The core promoter is located close to the start of transcription and absolutely needed to bind general transcription factors with the RNA-polymerase. The *enhancer*-parts of a promotor (also called: proximal $<$ 200bp and distal[7] $>$ 200bp (Werner, 1999)), which

[6] Unfortunately, in some publications, "promotor" has the same meaning as "binding site", see (Praz *et al.*, 2002) or (Blanchette *et al.*, 2000), which is very unusual, according to most introductory books about molecular biology that the authors has seen.

[7] These distances vary from author to author, some still call 500 bp "proximal" (Zhang, 2002)

Model Organism	Size of Genome	Perc. of noncoding DNA	Binding site database	>75 % region	Source
S. cerevisiae	12 Mbp	28 %	SCPD	500bp	(Harbison *et al.*, 2004)
A. thaliana	125 Mbp	65%	Plantcare	800 bp	(Lescot, 2002)
D. melanogaster	180 Mbp	85%	Transfac DGDB	1600 bp	(Lescot, 2002)
H. sapiens	3.2 Gbp	97%	Transfac	1200 bp	(Elkon *et al.*, 2003)

Table 1.2.: Model organisms and their promotors

The organisms yeast, plant, fruitfly and man and their >75%-region where binding sites occur. This region has been determined by searching the binding site database from the list for the exact position of its sites relative to TSS. (at least 75% of the Transfac's binding sites for drosophila are 1600bp upstream of TSS, for example). For details on the databases, see Table 1.3. Binding sites in all databases might be biased towards certains kinds of transcription factors that are interesting to the respective research community. However, for us, there is no way to deal with this kind of bias at the moment. We can just hypothesize that longer sequences, like 10 kb, would be advantageous to catch the more specific elements.

can be found farer away and influences the strength of binding, level of transcription or conditions when to do the transcription. The long distances are possible since DNA can be bent by certain transcription factors (e.g. the TATA-binding protein (Forget *et al.*, 1997)), and so a site far away on the sequence can be very close if the double helix is viewed in 3D. See Figure 1.4, taken from (Purves *et al.*, 1998). As we cannot possibly describe everything around a gene and search for promotor elements everywhere, biologists seem to concentrate on particular regions upstream to TSS where most binding sites are concentrated. Table 1.2 gives an impression of the these regions' size. For bacteria, the distance is considerabely smaller, roughly 60-100 bps. ((Lodish *et al.*, 2000), p.358).

The part of the genome we are mainly interested in is therefore the 0.5-2.0 kbp stretch upstream of the genes' transcription start site. Unfortunately, for complex eukaryotes, the TSS is difficult to find. This is a fundamental problem for everyone working on human promotors as a large number of sequences might contain no single binding site and further processing can be completely flawed, making motif discovery futile. This problem is discussed in Appendix B on page 105 and algorithmically very different from motif discovery.

It leads to the consequence that most programs which analyze promotor regions on human, mouse or rat genomes do not extract the 1200 bp suggested by table 1.2 but use 2000 bp instead, in order to catch at least part of an incorrectly placed

promotor. Actually, since binding sites can be placed much farer away upstream, so extracting more would pose many advantages but as it will also increase the noise and false predictions, 2000 bp seem to be a commonly accepted compromise at the moment.

When speaking about promotors, a simple kind of classification from biological literature should be noted: The distinction between TATA-less and TATA-promotors. This means that the corresponding promotor either lacks or contains a binding site for the TBP-transcription factor, which recognizes roughly the sequence "TATAAT". It is the most well-known basal binding site also called *Pribnow-Box*, described in the early 80s; it has special mechanical properties (Fukue *et al.*, 2004), 20% of promotors in yeast seem to contain it (Basehoar *et al.*, 2004) (Cliften *et al.*, 2003) and roughly 32% of those in humans (Suzuki *et al.*, 2001). Similar prominent basic or basal elements like TATA include INR, the GC-Box (88% (Suzuki *et al.*, 2001)) or the CAAT-box(60%), which all bind general transcription factors. A subset of them can be found in almost every promotor.

1.1.5. Promotors arrange sets of transcription factors that form huge complexes in high eukaryotes

As promotors group together a plethora of different binding sites which in turn attach to each other, to co-activators or the RNA-polymerase directly, it is obvious that the resulting assembly of various proteins has a rather complex structure and the influence of every part on the whole is difficult to predict. See Figure 1.4 for an example from (Lodish *et al.*, 2000) that shows an activated transcription initiation complex for the TTR-gene in mice. The difference to Figure 1.1 is striking.

This could be the reason why complexity of an organism is not directly reflected in the number of transcription factors. It is more the number of combinations that count. Although a human cell should be much more difficult to regulate than that of yeast, the number of transcription factors did not increase with genome size (see Table 1.1, fourth column). Admittedly, other explanations might also be possible, like basic network laws[8].

As for binding site combinations, another idea is never explicitly noted in biology books; the different forms of how transcription factors can interact suggests an anal-

[8] In networks, the number of possible links increases exponentially with the number of nodes. Here, genes could be seen as nodes and transcription factors as links.

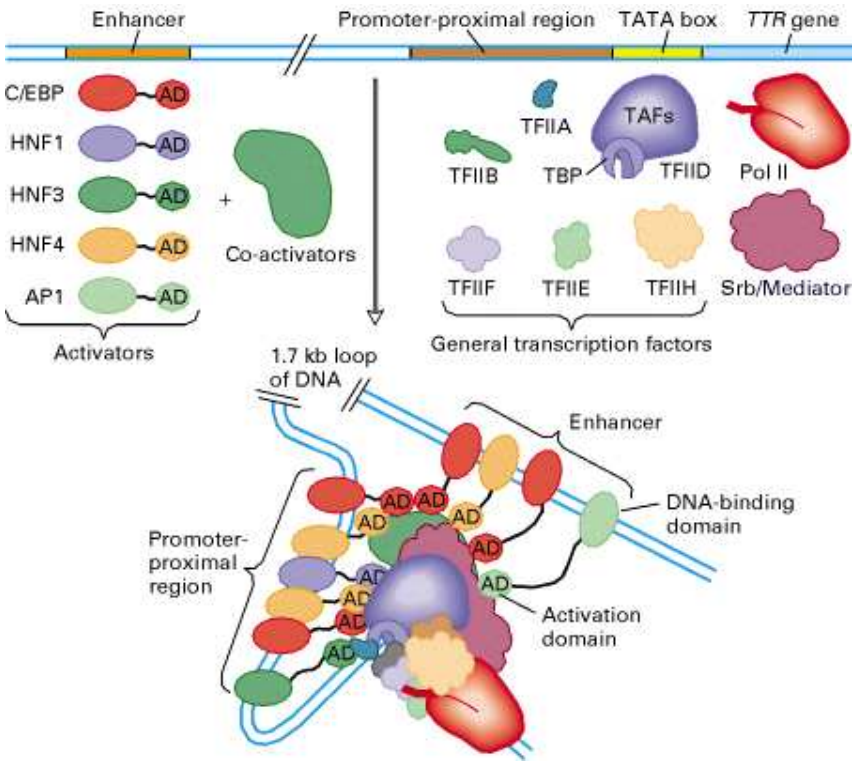


Figure 1.4.: Eukaryotic Promoter Complex with TFs

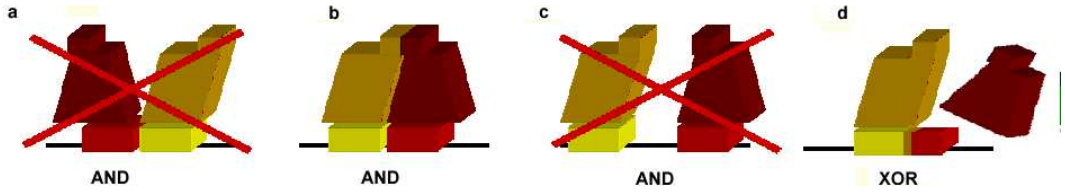


Figure 1.5.: Transcription Factors' ways of cooperation and analogies to boolean operators

ogy to logical expressions[9]: If two factors bind to each other and are only recognized by the polymerase if both are present, then they could be seen as connected by an AND-relation. If one of the TFs blocks the other one, e.g. when their binding sites overlap, they encode an XOR (the blocking TF is then called a repressor). If either one can induce transcription by binding the polymerase, we could speak of an OR-relation. This kind of logic is written on the DNA, by virtue of the right arrangement of binding sites. And since transcription factors can be layered on top of each other, they can also express a concept like brackets in algebra (imposing a certain order on the boolean expressions).

It is hard to find biological examples for all of these cases, but the idea illustrates that – at least theoretically – there is little limit to the complexity of control that could be encoded using binding sites as all basic boolean operators are present.[10] Therefore, "any gene will typically have its very own pattern of binding sites for transcriptional activators and repressors ensuring that the gene is only transcribed in the proper cell types and at the proper time during development" (Pedersen *et al.*, 1999). So predicting the function of a gene from its promotor-sequence alone will be close to impossible, given the number of transcription factors involved and their mostly unknown side-effects on each other. Another conclusion is that small deletions or even point-mutations in a promoter can have huge effects on the transcription factor assembly. The modification of one single base can turn upside down the whole transcriptional program of a gene, as it may trigger – in rare cases–the binding of other factors. We know from computer simulations that common single point mutation rates are capable of producing adequate genetic variation to create the complex binding sites known from real genomes. (Hengen *et al.*, 2003, Stone & Wray, 2001)

1.1.6. Other factors influence transcription

There is a large percentage of the genome, once called "junk-DNA", that is not dedicated to encode proteins, but serves other purposes, like regulation. See Table 1.2 for an impression of the huge amount of still largely uncharacterized sequence data. With the huge genomes in higher eukaryotes like plants or humans, there is only

[9] The analogy of a "molecur flip-flop" has been suggested by (Hengen *et al.*, 2003). They describe overlapping Fis-binding sites in the E. Coli genome which (in the nomenclature) of this study would represent AND-relations.

[10] Other forms that resemble XOR: Attachment by one transcription factor to another one, in a way that prevents it from binding to the DNA.

very little space constraints for the placement or number of binding sites and also enough place for other elements that regulate transcription.

DNA is not linear in the cell but wrapped around histones. Their shape can prevent the RNA polymerase from binding, if the core promotor is hidden in wrapped DNA by special proteins. Other proteins can deacetylate the histones, which makes them positively charged and binds the double-helix tighter, preventing general transcription factors from attaching. The opposite has also been described, hyperacetylation, which facilitates binding of transcription factors. Another way to "silence" a promotor is modifying the DNA itself, by methylating[11] the cytidine nucleotide which binds the deacetylase protein just mentioned, which in turn will repress transcription (for a recent review, see (Felsenfeld & Groudine, 2003)). In short, there might be other factors, invisible from the sequence alone, that influence transcription. So even if all possible effects of the many combinations of factors and sites mentioned in the last section were known and verified by *in vitro* experiments, many of them might still never touch an RNA-polymerase *in vivo*.

As we do not know enough about the intricate interplay of transcription factors since experiments take a lot of time, we cannot predict binding sites from sequence alone with enough reliability (see below) and do not know the other factors inhibiting or directing transcription, the documented examples of annotated binding sites for a given set of promoters are very rare. The authors assume that at the time of writing this renders ideas futile that try to predict function of promoters with formal languages, binary operators or grammars, since there is only very little data to train these algorithms on. A good starting point might be the genome of yeast, for which a collection of *all* binding sites has been assembled recently. (Harbison *et al.*, 2004)

1.2. Binding site match in silico

1.2.1. Binding sites are hard to find with biological experiments

If a transcription factor is already known and available, then an amplified piece of DNA can be cut into many pieces and mixed with the factor. With a Southern Blot, pieces can be filtered by weight, extracting only those that are bound by a protein. If a certain segment of DNA is known to be bound by a protein, then the DNA-region in consideration can be "digested" by DNAase, an enzyme that removes all DNA that is unbound. Having removed the transcription factor, the rest can then be sequenced. After all this, only one single binding site is known.

[11] Adding a methyl, CH_3, group

With newer methods like SELEX (systematic evolution of ligands by exponential evolution), many different short random DNA-oligomers can be created together, are filtered, re-amplified, filtered again in cycles and finally sequenced. By this, many different sites can be obtained in parallel. As there is criticism that SELEX does not model the natural selection (Shultzaberger & Schneider, 1999), DNA microarrays (Bulyk *et al.*, 2001) might be the best solution for high-throughput binding site determination. For this assay, many double-stranded short oligomers are bound to a glass plate (like in RNA microarrays), the proteins are then poured onto the plate. Binding is signaled with flourescent markers and many bound sites can be determined with just one assay.

However, all this just gives *in vitro*-results. To find out if a certain position on the genome really contains a *functional, active* binding site, one has to pass a lot more time in the lab. The promoter-DNA of cells has to be extracted, the binding sites in question deleted and the whole sequence put back into the genome and the cells (transfection). Then the cells are grown, possibly until a new organism is developed, e.g. a full arabidopsis plant. If the plant changes or the organism is different or dies, then it is obvious that the deleted sequence was a binding site for a protein.

A quicker way to determine binding sites is Chromatin-Immunoprecipitation (ChIP), but it is only available if both the transcription factor is known and an antibody (=a protein that binds to the TF) for it is available . For ChIP, all proteins are fixed to the DNA in living cells, so they remain at their current location. Then the DNA is sonicated into many small fragments, the parts with the TFs are filtered out using the antibody and the DNA is separated and sequenced. These sequences can be treated with DNAase and sequenced afterwards to discover the real binding sites.

As a quicker alternative, they can be poured onto a DNA-microarray that contains all n-long oligomers of the sample sequences. The DNA will then hybridyze to the spots that resemble the reverse-transcribed binding site. This technique is called *ChIP-on-Chip*, *ChIP-Chip* or *Chip*^{2}. Its results are a set of sequences that are n bp long, where n can exceed some hundred basepairs. (Liu *et al.*, 2002) Therefore, in the final step, the sequences from the microarray have to be searched for a word they have in common. (An application of motif discovery, as introduced in section 1.3, but on data that are a lot less noisy and much clearer and simpler to analyze than promoters.)

Sometimes an antibody is not known or difficult to obtain. For one of the cheaper ChIP-kits, Active Motifs ChIP-IT, for example, only 10 antibodies are readily available. Other companies like Acris sell more than 22,000 different ones. But when we do not know the transcription factor, several antibodies have to be tried. Referring

to table 1.1, we see that 1500 transcription factors are assumed to be present in a given plant cell, so the applications of ChIP are limited as a biologist can not try all known antibodies.

Therefore, if the TF is unknown, one has to delete parts of the promotor region in order to find the binding site. This means iteratively cutting out a sequence of 100 bp for instance, then transfecting the cell with it, possibly followed by deletions of smaller regions, e.g. 10bp. By repeating this, one is finally able to drill down to the active binding site and hopefully also the transcription factor bound to it. In the end, the site will be part of a published article. To avoid duplication of work, published sites will be picked up by special companies or academic teams who will feed the site, gene and transcription factor into databases. These can later be consulted by other biologists before starting their own experiments on a gene (see Table 1.3).

1.2.2. Known binding sites in databases are mainly represented as consensus strings or weight matrices

As mentioned in section 1.1.3, binding sites are still recognized even if some of their nucleotides were not perfectly conserved during evolution. Yet over the whole length of the binding site, not all positions change equally. After multiple sites that are recognized by a transcription factor have been identified and are put into databases (see Figure 1.7[12], table a), one can count the number of times a certain nucleotide at a position has been found. The result is called a frequency matrix, table b). The nucleotide most often found at a position could be seen as representative and is used to construct a "consensus"-string for the binding site (see also Figure 1.7, part c) However, it also evident that positions exist where several bases can occur[13] and some that really never change the nucleic base. This can be explained by the physical factor-DNA interactions, as the TF is sometimes not able to distinguish an A from T or other letters, especially when the DNA is far away from the protein's binding domain, yet another protein comes in the way or the nucleic acids are hidden inside a bent double-helix structure.

Biologists know this very well since the 70s (Pribnow, 1975), but until about 10 years ago, there was no commonly accepted method to distinguish between "strong" and "weak" bases. Coloring the letters is sometimes used (Lodish *et al.*, 2000) or

[12] Adapted from (Wasserman & Sandelin, 2004): The original table d is missing some minuses. In addition, the pseudocounts used to calculate table d are not $\sqrt{8}$ as stated in the paper but roughly 0.711, according to Mathematica's solver...

[13] Still, "it is often a transition (that is, the substitution of a purine for a purine, or a pyrimidine for a pyrimidine) rather than a transversion" (Sinha & Tompa, 2002)

Name Article	Focus on ...	Number of Sites / Matrices	Scan Algorithm	URL
Transfac Prof. 8.3 (Matys *et al.*, 2003)	Mammals	14,406 735	MatInspector Match/Patch	`http://www.biobase.de` (EUR500/year)
Genomatix Library No article	Mammals	N.a. 600	MatInspector	`http://www.genomatix.com` (EUR360/year)
TRRD (Kolchanov *et al.*, 2002)	Mammals	9500 N.a.	BLAST	`http://wwwmgs.bionet.nsc.ru/mgs/gnw/trrd`
Transfac Public 6.0 (Matys *et al.*, 2003)	Mammals	6,627 336	MatInspector Match/Patch	`http://www.gene-regulation.de`
ooTFD (Ghosh, 1998)	Unknown	∼6000 N.a.	TFSiteScan ProfileScan	`http://www.ifti.org/ootfd/`
Plantcare (Lescot *et al.*, 2002)	Plants	435 435[a]	MatInspector	`http://intra.psb.ugent.be:8080/PlantCARE/`
Place (Higo *et al.*, 1998)	Plants	ca. 400 N.a.	Signal Scan	`http://www.dna.affrc.go.jp/PLACE/`
Jaspar (Sandelin *et al.*, 2004)	Mammals	? 111	Consite	`http://jaspar.cgb.ki.se/`
DPInteract (Robison *et al.*, 1998*a*)	E.coli	800 55	BlastN	`http://arep.med.harvard.edu/dpinteract/`
SCPD (Zhu & Zhang, 1999)	Yeast	530 25	?	`http://cgsigma.cshl.org/jian/`
DGDB (Mohr *et al.*, 1998)	Fly	? ?	?	disappeared?

Table 1.3.: Some binding site databases and their motif search algorithms

[a] Plantcare does not seem to make a difference between site and matrix. We do not know how MatInspector is supposed to find the best five conserved positions in this case.

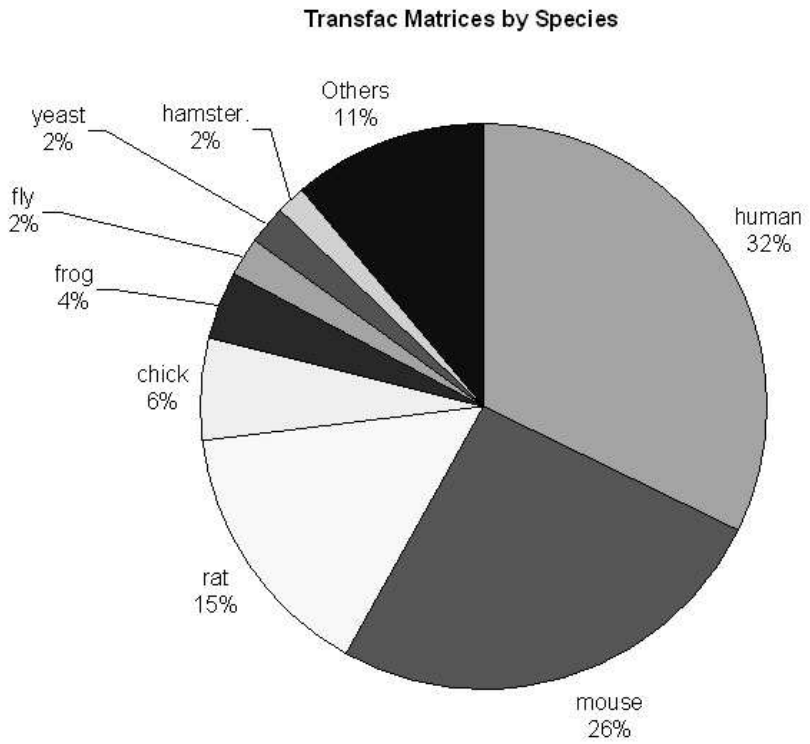


Figure 1.6.: The share of model organisms in binding site database entries
Transfac Pro 7.1 MATRIX-table was parsed with Perl-Scripts. If a matrix was created from more than one organism, it was counted several times. These data represents roughly 14,800 individual binding sites.

bold font **TA**TAA**T** (Alberts *et al.*, 1994). Better methods include the RegExp-like patterns already known from protein domain descriptions, like in `TATAA[T/A]`. For degenerate consensus strings IUPAC[14]-abbreviations are often applied to represent ambigous positions which would, for our example, lead to `TATAAW` (W representing a T or A).

The now generally accepted model of how to store all acceptable binding sites for a given TF is, however, a table that states for every position four counts, one for each of the letters A,T,C,G (see Figure 1.7, table c). To the user, the simple counts are usually presented as in part d of figure 1.7, obtained by calculating frequencies from table c and taking the logarithm of it. Zeros are replaced with pseudocounts first to avoid $log(0)$. These likelihoods are also called weights (Staden, 1984). They are calculated according to the following formulas:

$$p(b,i) = \frac{f_{b,i} + s}{N + 4s}$$

$$W_{b,i} = \log \frac{p(b,i)}{p(b)}$$

where $p(b,i)$ denotes the probability of nucleotide b at position i, $f(b,i)$ is the count/frequency of nucleotide b at position i, s is the pseudocount, $W_{b,i}$ is the resulting weight and $p(b)$ is the background-probability of nucleotide b.

The *position weight matrix* can also be represented graphically as a *sequence logo* (table f), with letters scaled proportional to their frequency, stacked on top of each other. At the same time, the whole vertical size of the stack represents the information content or conservation of a position: The higher the information content (IC) for a position the better a transcription factor can recognize a nucleotide at this place of the binding site. If there is only one base, the information content is 2 bits. It will decrease as the data for a position gets more "noisy" and will finally reach 0 bits if all nucleic acids are equally probable (the formula for the IC is given on page 20). With the logo/matrix approach, one can easily distinguish strong from weak positions and can even score the degree of conservation, as opposed to consensus strings.

Therefore, the original string-based consensus sequence model should be avoided (at least the one without ambiguity symbols), as the differences to the weight matrix accumulate with every letter: Only 14 out of 291 binding sites really are identical to `TATAAT`, all the other ones have mismatches at the "weak" positions. (Schneider,

[14] International Union of Pure and Applied Chemistry, there are 11 symbols for every possible combination of `A,C,T,G` the most common one is N, for "aNy" nucleic acid

a

Site 1	G	A	C	C	A	A	A	T	A	A	G	G	C	A
Site 2	G	A	C	C	A	A	A	T	A	A	G	G	C	A
Site 3	T	G	A	C	T	A	T	A	A	A	A	G	G	A
Site 4	T	G	A	C	T	A	T	A	A	A	A	G	G	A
Site 5	T	G	C	C	A	A	A	A	G	T	G	G	T	C
Site 6	C	A	A	C	T	A	T	C	T	T	G	G	G	C
Site 7	C	A	A	C	T	A	T	C	T	T	G	G	G	C
Site 8	C	T	C	C	T	T	A	C	A	T	G	G	G	C
	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Source binding sites

b

B	R	M	C	W	A	W	H	R	W	G	G	B	M

Consensus sequence

c Position frequency matrix (PFM)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	0	4	4	0	3	7	4	3	5	4	2	0	0	4
C	3	0	4	8	0	0	0	3	0	0	0	0	2	4
G	2	3	0	0	0	0	0	0	1	0	6	8	5	0
T	3	1	0	0	5	1	4	2	2	4	0	0	1	0

d Position weight matrix (PWM)

A	−1.93	0.79	0.79	−1.93	0.45	1.50	0.79	0.45	1.07	0.79	0.00	−1.93	−1.93	0.79
C	0.45	−1.93	0.79	1.68	−1.93	−1.93	−1.93	0.45	−1.93	−1.93	−1.93	−1.93	0.00	0.79
G	0.00	0.45	−1.93	−1.93	−1.93	−1.93	−1.93	−1.93	−0.66	−1.93	1.30	1.68	1.07	−1.93
T	0.15	−0.66	−1.93	−1.93	1.07	−0.66	0.79	0.00	0.00	0.79	−1.93	−1.93	−0.66	−1.93

e Site scoring

0.45	−0.66	0.79	1.68	0.45	−0.66	0.79	0.45	−0.66	0.79	0.00	1.68	−0.66	0.79
T	T	A	C	A	T	A	A	G	T	A	G	T	C

Σ = 5.23, 78% of maximum

f

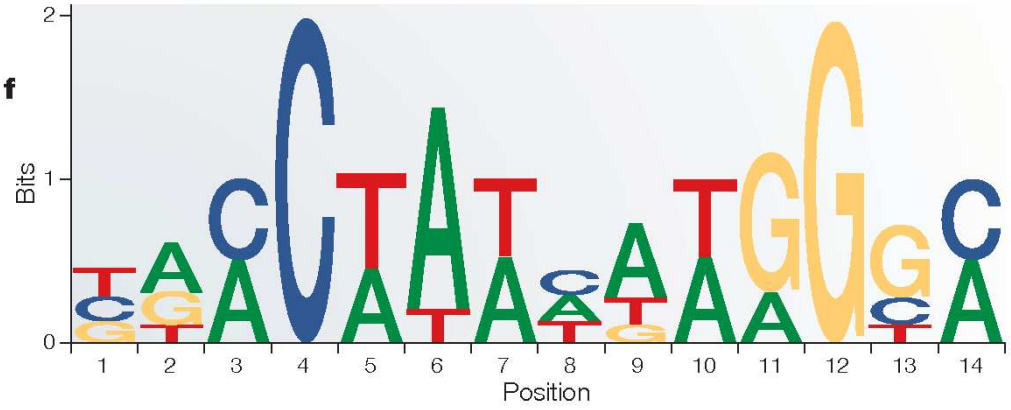


Figure 1.7.: From validated binding sites to the motif weight matrix

2002) Even with IUPAC-wildcards, consensus strings can suddenly change a lot, as soon as a rather unusual site is added, see table b of figure 1.7, where site 8 has an extreme effect on the final consensus string.

On the other hand, consensus strings will not die out, as they can be simply written down on paper by anyone. It is impossible to search Google for matches to a matrix but every search engine accepts strings. And when we try to make a list of all possible motifs, an exhaustive enumeration of all consensus strings poses no unsolvable problem, whereas trying all different matrices is clearly impossible[15].

But the weight matrix represents the different binding energies of the transcription factor to every base: The better a site conforms to the weight matrix, the more often the factor will bind to it and vice versa. It is a model that reflects better the subtle mechanisms of physical binding which allow for very fine-grained control over the gene's strength of expression. (Stormo, 2000) Binding is not a yes/no-decision—every letter contributes to the process and every letter with a different weight. Just as every binding site contributes to the final binding specificity of the transcriptional complex with a different weight. If consensus strings are used, these weights and their information are lost.[16]

Apart from the matrix representation, more expressive models have been elaborated as well, for it has been suggested that the positions in a binding site are not independent of each other. This is a typical application for machine learning algorithms, like hidden markov models or bayesian networks, both of which have been applied (King & Roth, 2003) (Barash *et al.*, 2003) to the problem, or more elaborated matrices (Zhou & Liu, 2004) (for which details will be given in section 2.3.9). Nevertheless, it seems that the more complicated the model gets, the more training data would be needed. But at the moment — given laborious biological experiments — only a couple of sites have been collected for every annotated transcription factor.

[15] Even if only a precision of 0.1 is needed, there are still $(10,000)^l$ possibilities for all matrices of length l, as opposed to the consensus strings, where there are only 4^l different ones without and 15^l with IUPAC-symbols.

[16] If word-based models do not incorporate the contribution of every single base into their model, their decision of binding/not binding ignores these weights. Example: When doing matching to a grammar, in a word-based model, two binding sites might be equally probable, as they only differ in one letter from the consensus. However, in one case this letter could be located at a position that has a high weight. In the other case, at a position with a low weight, i.e. the mismatch would have no consequence. Both together might indicate binding of a certain complex, for which a grammer might indicate other factors that bind nearby. However, all conclusions would be invalid, as the first binding site is not an active one. Completely different grammatical rules should be applied instead, but the system would rather be mislead by the many false positives.

So for most of the known binding sites, the probability weight matrix is not too bad an approximation, as shown by (Benos *et al.*, 2002)).

The company Genomatix further groups matrices in their database, unlike Tranfac, into families. The sorting is done by a program which compares all pairs of matrices and their information content. (according to Dr. Klaus May)

1.2.3. Calculating information content of a probability weight matrix

Information Content (IC) is well-known to all computer-scientists from Shannon's information theory. The height of the stacked letters in a sequence logo is the IC for one position. In Figure 1.7, part f, the Information Content of position 4 is the full 2 bits, so the transcription factor can distinguish all four bases on this position. As all binding sites contain the letter C on the 4th position in table a, we see that evolution did not allow other nucleotides there and the TF can gain the complete 2 bits of information from this nucleotide, which leads to a very high letter C in table f. With $b \in \{A, T, C, G\}$ and f_b as the frequency of base b at the current position, known from the frequency table (see Fig. 1.7, table c), we can calculate:

$$IC_{pos} = \sum_b f_b log_2 \frac{f_b}{p_b} \tag{1.1}$$

The variable p_b is the frequency of b in the whole genome. It is needed to make up for biased genome compositions, found in some organisms, like *S. cerevisiae* (A+T: 64 %).

Equation 1.2 is known as the information content relative to background, also called *Kullback-Leibler-Distance* or *relative entropy* (Hertz & Stormo, 1999). A nice property of all log'ed-likelihoods is that the probability of a sequence can be calculated by simply adding over the individual likelihoods of all positions instead of multiplying them. Given a matrix of length len, its total IC is therefore:

$$IC_{matrix} = \sum_{pos=1}^{len} IC_{pos} \tag{1.2}$$

IC_{matrix} can be seen as a measure for motif conservation: The better a motif is conserved, the more its IC will be close to $2 \cdot len$ bits. It can also be interpreted as the information gain of a transcription factor after having recognized the site. For example, with a genome of 12 Mbp and 1 binding site to be identified we would expect information content to be close to 20 bits ($log_2(12,000,000)$). If the IC is lower, the motif becomes too weak and wrong sequences would be incorrectly recognized. If

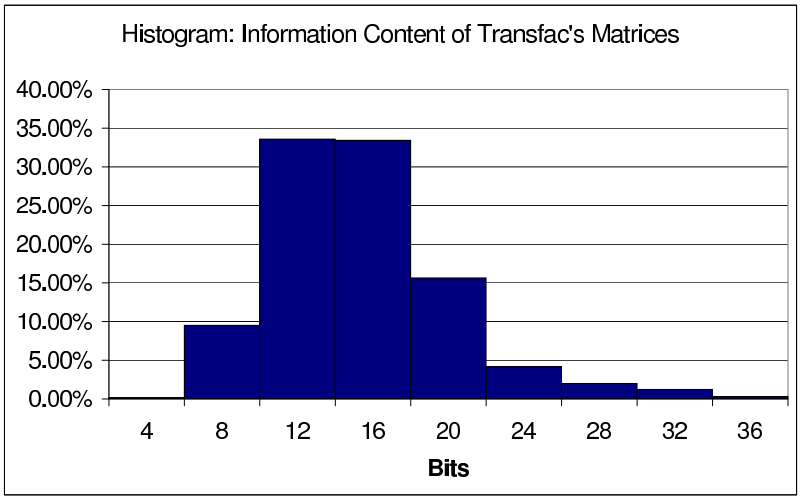


Figure 1.8.: Information Content of matrices in Transfac 7.4 Pro
For details on this diagram's data preparation, see page 5

the IC is higher, the organism is wasting energy on a too specific transcription factor. See Figure 1.8 for an impression what values of IC_{matrix} can be found in Transfac.

The IC is also related to thermodynamics, as it measures the relationship between the strength of the physical interaction between a protein bound to its motif and the protein bound to any arbitrary DNA sequence (Hertz & Stormo, 1999)[17] This is the physical binding strength between the two molecules[18].

The IC could be viewed as the amount of information the transcription factor can distinguish. For computer scientists, this should be easy to grasp via the following analogy: The genome is something like the memory. The transcription factor is an object that searches – or better: transmits – an address of a certain type, like the address bus in a computer. The addressing scheme is the weight matrix. And the binding-site is something like an "address": It identifies a particular place. When addressing one single position on the genome (a rare case, just for illustration here), we need as many possible combinations in our adressing scheme as there are

[17] And is a good example why Shannon referred to Boltzmann/Gibb's works when establishing his information theory, as the latter is only a generalization of rules already known from thermodynamics.
[18] Yet another way to arrive at the Information Content is a statistical model, assuming that a sequence is either generated by the background or by a weight matrix. Then the maximum log-likelihood that a given substring was generated by the weight matrix is the information content (times the substrings occurrences). (Eskin, 2004)

basepairs on the genome. Therefore, the IC of a matrix should correspond to the expected IC which depends on the genome size and the number of occurrences of a binding site ($IC_{exp} = log_2 \frac{size}{occurrences}$). The theory seems to be valid for bacteria, where there is really only one transcription factor (activator) "searching" for its sites (Schneider *et al.*, 1986). However, since eukaryotic transcription factors are combinatorial by nature and never occur alone, we would have to sum over the IC of all motifs involved in the regulation of a eukaryotic gene to do similar estimations there.

1.2.4. Sequences can be scanned for matrices

Since there are databases that list binding sites with matrices or consensus strings (see table 1.3), the logical next step are programs that scan a promoter for these. A piece of promoter, e.g. GTATCGTATATTGGC can be scanned for a consensus sequence TATAAT. This could also be called *site scanning*, *site matching* or *scoring a sequence* (Quandt *et al.*, 1995) against a matrix. In this example, one site is present, though with one mismatch. However, we know from literature that the consensus TATAA*W* is a better description of the Pribnow-Box, as its last nucleotide is only weakly conserved and allows for T and A. With the new consensus now there is something like a "half" mismatch in the example sequence. For a more complex example, see Figure 1.7, where the consensus BRNCWAWHRWGGBN is derived from some experimentally verified binding sites. We could now scan parts of a promotor (e.g. a substring of the same length, TTACATAAGTAGTC) against this motif to find its instances and count how good the hit fits the search string (here: four mismatches, several "half" mismatches).

TESS (Baxevanis, 2002), for instance, is one of the scanners for consensus sequences. It calculates a score on the full and "half" mismatches – the better an instance corresponds to the ambiguous symbols, the higher the score.[19]

But since weight matrices better indicate how well an instance conforms to the motif, we should use the whole matrix from part d of Figure 1.7 instead of the consensus, as no information is lost there. If a given promotor contains the 14bp-long sequence TTACATAAGTAGTC, then, by summing over all weights for these letters from the matrix in part d, its substring would (see part e) score 5.23 bits. This seems to be the scheme applied by Signal Scan (Prestridge, 1991), one of the first

[19] Actually, for the case of TESS, this score is the same as converting the ambiguous symbols to a weight matrix and then calculating normal log-ed probabilities again. See next paragraph.

implementations, and the scanning part of the more recent Consite (Sandelin *et al.*, 2004) (see below).

If the weights were not given as log-likelihoods but as probabilities p_i, the product of the probabilities of all letters from a string would give the total probability that the string was generated by the matrix.

$$P_{string} = \prod p_i \tag{1.3}$$

There are numerous ways to improve on this simple scanning, as it has been noted, especially by the people working on the Transfac database, that it is usually sufficient to count only the well conserved positions of a motif. Therefore, MatInspector (Quandt *et al.*, 1995), still the most used scanner at the moment, looks only at the five most conserved nucleotides (the core), as it considers the others too unreliable (taking also into account that they are not calculated from many too many sequences anyway). Match (Kel *et al.*, 2003) is an improvement that sums the weights of the core and the other positions in two separate scores. A hit is only signaled if both exceed a certain threshold that is specific for every sequence and developed by the matrix database curators. To derive these, positive sequences that contain the motif and negative, second-exons regions (supposed to contain no binding sites at all) are scanned. Then the thresholds are set to minimize either the false positives (# of found instances on the negative set) or the false negatives (# of missed instances on the positive set).

OTFBS (Zheng *et al.*, 2003) is a very similar motif scanner but tries to reach certain predefined p-values for the thresholds. PRIMA (Elkon *et al.*, 2003) and the methods in (Marino-Ramirez *et al.*, 2004) set thresholds by scanning HMM-generated background sequences instead as a negative set and estimate p-values from these. MotifViz (Fu *et al.*, 2004) is a web interface with four different methods of scoring matrix hits. It contains Clover (Frith *et al.*, 2004), which uses the whole genome as a background set to assess if a motif is over-represented in a set of sequences[20]. Pobo (Kankainen & Holm, 2004) uses all upstream sequences of the selected model organism, a positive and a negative sequence set, and tries to find the most discriminative list of motifs to separate both from the background. QPMEME[21] (Djordjevic *et al.*, 2003) uses a different binding energy model and therefore a completely new scoring method, claimed to be superior to standard weight matrices for unspecific motifs. MotifScanner (Aerts *et al.*, 2003), with matrices from

[20] And is therefore very similar to the scoring for motif discovery programs, introduced in later sections.
[21] **Q**uadratic **P**rogramming **M**ethod of **E**nergy **M**atrix **E**stimation, there is no relation to the program called MEME as presented in section 2.3.1

Transfac, applies an HMM to estimate the background frequency for a certain substring, filtering out motifs that are too recurrent in the background sequences. It also gives a P value for each motif hit which is calculated from a Bernoulli model. Its probability is derived from the number of hits in the EPD. As a result, this kind of P value helps to distinguish ubiquitous, basal putative sites from rather specific ones.

With the advent of databases of cis-elements and whole genomes, more and more genomes are completely scanned for potential binding sites, like *E. coli* (Thieffry *et al.*, 1998), *A. thaliana* (Davuluri *et al.*, 2003) and human/mouse conserved regions (Loots *et al.*, 2002), and results can be readily viewed by just typing in a gene's name. Then its promotor together with all hits against previously established motif models are shown. This makes the scanning much easier to use for biologists.

1.2.5. Scanning for known binding sites leads to too many false positives

The big drawback of site scanning, striking most biologists working with it for the first time, is that there are by far too many matches. The stretches of DNA that are recognized by the transcription factors are too short to occur anywhere specificially on the genome. A pure search for one single weight matrix matches every 500bp. (Wasserman & Sandelin, 2004) Databases now include up to 600 matrices (see Table 1.3), so in a putative promoter segment of 1200bp, a hit occurs virtually everywhere. It is not a deficiency of the algorithms, as these sites really could bind a TF *in vitro*, as proven by (Tronche *et al.*, 1997). But in the real cell due to the current context and the combinatorial nature of regulation (section 1.1.6), only 0.1% of the putative sites will be functional, according to (Wasserman & Sandelin, 2004), who assert that "essentially all predicted transcription-factor binding sites that are generated with models of individual TFs will have no functional rule" and call it the "futility theorem".

Therefore, to improve the results of scanning, especially for higher eukaryotes, a combination of two different binding sites already known (Kel-Margoulis *et al.*, 2000) ((Werner, 1999) reviews a couple of similar approaches) or several copies of the same binding site that occur close together, can be searched on the genome, see for the fly (Rajewsky *et al.*, 2002*b*, Berman *et al.*, 2002) or for the human genome (Sharan *et al.*, 2004). This reduces false positives considerably, so much that biologists could successfully verify the few hits by experiments, see (Berman *et al.*, 2004, Markstein *et al.*, 2002, Halfon *et al.*, 2002).

The most fruitful and promising approach, however, will be the use of comparative genomics. If promotors of several close species are aligned, scanning can be re-

stricted to conserved regions as they are the ones that are most susceptible to contain meaningful patterns; see Figure 1.9. There exist some successful cases for human-mouse alignments, but for many applications, evolutionary distance between these species seems too far for this approach. (Pennacchio & Rubin, 2003). Given enough sequence data, even scanning for known matrices is not needed any more; a multiple alignment alone might be sufficient to discover regulatory elements. Called *phylogenetic shadowing*, this approach would necessitate the sequences of 4-6 monkeys to uncover most of the human binding sites by conservation alone. (Boffelli *et al.*, 2003) Sites found by this method were recently discovered to be functional, but some remain absent even in the very close primates. (Frazer *et al.*, 2004) Still, unraveling binding sites was one of the five reasons for the chimpanzee sequencing project. (Olsen et al, 2002) Given that there are currently 10 different types of drosophila flies already being sequenced, this type of analysis might be feasible very soon[22]. Its effectiveness has been already shown using only two fruitfly genomes. (Sinha *et al.*, 2004)

In *E.coli*, three species are sufficient to uncover 74% of binding sites, if they have the right phylogenetic distance. A crude resume of this distance is: neither too far nor too close. (McCue *et al.*, 2002) But since binding site evolution seems to be negatively correlated with the number of genes regulated by a transcription factor (Rajewsky *et al.*, 2002*a*), the very best phylogenetic distance could depend on the specific factor researched. An interesting observation was made by (Rajewsky *et al.*, 2002*a*): If a binding site has one position changed in one species, which lowers its score and the binding strength to the protein, this does not lead to another position changing in turn to increase the affinity back to the old level. Therefore, binding affinities can sometimes change and lead to to differences in gene expression. Changes are not correlated during evolution.

Similar estimations were done for yeast, see (Moses *et al.*, 2003, Chiang *et al.*, 2003, Kellis *et al.*, 2003, Kellis *et al.*, 2004). The findings are roughly the same as in *E.coli*: 1) The lower the IC in the databases for a position, the more will nucleotides change at this place. This is the same in cross-species/single gene and single species/cross-gene comparisons. 2) Binding sites are not perfectly conserved but show a certain profile of mutations reflecting their IC profile. An alignment alone is not the optimal approach to uncover regulatory elements but constitutes one part of it. The authors suggest to incorporate these results in motif discovery programs (see section 1.3).

[22] see NHGRI website, list of ongoing projects

Given the findings from flies, bacteria and yeast, the rather distant mouse/human/rat-comparisons will not lead to optimal results when searching for binding sites. However, a lot of tools have appeared on the web which do exactly that, as this is all the sequence data available at present for mammals that could be used for comparisons. One example is Consite (Sandelin *et al.*, 2004), that compares upstream regions from human/mouse orthologous genes. The approach seems to be working in some cases, detecting 68% of verified binding sites in conserved regions and reducing false positives by factor ten (the results, however, had still a false positive every 5-6 basepairs (!) (Lenhard *et al.*, 2003)). CONFAC (Karanam & Moreno, 2004) is a similar database for mouse/human comparisons, just like CORG (Dieterich *et al.*, 2003), they both list human-mouse conserved elements and hits to Transfac. CONFAC is retrieving orthologs from the genome databases at UCSC and Ensembl and applies the same scanner, but also searches for clusters of binding sites, just like CREME (Sharan *et al.*, 2004). SMASH (Zavolan *et al.*, 2003) is a software run from a web-interface, which extracts orthologs from RefSeq, extends the 5' regions using ESTs[23] and then scans with a subset of Transfac matrices. TraFac (Jegga *et al.*, 2002) starts with a BlastZ-search for orthologs, scans with all Transfac matrices and displays a nice syntheny-view of both results. Theatre (Edwards *et al.*, 2003) does not find orthologs automatically but includes some more tools, like a Gene Scanner, Repeat Masker and a CpGplot. CONREAL (Berezikov *et al.*, 2004) uses a slighly different approach, scanning first and then trying to find non-overlapping, identical hits across the aligned sequences. rVista (Loots *et al.*, 2002) is similar while being one of the most well-known tools for comparative genomics, as it also displays a conservation plot on the sequence that can be used to manually estimate the validity of the Transfac matches.

Since for all cross-species comparisons multiple alignment algorithms become more and more important, (Stojanovic *et al.*, 1999) did a benchmark of five older ones on one type of data and found that the results did not differ much. (Pollard *et al.*, 2004) did a similar benchmark again with heterogenous data sets and found that results differ a lot and depend on evolutionary distance: Similar species are better analyzed with global alignment programs, farer ones demand local alignment software. DiAlign is a mixture that performs well in both contexts. The very recent Tracker-program (Prohaska *et al.*, 2004) was not evaluated, but might perform similar to DiAlign, given both algorithms similarity.

Stubb (Sinha *et al.*, 2003) is a logical result of these comparisons: A motif scanner that incorporates cross-species information and searches for clusters of conserved

[23] See appendix 2, for an explanation of the term EST and the problem of promoter 5'-extension

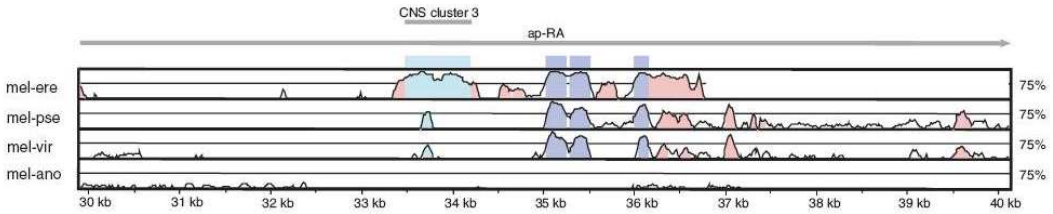


Figure 1.9.: VISTA-conservation plot of the AP region
The organisms compared are D.melanogaster, D.viriliis, D.erecta, D.pseudoobscura and A.gambiae. Conserved regions roughly correspond to binding site scanning hits, verified partly by transfection experiments. The figure was taken from (Bergman et al., 2002).

binding sites with a HMM; it has been applied to two drosophila genomes (Sinha *et al.*, 2004). Using combinations of sites *together* with comparative genomics is the next step in motif scanning. The authors of this study is convinced that the following months will see a couple of similar algorithms appear in the journals, like (Grad *et al.*, 2004), that search for *groups* of regulatory sequences *across genomes*.

1.2.6. Coexpression implies coregulation implies common binding sites

From the last paragraphs it has become obvious that if a model for a binding site is available, promotor sequences can be searched for it, though plagued by a plethora of false positives. However, sometimes biologists have a set of genes at hand that are co-regulated but share no common known pattern.

This is often the case since microarray data began to emerge. RNA microarrays are glass or plastic plates onto which short pieces of RNA are affixed. If the contents of a living cell that are marked with fluor dyes is then poured onto the glass, fragments of RNA that were just being transcribed from DNA will attach to their reverse-complements on the glass. With a scanner and appropriate software, researchers can identify which genes were just being read by the cell.

If the cell had been put into an oven before the experiment, for instance, many of the genes might be related to heat-shock. The assumption is that genes are regulated by a common mechanism (a transcription factor, for example) if they are expressed

at the same time[24]. And if they are regulated by the same transcription factor, most of them should contain the same binding site.

However, microarray data are not very reliable and therefore very noisy. There is vast literature on the topic of extracting halfway valid information from microarrays and, like in motif discovery, methods and opinions compete with a lack of accepted benchmarks. See (Murphy, 2002) for a review of problems when postprocessing coexpression data. It is important to keep in mind when analyzing toy-world examples like in the motif discovery assessment (see below) that real-world data include, as a matter of fact, a lot of wrong sequences.[25]

Even if data processing is done properly there are some reasons why, even if two genes are clearly coexpressed, they are not regulated by the same factor and don't share a common motif. Imagine the case that geneA activates a transcription factor, which in turn activates geneB. Although geneA and geneB are expressed at almost the same time they won't share a specific binding site, see (Helden *et al.*, 1998) who is citing the diauxic shift in yeast cells as an example where a lot of different factors are involved, which makes their target genes share only very few common patterns. Examples have been described (human muscle, sea urchins) where "there is often more than one promotor structure to achieve similar expression patterns" (Werner, 1999).

Apart from coexpression, other ways can be used to detect genes that might share common binding sites: (McGuire & Church, 1999)

1. Conserved operons: Operons in procaryotes are groups of genes that share one promotor, so they are necessarily coregulated. If these genes are far apart in other microbial genomes they are good candidates for coregulation.
2. Conserved divergently transcribed genes: In eukaryotes, some genes – especially in *S. cerevisiae* – are divergently transcribed[26] and therefore share their upstream region. So if these genes are scattered on the genome in other species, they are also good candidates for coregulation.
3. Functionally related genes: Proteins that function together in the cell are likely to be coregulated. They can be extracted by searching for common phyloge-

[24] If there is already a hint of what the mechanism could be, one can also search for genes that – over some time – follow the expression pattern of a given transcription factor. (Zhu *et al.*, 2002)

[25] Add to this the error induced by wrong promotor extraction (see appendix B) on mammalian genomes. Both errors together make any kind of analysis very difficult for promotors from more complex species.

[26] *Divergently transcribed* in this context means that a given gene is read from right to left and its immediate neighboring gene is read from left to right. The direction of transcription between both diverges and the promotor in between them is often bi-directional.

netic profiles or from metabolic or functional pathways. Also genes that are known to work together in one organism are candidates to fulfill a similar function in other organisms.

4. On one single gene: Some genes, notably the drosophila developmental enhancers that control body growth in a fruitfly-embryo, are packed with binding sites. There are so many sites together that it is feasible to find them by applying motif discovery to one single upstream sequence alone. (Papatsenko *et al.*, 2002)

1.3. Motif Discovery Searches for Words Generated by the Same Weight Matrix

Selected genes that are suspected to share a common pattern can be searched by what is called "motif discovery", a quest for the most unusual word of a set of special promotor sequences. They are usually selected by microarray experiments. The intention of motif discovery is to pick out the pattern that is recognized by the yet unknown transcription factor which supposably activates the set of genes.

Sometimes whole genomes are searched for overrepresented binding sites. (Galas *et al.*, 1985) (Bussemaker *et al.*, 2000*a*) (Vanet *et al.*, 2000) (Marino-Ramirez *et al.*, 2004) But the discovered elements are then ubiquitous, not very specific to a co-regulated group and are usually already known to biologists (like the TATA or GC-Box).

Motif discovery is not a local alignment, since it compares more than two sequences and most traditional alignments will fail due to the surrounding noise. The problem has already been described as "multiple local alignment" (Hernandez *et al.*, 2004), but we don't follow this terminology any more, as the promotors can be completely different: Only some small fragments are identical and an alignment can include segments from one and the same sequence. In an extreme theoretical case, they can all lie on one single sequence (which poses some visualisation problems for graphical interfaces and makes not much sense biologically, of course). However, the notion of "alignment" is still present, as the final result of the algorithms is a list of positions where the putative motif occurs.

Applying motif discovery is not a new application in bioinformatics, the first implementation we know dates from 1985 (Galas *et al.*, 1985). One of the first experiments that used gene-expression data gained from microarrays for motif discovery are the well-known yeast cell cycle regulated genes. (Spellman *et al.*, 1998) (Zhang, 1999) Spellman's data is still among the oldest and most used and cited for analyzing microarray-results, whereas less well-known for its binding site search. The

expression data are exactly the reason why transcriptional regulation, motif discovery and -scanning in the following became such an interesting topic in bioinformatics: Starting at around 1998, many biologists had RNA-microarrays at hand that produced huge lists of co-regulated genes that couldn't be explained with already known binding sites from a literature research. Therefore, many computer scientists started working on this topic.

In the following, we call the hidden, unknown putative model of the binding sites recognized by a transcription factor a "motif". It is usually represented as a position weight matrix or a consensus string. The matches for this motif on the sequences will be called *motif instances*, *putative sites* or simply *sites*. Biologically active and validated sites will be called as such. A run of a motif discovery program will therefore result in a list of motif instances, the alignment, from which can be derived the motif as a weight matrix or a consensus string.[27]

[27] Some algorithms directly return the motif in the form of a matrix, some algorithms only produce a list of instances.

2. Algorithms in binding site discovery

The following two pages list a couple of published algorithms that apply motif discovery on promoter sequences to discover yet-unknown binding sites (*de novo* prediction). In this chapter, we will try to classify them and explain the basic process that they follow. Since references are already given in the table, they are omitted from the textual descriptions. Although not exhaustive, the list is the most complete one that we have found until now. Lack of time prohibited us from describing the following ones: Pattern-/ProfileBranching (Price *et al.*, 2003), Tsukuba BB (Horton, 2001), Mermaid (Hu, 2003), Mermaid (Markstein *et al.*, 2004), (Li *et al.*, 2002), (Wu *et al.*, 2004), STARS (Mancheron & Rusu, 2003), SDDA (Gupta & Liu, 2003), TreeGibbs (Yper *et al.*, 2003), Superposition (Shinozaki *et al.*, 2003), ELPH at `http://www.tigr.org/software/ELPH/index.shtml`, Cubic (Olman *et al.*, 2003), GRAM (Takusagawa & Gifford, 2004), ScanSeq (Papatsenko *et al.*, 2002), GLAM (Frith *et al.*, 2004), BiPad (Bi & Rogan, 2004), BioOptimizer (Jensen & Liu, 2004), FMGA (Liu *et al.*, 2004), MISAE (Sun *et al.*, 2004), PRUNER (VijayaSatya & Mukherjee, 2004), RegulatoryTrees (Phuong *et al.*, 2003), CONVERGE (from the supplementary documentation of (Harbison *et al.*, 2004)), NestedMICA (Down & Hubbard, 2005). They form a good topic for a follow-up study.

2.1. Deterministic approaches: Word-counting

2.1.1. YMF, Oligoanalysis: Pattern-Driven String Enumeration

To discover words that occur more often than usual in a sequence, the simplest, brute-force approach is to make a big list of all possible motifs of length l and then scan along the input sequences, incrementing the counter of every string in the list that is found. Possibly also incrementing all counters of strings with a mismatch of one or more bases to the found one. This necessitates 4^l bytes of memory but given current computing power and RAM sizes is still doable for common motif lengths. This approach was certainly difficult to take when motif discovery started (Galas *et al.*, 1985) but today the situation changed. Storing all words of length 16, which should be sufficient even for mammals, would take only 4GB of memory and since the RAM could then be used as a big table, access to every byte would be reasonably

Algorithm Name	Model	Search	Scoring	Publication	Software	University, Country	Tested on
by Galas et al.	String	Enum	?	(Waterman *et al.*, 1984) (Galas *et al.*, 1985)	None	US,UCLA	Bac.
by Mengeritsky et al.	Enum,PD	String	?	(Mengeritsky & Smith, 1987)	None	US,Harvard	
by Staden	String	Enum,PD	?	(Staden, 1989)			Bac.
Wordup	String	Enum,PD	χ^2	(Pesole *et al.*, 1992)	None	IT,Bari	EPD
Gibbs Sampler	Matrix	Gibbs	Wilcoxon	(Lawrence *et al.*, 1993)	Web, Src	US,NIH	Proteins
MEME	Matrix	EM	P(IC)	(Bailey & Elkan, 1994)	Web, Src	US,UCSD	Proteins
MACAW	Matrix	Gibbs	?	(Liu, 1994)	Bin:Win	US,Harvard	Bact
CoResearch	String	Enum,PD	IC	(Wolfertstetter *et al.*, 1996)	Bin:SunOS	DE,GSF	Mammals
R'MES	Matrix	?	Markov+?	(Schbath, 1997)	Removed	FR, INRA	None
Oligo-Analysis	String	Enum	Sig(Genome),Pos	(Helden *et al.*, 1998)	Web	BE,Brux.	Yeast
Dyad-Analysis	String,Dyad	Enum,PD	P(Genome)	(van Helden *et al.*, 2000*c*)	Web	BE,Brux.	Yeast
AlignACE	Matrix	Gibbs	MAP,Spec,Pos	(Hughes *et al.*, 2000), (Roth *et al.*, 1998)	Web,Src	US,Harvard	Yeast
Teiresias	String	Convolution	None	(Rigoutsos & Floratos, 1998)	Web,B:Lin/Win	US,IBM	Protein
Yebis	HMM	Enum	χ^2,IC	(Yada *et al.*, 1998)	Web	JP,JST	Human (GSF)
Consensus	Matrix	Consensus	P(IC)	(Stormo & Hartzell, 1989) (Hertz & Stormo, 1999)	Web,Src	US,WU	Bac.
Winnower	String	Graph	Mismatch	(Pevzner & Sze, 2000)	N.av.	US,UCSD	Bac.
SP-Star	String	Graph	Mismatch	(Pevzner & Sze, 2000)	N.av.	US,UCSD	Bac.
Ann-Spec	Matrix	Gibbs	IC	(Workman & Stormo, 2000)	Web,Src	DK,DTU	Bac.
SMILE	String,Dyad	Suffix	z,χ^2	(Marsan & Sagot, 2000) (Marsan, 2002)	Src	FR,IGM	Bac.

Table 2.1.: Some discovery algorithm implementations as of oct 2004

Legend: Dyad: Supports the search for two elements close to each other, PD: Pattern-driven, SD: Sample-driven, IC: Information Content, Suffix: Suffixtree, Prefix: Prefixtree, (Src): Sourcecode bound to a (possibly non-disclosure) licence, z: Z-Score, p(bin|HMM|gen|IC): P value calculated relative to binomial/HMM model or relative to a whole genome or for a certain IC, Distr: Distribution over genome, Spec: Group Specificity Score

Algorithm Name	Model	Search	Scoring	Publication	Software	University or Country	Teste
Verbumculus	String	Suffix	z	(Apostolico *et al.*, 2000) (Apostolico *et al.*, 2003)	Bin:Lin,Sol	US,UCR	Yeast
MobyDick	String	Dictionary	z	(Bussemaker *et al.*, 2000*a*)	None	US,Rockef.	Yeast
by Anderson & al.	String	Enum,PD	Spec.	(Anderson & Parker, 2000)	Perl Src	US,HHMI	Yeast
YMF	String	Str.Enum.	z(Markov),Distr	(Sinha & Tompa, 2000) (Sinha & Tompa, 2002)	Web, (Src)	US,UW	Yeast
Bioprospector	Matrix,Dyad	Gibbs	IC	(Liu *et al.*, 2001)	Web,(Src)	US,Stanf.	Yeast
Co-Bind	Matrix,Dyad	Gibbs	IC	(GuhaThakurta & Stormo, 2001)	Src	US,WUSTL	Yeast
ITB	String	Enum	z(Markov)	(Kielbasa *et al.*, 2001)	None	DE,HU-Berlin	
Mitra	String,Dyad	Prefix/Graph	Back,Mism	(Eskin & Pevzner, 2002)	Web	US,Colum.	Yeast
Spexs	String	Suffix	P(Bin)	(Vilo, 2002)	Web	UK,EBI	Yeast
Mitra-PSSM	Matrix	Suffix	IC	(Eskin, 2004)	None	Israel,HU	Yeast
MOPAC	String	Enum,SD	None	(Ganesh *et al.*, 2003)	None	US,Texas A&M	Yeast
RISA	String	Graph?	None	(Danilova *et al.*, 2003)	None	Rus,RAS	Artif.
Multiprofiler	String	Multiprof.	Mismatches	(Keich & Pevzner, 2002*a*) (Keich & Pevzner, 2002*b*)	None	US,UCSD	Artifi
Projection	String	Hashing	p(Stat)	(Buhler & Tompa, 2002)	Src	US, WUSTL	Yeast
Weeder	String	Enum,PD	Core,IC,z,MAP	(Pavesi *et al.*, 2001) (Pavesi *et al.*, 2004*b*)	Web,Src	IT,Milan	Yeast
DMotifs	String	Enum,PD	TnoM	(Sinha, 2003)	None	US,U. of W.	Yeast
LOGOS	HMDM	EM	IC	(Xing *et al.*, 2004)	None	US, Berk	Yeast
cWinnower	String	Graph	Mismatch	(Liang *et al.*, 2004)	None	US, NASA	Artfi
EC	String	Genetic	Fitness	(Fogel *et al.*, 2004)	Request	US, Eli Lilly	Mam
GMS-MP	GWM	Gibbs	MAP	(Zhou & Liu, 2004)	Bin(Win)	US,Harvard	Mam
MDScan	Matrix	Enum+Gibbs	MAP(hmm)	(Liu *et al.*, 2002)	Src(Lic)	US,Harvard	Yeast
Kamvysselis	String	Enum	see text	(Kamvysselis *et al.*, 2003)	None		Yeast
MoDEL	String	EC	IC	(Hernandez *et al.*, 2004)	CH, SIB	Non-DNA	

Table 2.2.: Some discovery algorithms, part II

Legend: Dyad: Supports the search for two elements close to each other, PD: Pattern-driven, SD: Sample-driven, IC:Information Content, Suffix: Suffixtree, Prefix: Prefixtree

fast, but requires operating systems and compilers that can access an array of this size. This simple method which was termed "pattern-driven" (PD) (Brazma *et al.*, 1998*a*) is therefore limited to single-cellular organisms where motifs are shorter and rarely exceed a length of nine bases. However, it is important to note here that a brute force approach is not completely impossible anymore today when searching for single, isolated patterns.

Enumeration with some biological background as done by Dyad-Analysis, Oligo-Analysis and the algorithm by Manolis Kellis always stressed the fact that most motifs have a well-conserved core of usually 6 nucleotides, which is already exploited by motif scanners (see page 22). This reduces complexity to $4^6 = 4096$ different patterns to try. However, (Helden *et al.*, 1998) also admits that some longer motifs are better detected by more complex algorithms, like Gibbs or Consensus (see below). YMF's authors noted the same, but limit N-wildcard-characters to the middle positions of a motif, a configuration they claim to be have observed in real samples. Dyad-Analysis goes along this scheme by allowing two motifs at a fixed distance from each other, with any characters in between.

2.1.2. Sample Driven Enumeration

Sample- or Sequence-driven methods (SD) (Brazma *et al.*, 1998*a*) (Galas *et al.*, 1985) are similar but try only those patterns that occur in the input sequence exactly (like MotifAnalysis on the arabidopsis.org-website, unpublished) and similar ones instead of all possible patterns. The drawback is that they might miss some well "hidden" pattern, e.g. one that never occurs but of which many slight derivations exist in the sequence. Examples include the algorithms of Staden et al., Wordup, CoResearch and others.

MOPAC is an algorithm that enumerates all substrings of length l of the positive sequences and removes all l-substrings that are present in the negative sequence set – a rather drastic and unusual kind of filtering, which keeps results so small that no further score is needed. Then it calculates the distance between a given string and all others. If the distance is high, this string is eliminated from further processing, based on the assumption that single outliers would increase the computational complexity of the following step. MOPAC finally partitions the set of strings by similarity. They are clustered according to the hamming distance between the patterns until a certain threshold is reached. From every group the final consensus is calculated and reported to the user.

Sample and pattern driven ways are the main basic methods to search for common words. SD approaches are older, since current computing power permits more

complete enumeration. Nevertheless, all other more complex methods work along the input sequence patterns instead of exploring the whole solution space. They just differ in the way to further restrict the solution space inferred from the sequence. The following paragraphs will present these.

2.1.3. Teiresias: String Convolution Approach

This algorithm starts with a list of short patterns that occur in *all sequences*. They are assembled in an initial seeding phase and contain one wildcard character, N, which means any letter. Then the algorithm tries to glue these together ("Convolution phase") to find longer patterns. The extension is done by simply sorting the patterns, more specific first, the ones with more wildcards last and searching for pattern where the prefix equals a suffix of another one. By design, this algorithm assumes that an instance of every motif is present in every sequence (the model known as "One Occurrence per Sequence (OOPS)" from Meme). There does not seem to be any kind of ranking of motifs found nor applications to DNA sequences. Being a rather old approach, however, it is still well-known in the scientific community, included in older reviews (Brejova *et al.*, 2000) and even in a recent application (Montgomery *et al.*, 2004). It is similar to the dictionary based approach.

2.1.4. MobyDick: Dictionary-based

Bearing a rather unsual name[1], this method start with a dictionary D of words. It then looks for concatenated pairs of words p in D that have a low P Value and then updates D iteratively, by adding p to D.

and tests all concatenated pairs in D if they have a low P Value. It then keeps adds unusual ones – with the lowest P values – adds them to the dictionary and iterates. The initial dictionary simply consists of the letters A,C,T,G with their respective frequency as P value. As this method is rather space- and time-intensive, it can not treat strings longer than 8 letters but was applied to rather big datasets, like *all* upstream regions of yeast. The intention is to find unusual patterns by just using the genome sequence. The algorithm seems to be faster than a sample-driven

[1] MobyDick was used to mine the text of the famous english novel for common words. Not very surprisingly, *enough* and *harpoon* are the most recurrent of the longer words (Bussemaker *et al.*, 2000*b*). Ahab (Rajewsky *et al.*, 2002*b*) is a similar approach, citing the MobyDick-article, but instead of assembling words from letters, it tests combinations of complete binding sites in a 500bp-window (see page 24). It is an interesting name: In the novel, Captain Ahab and nearly all of his crew are beaten and killed by Moby-Dick...

complete enumeration as it will avoid exploring rare combinations from the start and can cache the list of occurrences and reuse them for the next iteration. It also optimizes motif length. On the other hand, operating on a genome scale is not very helpful from a biological point of view: Mostly basal binding sites will emerge that are either already known or not very specific to a certain group of genes. They are thus of limited interest to uncover transcriptional regulation.

2.1.5. Consensus: Profile Enumeration

Consensus was developed by the authors of the probability matrix model. It is a rare mixture between matrix and deterministic approaches. It is well-known and supported by many interfaces. Two versions are available: One that searches for fixed-width motifs and one that can determine the optimal width but needs an additional parameter.

Searching for a motif of width w is done as follows: For a substring s of length w, one matrix reflecting this single string is calculated. Then, with this matrix, the algorithm will scan the sequences for a set of other, similar strings. From this set, together with the original s, a matrix is calculated. Using different starting points s, this will give rise to many different matrices, of which only those with the highest IC are kept. From this list, the algorithm iterates; strings are added to the matrices which lead to new matrices that are filtered until a stop condition is met. Stop conditions can be: A limit to the number of sites for the matrix or the number of sites that every sequences has contributed. So the algorithm greedily tries to improve the Information Content of the matrix by adding sites to it.

If the width is not specified by the user, he or she has to use *wconsensus* instead which necessitates an additional parameter, as the algorithm cannot maximize the IC directly which increases linearly with the length of the motif. Therefore, the expected IC and a user-input multiple of the standard deviation of the IC, the "deviation bias", are substracted from the information content, which makes it negative and independant of motif length. The resulting "crude information content" is then maximized.

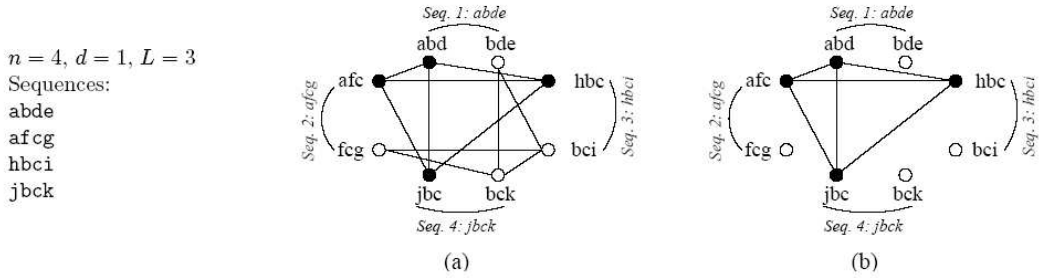


Figure 2.1.: Winnower removing edges from non-cliques nodes

2.1.6. Winnower, SP-STAR, cWinnower: Cliques-based Approaches

Designed to win an artificial contest, the algorithms by the Pevzner group were never really used by biologists[2]. The ideas are still interesting: The graph-based approach makes a list of all substrings of a certain length which are represented by vertices in the graph. Vertices are partitioned into groups corresponding to each sequence (indicated by the textual descriptions *Seq 1* to *Seq 4* in Figure 2.1). Any two of the vertices that are similar to other ones from different partitions are connected by edges. Similarity here means a Hamming distance which is greater than a certain limit.

The algorithm then tries to find *cliques* in the resulting graph. A *clique* is a set of fully interconnected nodes, see Figure 2.1: If string afc is similar to abd and abd to jbc and jbc to afc then there are edges between afc, abd and jbc. They form a 3-clique that could be imagined like a closed triangle. The final goal of the algorithm is to remove all edges that are not part of a *maximal clique*, i.e. a clique with a member in every partition (sequence) of the data set. The idea is that to be part of a k-clique, an edge needs to be part of at least a certain number of extendable-cliques; an extendable clique is smaller than a maximal one. Thus all edges that do not fulfill this criterion are removed. The algorithm thereby starts with low values of k and iteratively repeats this procedure.

See Figure 2.1, taken from (Brejova *et al.*, 2000), where the size of the clique n=4, the hamming distance is 2, the length of the string equals 3 and $k = 1$. The final score attributed to a clique is the number of mismatches between all members of a clique to all other members.

[2] Despite the algorithm's age, the author has not seen any biological applications until today. Winnower only works if every sequence contains at least one motif, an assumption which can not be guaranteed by any biological experiment.

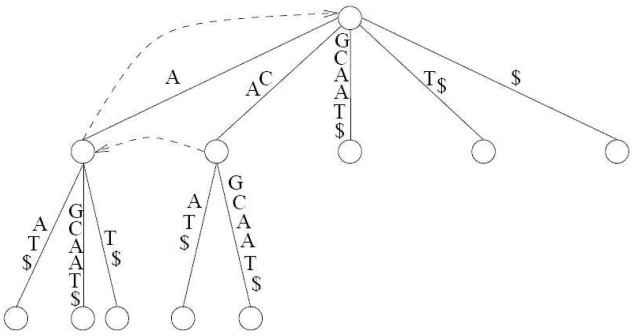


Figure 2.2.: CAGCAAT as a suffix tree

SP-STAR is an improvement to the algorithm in the same publication that uses a sum-of-pairs heuristic to score patterns found. cWinnower (Liang *et al.*, 2004) places edges only when a consensus criterion is fulfilled: For an edge between a and b, all nodes that are connected to a and b are collected into a list. From this, the consensus sequence is calculated. The edge is removed if the number of mismatches between edge and consensus exceeds a certain threshold. Therefore, cWinnower places edges only when a putative motif in the edges' vicinity is sufficiently conserved.

IRSA (Danilova *et al.*, 2003) seems to be a similar kind of approach but the algorithm is difficult to decipher from the two paragraphs devoted to it in the English article. It is reported to score better on artificial sequences than SP-STAR.

2.1.7. SMILE,Verbumculus,Weeder,Mitra-PSSM: Suffix Trees

A suffix tree is a data structure to quickly access the words of a text, a special index. As a result, it can be used to find recurring words (for example in Lempel-Ziv compressions like ZIP or gzip-files[3]) and is therefore an immediate candidate for motif discovery. The tree by itself does not reduce the solution space, it is merely a fast-access version of the sequence-strings; once the suffix tree has been built in memory, the original sequences are not needed anymore. It seems that the use of suffix trees in this context was first proposed in (Sagot, 1998).

A suffix tree is constructed by adding all suffixes with length 1,2,...n of the text into a tree. See Figure 2.2, taken from (Marsan, 2002), for an example which also

[3] For LZ77-based PKzip's deflate-algorithm or for gzip, suffix trees are currently not used, but in an extended version of the LZ77-algorithm, called LZSS

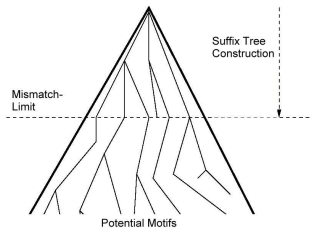


Figure 2.3.: Pruning of the suffix tree in Weeder

shows internal links that speed up tree construction to keep construction complexity linear in time. Using this tree, we can very quickly check if the text contains certain words, CAA, for example. Also words with mismatches can be searched, like nAA[4], CnA or CAn. While traversing the tree to find the string's instances – if spacers are not in the suffix of the string – all applicable branches have to be checked for every spacer.

The authors of SMILE improved over a simple tree. They are limiting the length of the motif to restrict calculations. They added a *structured model search*, where "jumping" in the tree is used to check if a motif of the form $word1 + flexiblespacer + word2$ can be found.

SMILE is building the tree first (Ukkonnen algorithm) and then looking up exhaustively *every possible string* up to a defined length in the tree. This PD-driven enumeration, helped by a suffix tree, is called "spelling" the model (SpellModel/EppelleModel, the original names of the algorithm). It then saves the strings with the most support to a table, adding statistical significance later when outputting it to the user. The FindModels-algorithm (Adebiyi & Kaufmann, 2002), is trying to implement the complete original idea from (Sagot, 1998) and therefore considers edit-distances on DNA. This does not make much sense for binding sites, as explained in the first chapter.

Weeder[5] (Pavesi *et al.*, 2001) also applies the spelling-model from SMILE, but its authors figured out that for higher numbers of allowed mismatches, it becomes very slow. Pevzner's motif discovery competition (Pevzner & Sze, 2000), with the [Width=15, Mismatches=4]-problem, seems to have triggered this finding. So Weeder prunes, from the start all paths that exceed a certain mismatch rate. See Figure 2.3 for an illustration of this, taken from (Pavesi *et al.*, 2001). As an exam-

[4] "n" is a wildcard for "aNy nucleic acid"
[5] to weed = jäten = désherber

ple, take motif length=15 and an error rate=20%: When Weeder is adding the 5. letter, it eliminates at the same time all paths that have more than one mismatch.

Obviously, some binding sites have very concentrated mismatches in their prefix, so they would be unduly eliminated very soon. Therefore, Weeder assumes that it will have missed some instances and lowers the sequence threshold, that is, the percentage of sequences where the pattern has to be found. So in the first round, too many wrong patterns will be extracted from the suffix tree. But Weeder will consequently start a second iteration, using the results of the first one as seeds, and now with mismatches allowed on all positions, just like the original SMILE. In this manner, the algorithm is much faster than SMILE, can treat longer motifs with more mismatches while keeping sensitivity high.

Apart from the algorithms built around ideas proposed by Marie- France Sagot, there are completely different ways to construct suffix trees. Spexs, for instance, does not add every word to the tree but instead builds it on all possible words in breadth-first order (PD), level by level. At every step, it is checking if the current word occurs in the sequence. If yes, all of the found word-positions are put into in the current node of the tree (called *write-only, top-down,* (*wotd*)-method). This is not the fast, linear-time algorithm for suffix tree construction, but the author claims that in practical applications due to memory paging[6] of the operating system it is not much slower than the Ukkonen-method.

Verbumculus (Apostolico *et al.*, 2000) is building a traditional suffix tree, annotates the tree with some derivations of the Z-score (Apostolico *et al.*, 2003) while building it and then extracts only the nodes with the highest scores. It was applied to the dataset from (Helden *et al.*, 1998). Mitra-PSSM (Eskin, 2004) is one of the rare weight matrix algorithms that is using a string-based search approach. It seems to build a kind of suffix tree first and then makes a list of possible weight matrices. Then the suffix tree is scanned for matches to the weight matrices. The results from this scan are subsequently improved with an EM-approach like MEME.[7] Motif discovery with suffix trees also seems to be fast enough to search huge datasets, like complete genomes. (Brazma *et al.*, 1998*b*)

[6] As a suffix tree's memory consumption should not exceed the available RAM, this does not make much sense. Instead of "paging" the author might have rather thought of "caching", an argument actually mentioned in the paper he cited, (Giegerich & Kurtz, 1995), with increased importance in modern processors that rely more and more on jump-prediction and caching.

[7] An implementation was never published. Whereas Mitra-PSSM seems to fit into a new category of its own, the lack of implementation is unfortunate. Mr. Eskin, when I repeatedly requested – since October 2003 – the implementation that was promised for July 2003, does not seem to reply to my Emails on this topic anymore.

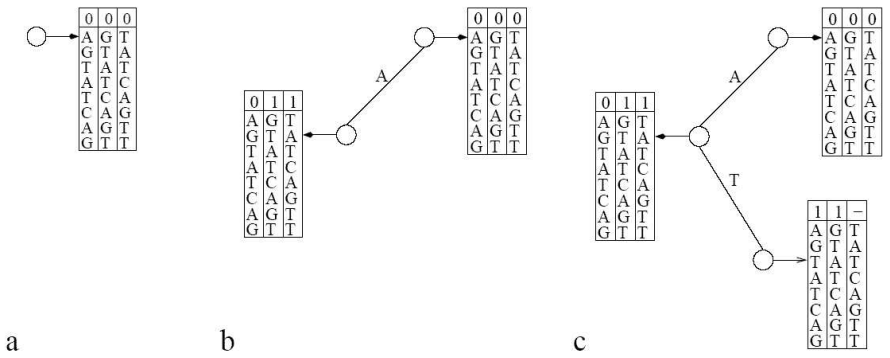


Figure 2.4.: AGTATCAGTT as a prefix tree
(a) shows the root node alone, (b) and (c) show the expansion of the tree for the A and then the AT prefix

2.1.8. Mitra: Mismatch (Prefix) Trees

These approaches consist of two steps where one is similar to Winnower and one similar to SMILE. The first step is building a tree, like a suffix tree. However, the edges represent partial prefixes here, so each path from the root to a node gives its complete prefix. See Figure 2.4, taken from (Eskin & Pevzner, 2002): The root edge (see part a) is empty and leads to a list of all substrings of a fixed length (here, 8 letters). The first edge from it (out of four possible, [a,t,c,g]) represents the prefix A (part b) and leads to a list of substrings that all either start with A or with another letter and consequently see their mismatch count increased by one . From this node there are again more edges leading down the tree. The edge labeled T, for example, leads to a node with all substrings that either start with AT or have their mismatch counts increased again (c). If only one mismatch is allowed, all strings that exceed this mismatch-threshold are marked as deleted, indicated here by a "-" in the mismatch-count field atop the strings. These strings are not expanded in the next step.

In contrast to suffix trees, the prefix version is not a 1:1 representation of the sequences. The construction itself already reduces the search space considerably, leading to sets of substrings which become smaller with every iteration. The filtering step of this first version of MITRA is similar to the improvements introduced by cWinnower, as in every node edges are removed that exceed certain mismatch limits.

With a further modification called MITRA-Graph, the substrings of every node are compared pairwise and a graph identical to Winnower is constructed. As such, all edges that are not part of a certain number of cliques are removed. If there are not cliques left, the node is removed and the algorithm backtracks. If a clique is not removed, its subspace of cliques is explored by examining childnodes in the prefix-tree. This filtering leads to a more efficient version of Winnower, as – according to the authors – the prefix is known while looking for a clique, which keeps the graph smaller. It is not as slow as it might look at first, since the results of the pairwise similarity can be cached and propagated down the tree.

Dyad-Patterns are supported by a pre-processing step in Mitra, where all substrings are concatenated with the substrings n bases next to them and the resulting set is searched. This is definitely not a very efficient way especially for long distances, but worked on biological samples.

2.1.9. Multiprofiler: Searching Neighbourhoods Improved

Multiprofiler searches for all substrings with exactly k mismatches to a given string[8]. This is called k-neighbourhood of the string. This neighbourhood is then scanned for the motif by searching it for recurring "wordlets", a list of letters and their occurrence at certain positions. They are enumerated and counted with respect to the neighbourhood. Wordlets are only processed if they occur at least a certain number of times. All possible wordlets are then checked against their total mismatch distance to the reference segment and the best ones are kept. This looks similar, but is more efficient than trying all possible combinations within a neighbourhood of the best motif.

2.1.10. Projection: Randomized Hashing

Randomized hashing means simply selecting – of an l long motif – x positions that are used as a hash: l-substrings are put together into one "bucket" if they have the same letter at these x positions. This is a preprocessing step that saves a lot of time compared to complete sample driven enumeration. Projection does not rely on this hashing alone, however. Afterwards, it runs five iterations of an EM-procedure like MEME (see page 52) on every bucket, and the resulting motif instances are further refined with SP-STAR (see page 37).

[8] This is doing the same as winnower's initial distance calculation for the graph. Multiprofiler, though, does not need to keep *all* distances into memory, but only those relative to a given word.

2.1.11. EC, MoDEL: Evolutionary Computation

It took quite a while until the first kinds of genetic algorithms appeared in motif discovery. Evolutionary computation is a general way to solve problems that can be applied virtually anywhere: It starts with many randomly or manually selected solutions (called a "population" here), then chooses one of the possible operators which will modify the solutions into offspring solutions, which are then scored. From the result a subset is selected to become the new starting population and the process iterates.

In our case, the initial population is a set of random motifs, each of them being an alignment of possible instance positions. The EC implementation then apply one of the following operators: (a) move putative motif positions to left or right at random, (b) move position to a region with similar GC-content and (c) the "window recombination"-operator: It simply takes two alignments and reassigns their instances arbitrarily to one of them. Then one motif is kept, the other one deleted from the population.

After the random modifications, the score (fitness, see below) of all motifs is calculated, duplicate solutions are eliminated and the motifs are saved if they either did not change much for a number of rounds or a specified time has passed and the algorithm has to end. All collected motifs are then sorted by score and output.

The basic setup of MoDEL is similar, however, the operators are not applied to the alignments but to words instead. Therefore, before the mutation-step, two selected alignments have to be converted to words (consensus of a weight matrix). The two words are then changed by one of the following four operators: (a) Exchange letters between words with a certain probability, one-by-one, (b) exchange all letters on the right of a randomly chosen letter, (c) shift both words until the overlapping ends align well and create a new word, (d) slide one word to left or right and fill up with random symbol to create a new word. The results of the evolutionary algorithm is then further refined with a local search that includes random shift steps of the motif instances (like Gibbs/Meme, see below). The score to find the best motifs is the relative informatinon content. Although MoDEL is applied to DNA-sequences in the article, the assumption "one occurrence per sequence" makes it useless for promoter analysis. TopModel is an improvement to remedy this (to be published, personal communication Y. Mescam).

2.2. Scoring potential motifs

Given a certain motif, the notion of "unusual" or "overrepresented" pattern has to be defined and put into a kind of score. Depending on the search algorithm, it sometimes guides the discovery into the more interesting directions but usually is just calculated and reported after a set of putative motifs has been assembled. Whereas for us it is difficult to follow the intricate details of the statistical models, several main outlines emerged during the years that are presented in the following. Background on statistics can be found in any undergraduate-level book on the topic; the author used (Sachs, 1999).

It is important to note that for most deterministic approaches presented above, the link between scoring scheme and search algorithm is rather weak. Most algorithms could apply any other reasonable score. The statistical null model in the following paragraphs is always a random sequence that does not contain *any* motif. Motifs are not part of it. Their number and structure is compared to the expected values from a statistical null model in order to measure how "unusual" the motifs are. The aim is to calculate the probability of finding a certain motif in background/random sequences and derive a score from it.

On the other hand, the probabilistic algorithms as presented in Section 2.3, already include the motifs on some unknown sequences, embedded within background nucleotides, at some unknown positions. They merely try to iteratively optimize these parameters to identify the real motif instances. They often do not need to calculate separate scores like the following ones, but sometimes report similar, derived measures. We therefore describe first some scoring schemes for word-counting algorithms and then start to explain probabilistic algorithms.

2.2.1. Reference sequences

If a certain word w was found, say, 5 times in the input sequences, it can be extremely difficult to estimate if this should be considered unusual or not. To evaluate whether a word is over-represented, before any calculation can be done, we need a reference to compare with. Some possibilities are:

1. Background sequences: Here, the user can specify a set of background sequences manually or they are already part of the program. Oligo-Analysis/Dyad-Analysis and the algorithm by Anderson et al. uses all upstream sequences of yeast to count how often a pattern occurs.
   Sometimes, "background" is considered not to contain any binding sites at all, to avoid any bias. The selection of this kind of *negative sequences* is not

easy since we don't know for sure if certain regions really contain binding sites. We even don't know if a region is not upstream of another gene as long as the start or direction of transcription of the neighbouring genes are not known. For bacteria and yeast, it seems to be difficult to assess whether a region is upstream of a gene, if it is not annotated yet. An idea from (Washio *et al.*, 1998) was explored by SMILE, MITRA and MotifSampler that extracted regions downstream between two divergently transcribed genes. Since this region is downstream of two genes, they cannot be upstream of anything and will contain only very few binding sites. It is interesting to note that the motif scanner MATCH also needs to set thresholds for its parameters, but is using exon2-regions (Kel *et al.*, 2003) as a reference, which are supposed to contain very few binding sites.

2. Shuffled input sequences: If there are no background sequences available, some algorithms give the possibility to shuffle the input sequences (Gibbs-variants). The idea is that the original A,C,T,G-composition will not be changed by this operation.
3. Background Markov-Model: Improbizer, MotifSampler and YMF do not compare their motifs with the whole background sequences but rather on a very compact Markov-model representation. This has usually the order three, though only MotifSampler gives detailed reasons for this choice, guided by experiments with different HMMs. (Thijs *et al.*, 2001)
4. Binomial/Multinomial models:[9] Most algorithms assume that a DNA sequence was generated by a probabilistic process (i.i.d.) and calculate the probablilty that a certain substring was created by it or simply generate long background sequences with a multinomial process. This applies to most of the scoring schemes presented in the following.

2.2.2. The Probability from a Binomial Model

For the binomial model, we assume that a DNA-sequence is generated by an i.i.d. process, where the next letter depends only on the current one. Every symbol is one of A,C,T,G, every symbol occurs with some probability. Assume, as an example, that all probabilities are identical, with $p = 0.25$, so now we could use a four-sided, tetrahedrical dice to generate a sequence. This is called the *Bernoulli-*, *Binomial-*

[9] The name multinomial might be more appropriate since there are multiple different results (at least four) for every experiment. Bernoulli is still often used in motif discovery papers, as we usually do the calculations only on one word at a time. This word can occur or not, which means only two different outcomes for an experiment.

or *Multinomial-Model* in statistics. In reality, the nucleotides are not all equally probable, but have a certain occurrence rate in the genome that differs between organisms. For yeast, these are, for instance, $p(A) = p(T) = 0.31$ and $p(C) = p(G) = 0.19$. (Helden *et al.*, 1998)

Since we are assuming independance, the probability of a word w is the product of the probabilities of its letters $s_1...s_n$:

$$p(w) = \prod p(s_i)$$

This only gives the probability for one single word w. It does not reflect the number of words found or the length of the sequences. It is not a "P value" yet, a measure of the "unusualness" of obtained words given the sequences; P values will be derived from it, but first another, more natural way to obtain $p(w)$ is presented.

2.2.3. The Probability from Reference Sequences

The binomial model does not fit a noncoding DNA-sequence well. Long stretches of poly-A, poly-T and AT-runs have been described (Helden *et al.*, 1998), so `AAAA` would receive the same p(w) as `CGGA`, although the latter can be found less often in the genome. Therefore it is more realistic to derive p(w) by counting instead of calculating: We use a set of reference sequences (for example all upstream regions, see section 2.2.1 on page 44 for details) and count how often we find a certain oligonucleotide. Division by the number of possible positions within the reference sequences gives an alternative, more realistic estimate for the probability that a certain word is occuring.

Since counting on large reference sequences is time-consuming and takes a lot of memory as well[10], some authors prefer to train a Hidden Markov Model on the upstream sequences of the genome first and then use this model to look up the probability of a word. Originally, this used to be a second-order HMM (Pesole *et al.*, 1992), as in WordUp, but it seems that a third-order model improves the results (Thijs *et al.*, 2001). The latter approach is taken by Improbizer, Meme (optional), MotifSampler, YMF and ITB.

[10] As long as suffix trees do not come to the rescue. But suffix trees are only used in motif discovery since a couple of years ago.

2.2.4. The Probability of several words and their significance

Given $p(w)$, the probability to find exactly x occurrences for word w in a Bernoulli-generated sequence of length n is

$$P(X = x, w) = \binom{n}{x} p(w)^x (1 - p(w))^{n-x}$$

Alternatively, we can use the Poisson-approximation since in our case np is rather small. It has the advantage that it is easier to calculate (used in Wordup). To make the equation easier to read, we define $\lambda := np(w)$:

$$P(X = x, w) = \frac{\lambda^x e^\lambda}{\lambda!}$$

Then the probability to find x or more occurrences of a word of length l on a sequence on length n:

$$P(X \geq x, w) = \sum_{i=x}^{n-l} P(X = i, w)$$

It gives the probability to find x or more occurrences just by chance. We could call it a P value for the motif.[11] It is used in this form by Spexs, for example. The lower this P Value, the more over-represented a word is. But we still do not know what P Value is high enough to give us a meaningful motif. Spexs uses many random sequences to find a reasonable threshold.

However, the P value then still depends on the length of the sequences n and the length of the word. Multiplying by the number of possible words of length l, $|W_l|$, and taking the logarithm of the result, we can calculate a measure called "significance"[12] :

$$sig(w) = \log(P(X \geq x) \cdot |W_l|)$$

It can be used to compare words of different lengths. In addition, $sig(w)$ has the nice property to increase for over-represented words ("high is good"). This is the score reported by Dyad/Oligoanalysis.

[11] Actually, the formula is usually not correct, as most patterns can not occur at each and every position (only those that consist of one repeated letter can). So the number n of positions where the pattern can be found is often corrected to exclude some positions, include the second strand and correct for palindromic patterns that will always be found on both strands (oligo/dyad-analysis).

[12] (Helden *et al.*, 1998) claims that it does not depend on neither l nor n. The author does not see how this follows from the formula.

2.2.5. The Probability as the number of possible positions

DWE (Sumazin *et al.*, 2005) assumes that a word can be potentially located on N_f positions in the foreground and on N_b positions in the background. If it was observed n_f times in the foreground and n_b times in the background, the probability to find this configuration just by chance is

$$P = \frac{\binom{N_f}{n_f}\binom{N_b}{n_b}}{\binom{N_f+N_b}{n_f+n_b}}$$

2.2.6. The Z-score of Words and its Probability

Instead of deriving a P value from the probability directly, an alternative is to modify the probability and scale it to a number which follows a known distribution in random experiments. We then can look up the probability of a certain outcome in tables or calculate it.

Given a value $p(w)$, we can calculate the expected number of occurrences in a sequence of length n, $E(w) = np(w)$, and the variance of it, $Var(w) = np(w)(1 - p(w))$, according to basic binomial laws. With these two values we normalize the number of observed occurrences $occ(w)$:

$$Z(w) = \frac{occ(w) - E(w)}{Var(w)}$$

Example: Let's assume we have equally probable letters with $p(A) = p(C) = p(T) = p(G) = 0.25$ and we look at the word `AAAA`. Its probability is $p(AAAA) = p(A)^4 = 0.25^4 \approx 0.004$. If our sequences are altogether 10000 bp long and we found the word 50 times in them, we expect this word to occur $10000 * 0.004 \approx 40$ times just by chance, with a variance of $10000 * 0.004 * 0.996 \approx 39.84$. Its Z-value is $\frac{50-40}{39.84} \approx 0.25$.

To make the assumptions more realistic, we have to account here for the two-strand, overlapping-motifs case, which corrects $Var(w)$ (introduced by (Pevzner *et al.*, 1989), extended by (Kielbasa *et al.*, 2001)) and is used in this form by ITB as a score.

If we did a random experiment, the Z-score follows the Z-distribution, so we can calculate the number of words that exceed a given Z-score if the sequences were generated by random:

$$N(Z) = (1 - P(Z)) \cdot |W_l|$$

where $|W_l|$ is the number of possible words of length l. Although this does not seem to be reported by the algorithms, it would be another, intuitive measure of how unusual a word with a certain Z-value is.

It has been suggested that Z-scores do not reflect well the situation in DNA, as the distribution of words seems to approximate very badly the normal distribution (Marino-Ramirez *et al.*, 2004).

2.2.7. χ^2-Values and their probability

χ^2 is very similar to the Z-score distribution, but does not require calculating the variance. Given the observed and the expected number of occurrences of a word w, the χ^2-Value is defined as follows:

$$\chi^2(w) = \frac{(occ(w) - E(w))^2}{E(w)}$$

The higher χ^2, the more overrepresented a word is. As for a random experiment, it follows the χ^2-distribution from which one can derive the expected number of words that exceed a given χ^2 value in a random experiment. The formula is the same as for the Z-score, and is used by Wordup and its HMM-building extension Yebis. (Hertz & Stormo, 1999) indicate that for n-long sequences $\chi^2(w)$ approaches $2n \cdot IC(w)$, if n is high and IC small.

2.2.8. Deriving a P value from an Information Content

Consensus focuses on the information content of a motif and therefore its null model requires an estimation of the probability that a given IC can be obtained by randomly aligning random sequences. One possible starting point of this calculation (based on (Staden, 1989)) is the probability to obtain a certain information content for a *single column* of the motif's weight matrix. This probability distribution can simply be tabulated by trying all values that can reasonably occur in the results, e.g. we could try all different weights possible for all of {A,C,T,G}, calculate their IC up to a certain precision and then count how probable every IC was. The obtained probabilities for each position are then summed over the whole motif length.

This leads to the value P_{mat}, the probability to obtain a certain IC for a matrix, given a certain method to construct the alignment. It is multiplied by the number of possible alignments for the current method, which is a user specified parameter (e.g. the number of motif instances per sequences are either $= 1$, ≥ 0 or≥ 1). In the simplest case, where one sequence can contribute exactly one instance, the

number of possible alignments A for s sequences of length n and motifs of width w is $A = (n - w + 1)^s$. According to the authors, the result $P = P_{mat} \cdot A$ is an approximation of the P value for the null-hypothesis.

2.2.9. Positional bias of words relative to TSS

Algorithms on yeast sequences often try to determine a certain bias of motifs to occur at certain positions relative to the start of transcription. In ITB, this position is compared to motif hits on thousands of randomly HMM-generated sequences and a P Value for the current motif's position is calculated. AlignACE's postprocessing program calculates a P value against a binomial null model (see page 60). The average distance to the start of translation in 50bp-steps is therefore reported. Its P value is calculated as the Bernoulli-probability to observe a certain number m or more sites in a 50bp window given an i.i.d. random experiment. PositionAnalysis (van Helden *et al.*, 2000*b*) reports a χ^2-score on the number of hits within a 10bp region and thus estimates how valid the assumption is that the motif is really located at a certain position.

2.2.10. Group specificity

Due to the combinatorial nature of binding sites in metazoans[13], we assume that a single motif will not be found only in coregulated genes but in many other ones as well. However, for yeast, a given motif is often considered specific for a well defined group of genes. This leads to the *representational score* (RS) from (Anderson & Parker, 2000), which is the motif's frequency in the group of coregulated genes divided by its frequency in the other genes. A P value for RS is calculated by generating many random gene-groups and counting how often a similar RS-value was obtained. A score from the probabilistic algorithm AlignACE which is calculated in a prost-processing phase, resembles this a lot (see page 60). There, group specificity is the probability of observing a certain intersection of x or more elements between result and training set in an i.i.d. random experiment.

2.2.11. Contained Motifs

Weeder in the original version reports standard scores like IC or Z-scores. But a new version (Pavesi *et al.*, 2004*b*) lists "interesting motifs" in addition. This is a rather

[13] Metazoans = Organisms that consist usually of more than one cell, like animals or plants, but not yeast

simple heuristic and is made possible since Weeder is fast enough to try different motif lengths. They are all returned to the user. If on this list short motifs are included within other, longer ones, Weeder returns the longer ones as "especially interesting". They are supposed to contain the smaller motifs as conserved cores, "a feature often encountered in real instances" (Pavesi *et al.*, 2004*b*). Apart from Neuwald et al.'s motif sampler (see below), which seems to have included some notion of core within its probabilistic model, this is the first clear commitment in the motif discovery community to the "conserved-block" property of real binding sites, exploited by motif scanners for years.

2.2.12. Fitness

This is a very special score, only used by EC. Actually, it is the weighted sum of two scores: similarity and complexity. Roughly, similarity is calculated as the sum of the differences of the motif's weight matrix from the consensus string, substracted from 1, and summed over all positions of the motif. It reflects how well the motif corresponds to the consensus, how well conserved the motif is. Compositional complexity is calculated as follows, where l is one of the letters $\{A, C, T, G\}$ and n_l is the number of letters in the string of length w:

$$Complexity = \frac{1}{w \cdot \log_4(\frac{w!}{\prod n_l!})}$$

As long stretches of the same letter, like `AAAAAT`, can create only few different words, they receive a high complexity. Obviously, this does not make much sense[14]. Altogether, fitness was supposed (without the mistake in the last formula) to be high if the motif conforms well to its consensus and its instances consist of many different letters, a rather unusual and crude filtering scheme. It would have been nice to see how real binding sites from Transfac score on these formulas.

2.2.13. Threshold Number of Misclassifications (TNoM)

DMotif considers motif discovery a discrimination problem. It first filters out patterns that exceed a certain score, those that occur *too* often, a very uncommon step that nevertheless could make a lot of sense to biologists (who wants to see TATAAT

[14] There is an error in the equation. The log should have been in the numerator. In addition, there was some normalizing done on the score, not mentioned in the paper. The authors acknowledged the mistake, after the author illustrated them with an example.

ranked number one all the time?). It then plots positive and negative motif occurrences in 2D-space and tries to fit a classifier to separate them. The P value is then calculated as the probability that this classifier is better than on randomly generated sequences.

2.2.14. Distribution over the genome

YMF (Sinha & Tompa, 2002) can postprocess its results with FindExplanators (Blanchette & Sinha, 2001) to eliminate those motifs whose instances were found very close together and keep those that are distributed equally over a certain subset of genes. This is the exact opposite of what is done when working with the Drosophila genome (Berman *et al.*, 2002), so there seems to be a significant difference in binding site distribution between the species. Another explanation could be that for *D. melanogaster*, work focuses on early developmental enhancers that might have a less specific promotor organization than other genes due to their special place in body development.

2.3. Probabilistic approaches: Expectation Maximization and Gibbs Samplers

As mentioned earlier, most probabilistic algorithms postulate a certain statistical model and then try to iteratively improve the parameters until they fulfill some stop-criterion[15]. A rather inconvenient result of the methods' iterative nature is the random starting point: Every run can lead to different results. Therefore, due to the lack of thresholds, most programs can return an endless number of motifs of which the user has to find out the top candidates.

2.3.1. MEME

Introduction and idea: *Multiple EM for Motif Elicitation* was developed by Timothey Bailey in his PhD-Thesis (Bailey, 1995). It is an Expectation Maximization (EM) approach which was originally written for proteins. EM is a general method to find maximum likelihood estimates (MLE) in data that depend on unobserved variables (see (Welling, nown) (Georgi, 2002) for good tutorials). The Baum-Welch algorithm is one of the most well-known EM-approaches: It calculates the MLE transition prob-

[15] To our current knowledge. The only exception is Meme, which reports E-values but optimizes on log-likelihood, see for (Moses *et al.*, 2004) for details.

abilities of a HMM from a sequence. EM iterates over two steps, updating the model θ until convergence:

1. Expectation-step: Given a model θ^t (the initial model is random), determine a formula for the expected value $E(\theta^t)$ of the model
2. Maximization-step: Use this formula to find the θ^{t+1} which maximizes E, e.g. by derivation.

Assumptions and Data Structures: In our application, the instances' positions – the alignment θ – are the unknown variables. MEME assumes that there is One Occurrence Per Sequence (OOPS-model). The more involved Zero Or more Occurrences Per Sequence (ZOOPS) and TCM (Two Component Mixture, instances can occur anywhere) models have more variables in θ, but the basic procedure remains identical. The number of instances per sequence, for example, can also have a random starting point and then be explored by EM. The occurrence-model can be selected by a parameter, just like the background Markov Model (which is a rather new addition, not mentioned in the original article) and (optionally) the width. The optimal width can also be sampled by MEME, by trying different widths, eventually shortening them by doing an additional EM-round without some of the flanking bases.

During the algorithm, MEME keeps a list of the weight matrices derived from the current putative instances (θ) and does calculations on them.

Search Algorithm:

1. Expectation-step: MEME roughly collects candidate sites for a given (or random) weight matrix derived from the model (by scanning the sequences) ...
2. Maximization-step: ... and then updates the model with those new sites in the sequences that lead to the highest expected value.

This is a kind of gradient decent search from a random starting point that will always reach a local maximum (like all EM methods) but might fail to find the global maximum. Therefore, MEME has to be run many different times with random starting points and the set of results is then sorted by E-Value. Actually, the sketch of the algorithm here is only valid for the OOPS-model. In ZOOPS and TCM-models, more data, like the number of sites per sequences, have to be guessed and improved as well.

Scoring: The E-value gives the number of times one expects a motif of this log-likelihood to be found in random sequences. The likelihood is the ratio of the probability to find this particular instance of the motif model divided by the probability

to find it in the background (random i.i.d. sequences with the same A/T-content as the analyzed sequences *or* sequences generated by a given Hidden Markov Model, which is a parameter). The log-likelihood can be directly obtained during the algorithm, as it is part of calculating the expected value.

Experiments: As Meme targeted protein motifs, there were no experiments on DNA-data. However, most other programs compared their results with Meme, usually with the result that their own implementation scored better (e.g. (Sinha & Tompa, 2003) (Workman & Stormo, 2000) (Pevzner & Sze, 2000)). In the motif discovery assessment (see page 77), however, MEME scored better than many of these methods. This might be due to difficult parameter setting and to the fact that MEME was originally aimed at motifs in protein sequences. Therefore, its results still seem to be plagued by rather too long motifs and too degenerate ones, like in the assessment, where MEME's predictions were "evaluated by eye" by the authors (see the "MEME – Results – Comments " section of the assessment website (Tompa *et al.*, 2005)).

One companion program to MEME, MAST (Bailey & Gribskov, 1998), does a fast search on a sequence database for the motifs discovered by MEME. Another companion, MetaMEME (Grundy *et al.*, 1997), tries to train a HMM with the instances found of MEME's output and as such can learn a group of protein motifs that occur close together. It is likely that due to the many false positives when searching for binding sites, this approach is not transferable to DNA motifs and as such we have not seen any applications to DNA yet.

2.3.2. Other EM-methods

Logos: This approach looks completely different as it is built around a different motif model, called hidden Markov-Dirichlet Multinomial (HMDM). Roughly, it assumes that a collection of weight matrices serves as prototypes for all final weight matrices. Every prototype contributes to the final matrix with a value sampled from Dirichlet distribution. So there has to be a collection of start matrices, given together with weight distributions how much they influence the result. All of these distributions are then iteratively sampled to maximize the information content of the final matrix. When a motif model has been determined, the sequence is scanned for it. The matching positions are then run through a Hidden Markov Model to find co-occurrences or clusters.

This model is meant to account for motifs where conserved positions in the matrices are preferably located next to each other and binding sites occur in clusters, which seems reasonable from a biological point of view. The MotifPrototyper (Xing & Karp, 2004) was later used to train the model's parameters and to scan for known motifs and discover unknown ones. When applied to yeast, the program is reported to outperform both MEME and AlignACE. It was also applied to Drosophila genes but no comparison was done there.

2.3.3. Original Gibbs Site Sampler

Introduction and Idea: The original "Gibbs Sampler for Motif Discovery" (Lawrence *et al.*, 1993) was published in Science in 1993 and is called just "Gibbs" in the following. Since one of its original assumptions was that there exists at least one instance of a motif in every sequence, the method is sometimes called the "site sampler". Gibbs is a Markov-Chain Monte Carlo (MCMC) approach: "Markov-Chain", since the results from every step depend only on the results of the preceding one (like in EM). "Monte-Carlo", as the way to select the next step is not deterministic but rather based on sampling, i.e. random-numbers. The only difference to Meme seems step number two, where EM selects the single instance that maximizes the expected value, whereas in Gibbs every instance has a certain probability to become selected. The statistical background of MCMC-methods is explained in the book by Jun S. Liu (Liu, 2001) and that of Gibbs in particular in a companion article (Liu, 1995).

Assumptions and Data Structures: Input is a set of n sequences, each sequence has to contain at least *one instance* of the motif. The length l of the motif is given.

As for variables, two matrices of length l are used: $MotifMatrix$ to store the current putative motif that the algorithm has collected so far, $BackMatrix$ to store the current background composition. $Alignment[1..n]$ is an array to store the current offsets (or: positions) of the putative instances in the sequences. For example, if we have $n = 5$ sequences and $Alignment[1..5]$ is [1,1,1,1,1] then the motif is supposed to start in every sequence at the first letter (up to the lth letter). It is obvious that given the array $Alignment$ we can calculate the $MotifMatrix$ (this was explained in detail on page 14)

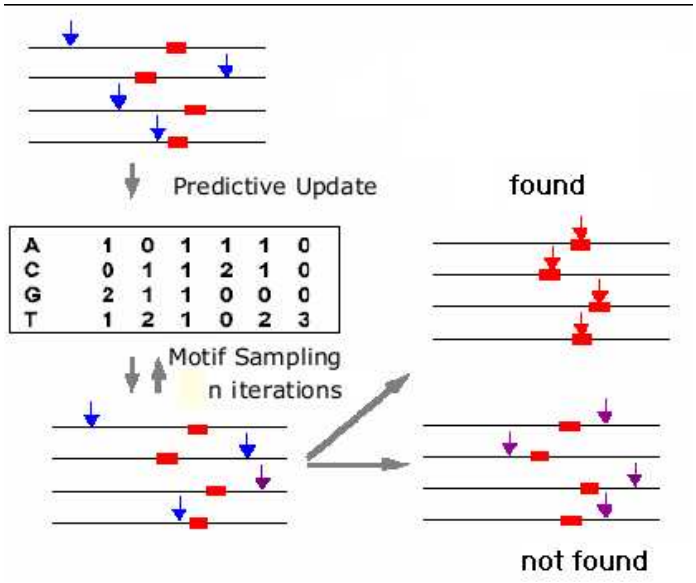


Figure 2.5.: The two basic steps of Gibbs Sampling Algorithms

Search Algorithm: At initialization, $Alignment$ is filled with random values that have to be valid offsets in the sequences. Then the following two steps are repeated:

1. Predictive Update Step: $MotifMatrix$ is calculated from the strings indicated by $Alignment$, except one. This is sequence z, completely left out of the subsequent calculations. We could imagine this as an algorithm that — assuming that one instance of its data is wrong — is trying to refine only this parameter. As $Alignment$ points to all instances of the current motif in all sequences, all *other* strings of length l in the sequences are now considered to be background, non-motif strings. Therefore, $BackMatrix$ is calculated from these. In this way, two matrices, one for the current motif and one for the current background, have been calculated from all sequences except z. A very simplified version of this is shown on figure 2.5, taken from Denis Thieffry's lecture slides.
2. Motif Sampling Stenp: These matrices can be used to measure the probability that a given string corresponds to $MotifMatrix$ and not to $BackMatrix$. To this end, all substrings of length l in sequence z are scored (see page 23). The ratio $A_x = \frac{p_m}{p_b}$ of motif probability to background probability for every string on sequence z is calculated. This number could be called a weight, because

the algorithm does *not* always select the string with the highest ratio. Instead, the weight is used to sample randomly (see below) one of the strings on z as the new putative instance of the motif. Its offset is stored in $Alignment[z]$. Randomized (Gibbs) sampling means that a particular string is taken with the probability

$$p_x = \frac{A_x}{\sum A_x}$$

As long as there is no hit on a real motif, the algorithm will sample just randomly through the sequences but as soon as one true motif has been selected by chance, more of the similar ones will be chosen in the other sequences, leading quickly to convergence. Of course, the process is not deterministic. It has to be repeated as often as necessary to give somewhat reproducible results. A "run" is a set of iterations that result in one motif.

Scoring: The various motifs found in several runs are compared and the best ones are kept. Since during the search the weights have already been calculated, they can be reused for motif scoring. We therefore search for the run that has the highest product of all the A_x. Or, equivalently, the maximal sum of the logarithms of A_x, given by

$$F = \sum_{i=1}^{l} \sum_{j=1}^{4} c_{i,j} \log \frac{q_{i,j}}{p_i}$$

where $c_{i,j}$ denotes the count of nucleic acid i at position j in the motif instances (strings) that are represented by $Alignment$ and $q_{i,j}$ is the probability that nucleic acid i at position j corresponds to $MotifMatrix$; p_j is the probability that position j corresponds to $BackgroundMatrix$. In short, the sum of the weighted (log'ed) likelihoods of all nucleic acids, summed over the whole length of the motif. Actually, this score is almost identical to relative information content, the only difference lies in $c_{i,j}$, that is an integer-count here and not a probability.

Details: If the length of the motif l is unknown to the user and not specified, the algorithm can search for the optimal one. In this case, several runs will be started, trying a range of possible lengths. However, to compare them and find the optimal one, score F has to be normalized with respect to different lengths of the strings. Otherwise, the score would always favor longer strings. Roughly, this is done by substracting the information needed to determine the location of the motif

in each sequence divided by the number of free parameters (in the case of DNA, $4l$). Lawrence et al. call this *information per parameter*.

The algorithm can easily be mislead by a shifted version of a pattern: If in an early iteration, positions two letters away from the start of the real motif's instances were selected, then this will always lead to the selection of similar, shifted instances in all the other sequences. Every couple of iterations, the current set of positions is therefore moved slightly to the left and the right to see if this improves the score F.

Experiments: The original article didn't try an application to DNA-binding sites, as the algorithm only targeted proteins. The problem of binding site identification didn't yet attract much attention in 1993, due to the lack of co-expression data. An intermediate improved Version was MACAW (Liu, 1994), that implemented the Gibbs Sampler for DNA and was reported to be more effective than Expectation Maximization on some human samples (CRP-genes).

2.3.4. The Motif Sampler by Neuwald et al.

Introduction and idea: Since Gibbs was not tailored to DNA sequences it did not take into account their special properties. Most notably, long stretches of arginins are ubiquitous in the genome, so receive high scores in the F-function but don't carry any biological meaning. It was therefore necessary to better adjust the site sampler's scoring model to the case of binding site discovery.

As the assumption "one occurrence per sequence" (OOPS) can rarely be guaranteed for data from real biological experiments, Neuwald et al. (Neuwald *et al.*, 1995) adapted the original Gibbs to remove this limitation. The new algorithm is the basis for all other Gibbs samplers on DNA, searches multiple motifs in parallel and highly improves the scoring scheme based on statistical foundations laid in (Liu, 1995), by distinguishing between highly-conserved and less-conserved positions in the motif.

Assumptions and Datastructures: The input is one long sequence that is assumed to contain k motifs. If the user supplies several sequences, they are simply concatenated. The result is expected to contain e [16] instances. No single $MotifMatrix$ is used, but a whole set of them, $MotifMatrix[0..k]$, as multiple motifs are searched. So $Alignments[0..k, 1..max]$ becomes a two-dimensional array. The variable max is a theoretical upper limit on the number of instances possible. There is no

[16] Actually, e is sampled from a Dirichlet distribution. Only some "crude guess" (Liu, 1995) has to be made on its value. It is also sometimes called *prior*.

$BackMatrix$, but $MotifMatrix[0]$ serves instead as the background distribution, for reasons explained below.

As a completely new concept, every $MotifMatrix$ has a $MatrixMask$. This is a bitmask that indicates which positions of the matrix are relevant when calculating the weights, i.e. the similarity between a string and the matrix. The concept reflects experiences by the authors, that some positions of a motif are biologically almost irrelevant. There is no width-optimization step, so the length l of the motifs must also be given.

Algorithm: The basic algorithm is very similar to Gibbs with the exception that it operates on one sequence instead of a set of them and multiple matrices instead of one. Therefore, at initialization, it randomly choses e l-long instances in the sequence for all k MotifMatrices and stores them in Alignments[1..k,1..e].

1. Predictive Update Step: Just as in Gibbs, the MotifMatrices are calculated from the positions of the instances. Instead of one matrix, here, e instances are mapped to one out of k motifs.
2. Motif Sampling Step: One substring s (which is not already an instance from $Alignments$) of the sequence is chosen. Now the $MotifMatrices$ are scored[17] and one of the matrices is weighted-randomly taken. This can also be $MotifMatrix[0]$, the background distribution. To save this result, substring s is counted as an instance of the newly sampled matrix and added to $Alignments$ (and, if it had been an instance of another matrix before, removed from the old matrix' alignment).
3. Column Sampling Step: Positions in the motif that are rich in information are more important for a motif than the others, therefore the algorithm uses only a subset of the columns to calculate similarity; these are called "on" in the following. But this subset has to be adapted to the changed matrix, so one of the on-columns is switched "off" at random. These columns are then sampled according to a weight that is developed in the article by Liu et al. The formula is rather complicated, but it has the property that positions that are close to information rich positions receive more weight.[18]
4. Near-optimum Sampling Step: After a fixed number of iterations of the first three steps, the best motif (see *Scoring*) is kept. Sampling the sequence for this

[17] Actually, only the columns that are switched on in $MatrixMask$, but that will be explained in step 3.

[18] If in this step the first letter is switched off, the whole motif is shifted one letter towards the right, so no additional special shifting step is necessary.

matrix continues without the column sampling step and with better values for e now. This makes sure that the motif now does not change very much, but renders slight improvements possible and new instances can be found that are closely related to the final motif but were not discovered before.

Scoring: During sampling: Information Content was replaced by the MAP-score (maximum a priori log likelihood). It measures the degree of overrepresentation of the motif in the sequences instead of the motif's similarity to its instances. The coauthors of the original Gibbs article had developed the formula (Liu, 1995)[19], of which in the original Neuwald-paper only a rough approximation is given as $MAP = NlogR$, where N is the number of instances of a motif and R is the degree of overrepresentation of a motif. R is not trivial to develop nor to understand but the authors claim that it increases if one of the following is true: many instances, better conserved motifs, less input sequences, tightly packed conserved positions and a high usage of rare nucleic acids.

With the final motifs: Based on a set of background sequences that are the shuffled input sequences, the significance of the motifs is evaluated. The shuffled sequences are therefore appended to the real sequences and the sampler is started again. Unrealistic motifs are now as likely to come from the input as from the control sequence part. A Wilcoxon-test is used to calculate a measure for motif significance from the distribution of instances.

The Mann-Whitney-Wilcoxon-Test compares two ranked samples A and B and tests against the null-hypothesis that the median of the two distributions is 0. The ranking is done on the log-odds-score that is a direct result of the Gibbs sampler. Then, taking every motif m from A, we count the number of motifs in B that score less. The sum over all possible m will give U, a score whose distribution is known to approximate the normal distribution and for which a P value can be looked up.

Experiments: Again, only protein data is given.

2.3.5. AlignACE

Introduction and idea: The improvements of the motif sampler were taken over by AlignACE, but extended to better suit real biological examples. Data on yeast was used to guide the process. AlignACE is also accompanied by tools that simplify the task of judging the biological importance of motifs and group similar motifs into

[19] Jun S. Liu is statistician. Therefore, his papers assumes the reader to have a high level in statistics.

clusters. The whole set of programs was used for several well-known studies on yeast (Roth *et al.*, 1998), (Tavazoie *et al.*, 1999) (Harbison *et al.*, 2004). Here we describe the latest version from (Hughes *et al.*, 2000).

Algorithm: Rather small changes to the search process: The priors for the nucleic acids' probabilities were fixed to the values for the yeast genome. Both strands are searched at both steps of the algorithm for the motif. As the assumption of one motif per sequence is usually not valid in promoters, the algorithm is adapted to multiple motifs: When an occurrence is found, its strong positions are marked and cannot be used in the following sampling steps anymore, which should eliminate the problem of finding the same motif several times in subsequent iterations.

Scoring: Apart from the usual MAP-scores, the probability which the Gibbs Samplers maximize, AlignACE tries to rate the motifs with two procedures that are more biologically motivated. ScanACE is a separate program to measure the importance of motifs in the context of the whole genome, which makes a lot of sense from a biological point of view. As input, it gets a list of motifs from AlignACE and all the upstream sequences of a genome. From this a group specificity is calculated, i.e. a measure if the motif is only present in the set of coregulated sequences from which the motif was learned (the original input to AlignACE, the training set) and not in the rest of the genome (the result set). In addition, a positional bia s is reported, already introduced on page 50

Experiments: Compared to other publications, the authors performed a very comprehensive analysis of their tools on real data, 6226 ORFs in yeast. Corregulated groups were extracted from various sources, from which AlignACE found ca. 3000 motifs. They were filtered and grouped into 25 clusters. ScanACE was run to find these in the genome and to sort them by group specificity. Of the 25 highest scoring ones, 16 were already described in literature. Whereas positional bias played a strong role for some, many motifs did not demonstrate significant positioning relative to the start of translation. Negative tests on random sequences were done as well as positive tests on data where the binding sites were already known, where of 29 known motifs, 21 were found.

2.3.6. ANN-Spec

Introduction and Idea: One of the oldest approaches in binding site scanning, even older than the weight matrix, is perceptron learning[20] (Stormo *et al.*, 1982). It is not suprising to see a related algorithm appearing in motif discovery 18 years later, published by a student of Mr. Stormo. But although ANN-Spec (Workman & Stormo, 2000) uses an **A**rtificial **N**eural **N**etwork to find the **spec**ificity of binding sites, it seems to be more similar to the original Gibbs Site Sampler: A perceptron is used to classify substrings as binding-site or non-binding-site, so the weights of the perceptron correspond to the weights of the matrix in Gibbs. However, the *update* of the weights is done with a gradient, the common method in perceptron learning. According to the authors, ANN-Specs' performance is slightly superior to both Gibbs and Meme.

Assumptions and Datastructures: Motif-containing sequences and background sequences are given. One perceptron is used for the scoring and the classification into the classes "binding site" or "background". The number of motifs per sequence k can be set a priori or is estimated based on scanning the background sequences in every iteration.

Algorithm: The weights of the perceptron are initialized to represent a randomly sampled site in the sequences. Then the following steps are repeated until a fixed number of iterations is reached:

1. Motif Sampling Step: According to the authors, the weighted sum of the perceptrons' inputs for a substring j, h_j, could be read as a binding energy or probability for fixation or collision of two molecules, namely the transcription factor and the particular sequence j. Therefore, the probability h_j is not used directly as in Gibbs. Instead, the Maxwell-Boltzmann function $\exp^{h_j}$ is applied. The result is a weight for every substring from which k sites are sampled.
2. Weight Update Step: The objective is the log-likelihood of the selected sites. Then a weight change is applied:

$$\Delta\Omega = \eta(\frac{\Delta U^*}{\Delta\Omega}) - \lambda\Omega$$

[20] A Perceptron is usually a class of very simple Neural Network, with only feed-forward connections. In Ann-Spec, only one single artificial neuron is used per letter, so this perceptron cannot be really called a Neural Network.

Ω is the weight-vector, η the learning rate, U^* the objective and λ the decay rate. So the change is the log-likelihood from the sampled sites (the new weight) divided by the old weight, corrected by learning rate and decay factor. The basic formula is a standard in perceptron learning and makes some difference between the current algorithm and the Gibbs site sampler: Here, every weight is adjusted into the direction of the newly selected sites. It does not directly reflect these sites.

Scoring: Log-likelihood is the objective and therefore the parameter that corresponds to the score. In this case, it is the Boltzmann probability of h_j divided by the number of similar sites in the genome (the background sequences), summed over all possible sites in a sequence. Its logarithm, summed over all sequences, gives a score that increases if the motifs are either well conserved in the positive sequences or they occur more often in the positive set than in the background.

Experiments: Random sequences were generated (500bp long) and mutated motifs implanted into them (10-80 sequences). Gibbs performed slightly better than Ann-Spec on these, but Meme and Consensus fall behind. On sequences where a non-random background was used and provided to the algorithms, giving Meme and ANN-Spec the possibility to exploit its composition, ANN-Spec scores much better than Meme and Gibbs, whole results do not differ much from each other. Another test with the same results on real yeast promotors is mentioned, but the respective paper was not accepted by the journal and is not available from the authors anymore, its work not persued any further (personal communication, C.T. Workman)

2.3.7. MotifSampler by Thijs et al.

Introduction and idea: Based on the observation that gene prediction software uses hidden Markov process models to distinguish genes from intergenic sequences, the authors of the MotifSampler (without a space) (Thijs *et al.*, 2001) replaced the background matrix, that was used in the other samplers, by a Hidden Markov Model. Applied to sequences from Arabidopsis, the authors claim that it improved the sampler's performance (Thijs *et al.*, 2001) (Lescot, 2002) (Marchal *et al.*, 2003), although this might seem surprising, given that separating binding sites from noise within promotors is a very different problem than distinguishing whole promotors from genes. It has to be noted, though, that Thijs et al. were not the first ones that added a markov model to a Gibbs Sampler (see Improbizer, on page 67). They also

wrote a motif scanner that is built around the same idea, calculating scores of hits via an HMM. (Coessens *et al.*, 2003)

Assumptions and Datastructures: Though published in 2001, the algorithm is an extension of the original Gibbs site sampler from 1993, whereas the name would imply rather an improvement to the motif sampler by Neuwald et al. Like the latter, Thijs et al. remove the inconvenient constraint "one motif per sequence" , so zero to C_{max} copies in a sequence can be present. Therefore, the list of positions has multiple entries for every sequence again. However, the number of motifs is sampled from a discrete distribution Γ that is updated in every step (see details) whereas Neuwald et al. use a Dirichlet distribution.

The distribution Γ can be saved as an array, where for every number of copies a probability is stored. The HMM used to represent the background is built from selected intergenic sequences and a couple of them are supplied by the authors for various model organisms. Therefore, the background probability can be calculated for every substring, and saved before the real algorithm starts, as the HMM does never change.

Algorithm: After calculating the background probabilities, intializing the alignments with random numbers and calculating the distribution of copies Γ from it, the following steps are repeated until convergence or until a maximum number of iterations are reached:

1. Copies Sampling Step: For every sequence, the number of expected copies C_k is sampled from Γ.
2. Predictive Update Step: As in original Gibbs, remove all entries of one sequence z from $Positions$ and reduce the putative sites in all sequences so that there are only C copies left. Calculate the motif matrix from $Positions$.
3. Motif Sampling Step: Calculate motif probability for every substring in z. Score each substring with motif- divided by background-probability. Sample C instances from this distribution and add them to $Alignments$. Update Γ.

Scoring: Three different measures are output: Information Content relative to background as in Gibbs (called Kullback-Leiber-distance[21] here), Information Content of the pure motif (called Consensus Score by the authors) and log-likelihood.

[21] Which should be called Kullback-Leibler-Distance as in equation 1.2, but this typo is recurrent, not only in many articles by Thijs et al.

This is the sum of the following "log'ed" probabilities: Probability that sequence is generated by background, probability that maximum C copies of motifs are observed and the probability that the motif corresponds to the sites, summed over all possible numbers of copies. Log-likelihood is the score that is optimized during sampling.

Details on the calculation of Γ: The probability to observe C copies of a motif in a sequence given the background is calculated according to Bayes' theorem. This is the probability to find C motifs in the sequence multiplied by a constant factor [22] divided by the probability to find a reasonable number of motifs in a sequence (all probabilities are relative to the background). The whole expression is calculated for every reasonable number of copies between 1 and C_{max} and for the two strands and stored in Γ.

Experiments: Extensive evaluation of its parameters has been performed with the MotifSampler, though no documented comparisons with other algorithms were done in the paper. Various model organisms were studied and multiple articles treat the results:

- *A. thaliana*, G-box: A set of 33 500bp-upstream sequences was known to contain the 6bp-long G-Box motif. Six motifs of length 8 were searched in 20 runs of which the algorithm found the motif in 15-18 runs. The higher C_{max} is set, the lower the number of successful runs was. Then random upstream sequences from *A. thaliana* that were known not to contain the motif were added to the set and the sampler re-run 10 times. Starting at 30 negative sequences, performance deteriorates but never falls below 5 successful runs. The version without the HMM as background model scores much worse, as little as one third to one tenth of the runs are successful. Only one strand was analyzed for all the searches.
- *S. cerevisae*, MET: Using data from (Helden *et al.*, 1998) eleven 800bp-upstream sequences on both strands were searched for a known MET-motif. The motif had the length 9bp, but best results were obtained using a shorter width of 8. In one third of the runs the real motif was identified. Performance decreased when length or C_{max} was increased.

[22] This is the probability to find C motifs in the background, but for simplicity it is replaced by a constant prior: The expected probability of finding a motif in a sequence input by the user, according to the authors 0.2 is a good value to start with.

- *Bacteria*, FNR: Data from various bacteria was assembled for a testset of 10 sequences that are regulated by the transcription factor FNR. MotifSampler is reported to have found the 14bp-long sequence as expected, but no probabilities nor parameters are given.

2.3.8. Other Gibbs Samplers

Bioprospector Bioprospector (Liu *et al.*, 2001) differs from the original Gibbs sampler in four aspects:

1. Markov background model: Segments are scored relative to a third-order Markov model.
2. Threshold sampling: When selecting segments, in the sampling step, two thresholds are used: All substrings with a score exceeding the high threshold are immediately selected, those with a score that fall below the low threshold are completely left out of the sampling. The sampling is only done on substrings that score between both thresholds. The new notion of a "too good" correspondance to a model is interesing; only DMotif incorporates similar filtering.
3. Specific motif models: The user can specifiy a two-blocks/fixed- spacer or a palindromic model. The sampler then uses two different matrices with a spacer or one matrix that is applied two times, reverse-transcribed and normal.[23]
4. Corrected motif scoring scheme: The Information Content of a motif can be good even if there are only very few instances. But a motif with 150 instances is biologically more meaningful although its IC is usually lower. Therefore, the information content of a motif is multiplied by the number of instances.

The algorithm was tested on yeast and bacteria data, but not compared to any other one.

MDScan Actually this algorithm from (Liu *et al.*, 2002) is a slighly modified version of the original Motif Sampler that is adapted to all experiments where information is available that a couple of sequences has a preference to contain binding sites. In ChIP-on-Chip-data, for example, there seems to be some ranking possible derived from the strength of binding, signaled by a high intensity of the spot on the array (see section 1.2.1). The highlighted sequences are more likely to contain the motif, so MDScan uses every l-long substring of them as the starting point for a "with-

[23] This is similar to the approach of Dyad-Analysis and Co-Bind (next page)

mismatches" scan over all sequences. All of the substrings that are found to be similar to the given word are then saved and a matrix is calculated from them. This matrix is used for the first predictive update step of the Motif Sampler, as a starting point for the search. MDScan is therefore something like a combination between Sample-Driven Enumeration and Gibbs Sampling, which is especially adapted to ChIP-data. It is consequently reported to score better than AlignACE, Consensus or Bioprospector on data where the rough number of instances is known. This is – like in the original Motif Sampler – still a parameter that has to be specified (the "prior"). MDScan was applied to 12 yeast ChIP-array data-sets and many artifical sets.

Co-Bind Co-Bind (GuhaThakurta & Stormo, 2001) is tailored to a specific motif model, the dyad model of two adjacent motifs separated by a spacer. It also uses gibbs sampling but with a different predictive-update step: Similar to Ann-Spec, gradient-descent is done instead of direct calculation of the new matrix from the newly selected instances. The scoring model explicitly calculates probabilities for two different factors being bound by one gene. Although differences are small, the authors claim that Co-bind finds significantly more motifs than Bioprospector.

Ampheta meme Jim Kent (the author of the UCSD genome browser and an active author in bioinformatics tools since many years) wrote another Gibbs site sampler in 2001 (also called Improbizer) that is using a Markov model for the background, can extend the motif length and is using background sequences to calculate a score. Unfortunately, apart from the source code and a small list of parameter descriptions on the website, no more information is available and no article was published until now. Example data on yeast can be found on the website.

SeSiMCMC "Yet another Gibbs Sampler digging for DNA-motifs" (the title of the poster) (Favorov *et al.*, 2002) is yet another gibbs sampler that searches for 0 or 1 occurrences of a motif per sequence, optimises the motif length while sampling, can improve found patterns by scanning the sequences similar to the near-optimum sampling in the motif sampler by Neuwald et al. and uses relative Information Content as a score. It was applied to bacteria. Only very short descriptions at the size of extended abstracts are available. Some took part in the assessment described in 2.7.

2.3.9. GMS-MP

Introduction and Idea The group around Jun S. Liu publishes many[24] derivatives of the Gibbs Sampler. At the time of writing, the most recent version is slightly modifying the weight matrix model. In a Generalized Weight Matrix (GWM), every letter along the motif has still a probability to occur at a position, but a given nucleotide can be correlated with another one. The authors claim that this model needs less parameters than any of the more complex ones like Bayesian networks or HMMs (see section 1.2.2 on page 19), while still being more expressive than a standard weight matrix and therefore can increase the performance of both scanning and motif discovery alike.

Assumptions and Datastructures The GWM consists of two types of matrices: One is the usual mononucleotide weight matrix, the other one are some dinucleotide matrices, which each have an attribute (x,y) that specifies the positions to which the matrix applies. The dinucleotide matrix consists of a list of all possible combinations of two nucleotides (16) and gives the probability to observe the two in this order at the positions x and y. For example, if positions $(2, 6)$ in a motif of length 6 are assumed to be correlated, then the matrix for positions 1,3,4,5 stays the same as usual, but for nucleotides 2 and 6 a seperate submatrix is created, that lists all possible combinations of letters for $(2, 6)$ (16 possibilities) and their respective probability.

The second change to the original motif sampler is that the prior is not a fixed, user-supplied parameter anymore. Lead by experiences from (Liu *et al.*, 2002) this is sampled from a Bernoulli distribution.

Algorithm The algorithm itself stays almost identical to the motif sampler by Neuwald et al. The sampling step now includes choosing a value for the prior and does not sample only the normal weight matrix but all the weights of the GWM. The score is the usual MAP-score, obtained and maximized as part of the sampling process.

Experiments To prove that the GWM outperforms a normal matrix, the authors used transcription factors for which there are many binding sites available (> 20) and scanned upstream sequences for them. Whereas they concluded that often GWM outperforms normal matrix scanning, it sometimes offers no advantages. Motif discovery with GMS-MP was done on the well-known CRP-sequence set ((Stormo &

[24] Roughly 17 out of the 57 papers, listed on his homepage, from 2000-2005 treat the topic of motif discovery, 8 different implementations can be downloaded.

Algorithm Name	Model	Search	Score	Publ.	Software
Gelfand et al.	String, Pal.	Enum	IC	(Gelfand *et al.*, 2000)	None
Phylocon	Matrix	Cons.	ALLR	(Wang & Stormo, 2003)	Src,Web
CompareProspector	Matrix	Gibbs	MAP	(Liu *et al.*, 2004*a*)	Web,(Src)
EMnEM	Matrix	EM	p	(Moses *et al.*, 2004)	Not yet
PhyMe	Matrix	EM		(Sinha, 2003)	Src

Table 2.3.: Phylogenetic footprinting algorithm implementations
Legend: Cons. = Consensus, ALLR=Average Loglikelihood Ratio, (Src)=Sourcecode available, but needs some kind of agreement signed first, Pal=Palindromic, Enum=Enumerative (Pattern Driven)

Hartzell, 1989), sequence length 100 bp) and compared to Bioprospector and the original Motif Sampler, which lead to roughly similar results, as long as the right prior was chosen for the two older Gibbs samplers.

On a second dataset (induced by the E2F-transcription factor, sequence length 300 bp) all three samplers were run again and there it turned out that since positions 1 and 2 of the embedded motif were really correlated, the GWM-sampler's motif instances came close to the original number of sites while still keeping the matrix similar to the original motif. There is no estimation, however, how significant the differences are.

2.4. Motif Discovery with additional data

2.4.1. Interspecies comparison algorithms

In section 1.2.5 we referred to the use of global multiple alignments between close species to uncover binding sites as "phylogenetic shadowing". If more distant species are used, the comparative approach is called *phylogenetic footprinting* (Boffelli *et al.*, 2003). Because of the many differences between the promotors involved, global alignments are not usable anymore. So motif discovery – similar to motif scanning before, as presented in section 1.2.5 – goes the way of inter-species comparisons to achieve the necessary performance improvement. Therefore, it won't be very suprising to see the same smorgasbord of approaches being modified and applied to many genes from a couple of different species leading to a new torrent of algorithms.

Three basic precedures have been followed until now: (1) Treat orthologous genes like co-regulated genes and run a standard algorithm, (2) align orthologs first and

then apply discovery only on conserved regions and (3) exploit evolutionary distances directly as an integrated part of the motif discovery.

Examples of the first group are not very new: (Gelfand *et al.*, 2000) applied Z-scores and an enumerative search for palindromic hexamers, (McGuire & Church, 2000) and (McGuire *et al.*, 2000) extracted upstream sequences with the methods 1-3 listed in section 1.2.6, which are rather specific to bacteria. But both just pooled the sequences together and then did a standard motif discovery, without taking phylogenetic relationships into consideration. (Kielbasa *et al.*, 2004) describe how to apply a combination of ITB, Transfac and Clover on orthologous genes but only intermediate results as graphics are given. It is still interesting since mouse genes are used and the general setup is realistic (no TSS given, co-expressed genes).

The second way and a one that makes better use of the data is to first filter out only conserved motifs and then apply the discovery on the conserved regions, the very same that was done in motif scanning before.

Kellis et al.: As yeast has a smaller genome, where a high percentage of transcription factors and binding sites are already known and where binding sites themselves have a simpler structure than in higher eukaryotes, it is among the first candidates for this type of method. One of the Best analysis so far, described in three articles, first sequenced three *Saccharomyces*-genomes (Kellis *et al.*, 2003) then wrote and applied their own motif discovery program with an extensive filtering scheme. (Kamvysselis *et al.*, 2003)

They first searched for orthologous genes with a special algorithm that uses similarities to build a synteny graph which is then pruned until genes are identified that were conserved in the other species. Their upstream and coding regions are then globally aligned. The authors point out that alignments for coding and non-coding regions differ sharply in general and used this property to filter out 10% of the open reading frames in *S. cerevisiae*, as they probably do not constitute real genes, although they are marked in databases as such. (Kellis *et al.*, 2004)

Then two different algorithms are applied. One searches for conserved trinucleotides, separated by a gap of length 0-21 bp. Then a three-level filter is applied: The motifs have to be totally conserved between species, preferentially conserved in intergenic vs. coding regions and over-represented in upstream vs downstream sequences. The results are extended in all directions but don't have to be fully conserved there, as for the extension IUPAC-wildcards are also tried. Every base that discriminates the motifs against the non-conserved motifs is taken as an extension. The produced motifs are then clustered by similarity, which leads to fewer than 200

strings (It might be just a coincidence, but this corresponds well to the number of estimated transcription factors in yeast, see Table 1.1). As a last step, motifs are searched that are located preferentially upstream of functionally related groups of genes. To the authors' surprise, many groups shared motifs. So it seems that no motif is specific for a certain function and no function entails a certain motif in yeast, due to the combinatorial nature of transcription factors. For eucaryotes, this has been noted in quite a few reviews before and is well-known from basal binding sites. (Werner, 1999)

In a follow-up study, (Chiang *et al.*, 2003) searched for hexamers that were conserved in three of the four species and kept only those that had some preferred spacing to any other of these words. Then they searched in published gene-expression data for the selected hexamer-pairs. They conclude that the motif pairs really had significant influence on expression and that spacing is therefore important. They also acknowledged that using weight matrices, more motifs could have been found but trying all 2-combinations of all weight matrix would have been computationally too expensive. It seems that the whole methodology was evident at the time, as a very similar article was published four weeks later that also sequenced three yeast species and did similar, while less automatized, aligning and filtering of conserved motifs. (Cliften *et al.*, 2003)

CompareProspector (Liu *et al.*, 2004*a*) is a variant of BioProspector (Liu *et al.*, 2001) that constructs global alignments first using Lagan (Brudno *et al.*, 2003) and then directs the search towards regions of the sequences that are better conserved, by giving conserved regions a higher weight for the sampling step.

Pritsker et al. (Pritsker *et al.*, 2004) is a very unusual kind of interspecies-filtering that does not really fit into any category: The authors first applied AlignACE to orthologous upstream groups and took the best scoring motifs. Then they searched for these motifs in two different genomes and listed for every genome those genes where the predicted motifs could be found. The better the two gene lists overlap, the higher the final score of the motif. It is assumed that a transcription factor will usually regulate the same set of genes, a fact which is called "network-level conservation" in the article. The authors speculate that 30% of the motifs that they found could be valid ones.

The following algorithms do not follow the "filter-first, search-later"-method but integrate the similarity of sequences into the motif discovery process. According to the typology of the preceding page, this is method (3) of inter-species discovery:

FootPrinter seems to lead the field in terms of age, problem specificity and "publication count" (Blanchette *et al.*, 2003, Blanchette & Tompa, 2003, Blanchette & Tompa, 2002, Blanchette *et al.*, 2000). It assumes that all sequences are not equally close to each other. The reason: If they were treated the same, then a group of similar sequences would carry too much weight for the motif finding. Therefore, the input is not only the sequences but also a phylogenetic tree that specifies how the organisms developed during evolution. FootPrinter then searches the closest sequences (close to the leaves in the tree) for all w-long substrings, goes up the tree and recursively continues. The user has to specify the evolutionary distance that should be covered by the motif and those that exceed the threshold are output. A detailed benchmark against Meme and two multiple alignment programs (ClustalW, DiAlign) has been published (Blanchette *et al.*, 2003) but only on artificial sequences. And even here, Meme scored similarly well for examples that were not specificially tailored to footprinter's assumptions.

OrthoMeme (Prakash *et al.*, 2004) is a version of Meme that, as the name implies, works on a pair of orthologous sequences. The algorithm is almost identical to its predecessor, except that here an instance of a motif has two positions, one for each organism, and for every position within this motif a 4x4-matrix is stored that specifies for every nucleotide the probability that another nucleotide occurs in the ortholog motif. All parameters concerned are initialized with random values and EM is used again to find the combination that leads to the highest P value of two motifs.

EMnEM (Expectation Maximization on an Evolutionary Mixture Model) (Moses *et al.*, 2004) is yet another EM algorithm like Meme. It needs the phylogenetic tree as an input, just like FootPrinter, takes sequences close together in the tree and searches alignments in them. The probabilistic model is hard to understand, but EMnEM seems to calculate the probability that a certain base is not generated by the (non-Markov) background and that – under a certain evolutionary model given as a substitution matrix – the base changed to one of the observed bases in one of the current alignments. This is similar to OrthoMeme except that OrthoMeme does not include a substitution model and only works on two sequences. The authors

tried different substitution models and found that the results depend heavily on it. A filtered, normal run of Meme often scores better than EMnEM with a bad substitution model, except for higher evolutionary distances, where EMnEM takes the lead.

PhyloCon (Wang & Stormo, 2003) is, not very surprisingly, the extension of Consensus for phylogenetically related sequences. If first runs wconsensus on all orthologous upstream regions for *one gene* and keeps all results that exceed a certain score (Consensus is calculating p-Values on information content). This is repeated for all genes. The resulting alignments are converted to weight matrices and those that originate from different genes are compared and collapsed if they are sufficiently similar (matrix similarity is called here "Averaged Loglikelihood").

So PhyloCon is two-step procedure that first searches short alignments across the species and then tries to find the same matrices on alignments from other genes. The authors justify their use of wconsensus by the fact that it allows no gaps and local-alignment tools like BLAST/CLUSTALW were not usable on this problem. The article contains detailed comparisons with other (non-phylogenetic) motif discovery programs. It is the first one that we have seen where sequence length of 10 kbp is applied without a decrease in performance.

2.4.2. Motif Discovery with expression data

As there is some error rate in the clustering of genes into co-regulated groups, the performance of discovery should improve if the unclustered complete expression data is incorporated directly into the motif discovery.

REDUCE (Bussemaker *et al.*, 2001) was the first approach of this kind. It runs an exhaustive motif discovery[25] for a motif length of 6 or 7 bp and then tries to fit a multivariate regression where each motif is supposed to have a linear influence on the final regression ratio. It was applied to yeast (Koerkamp *et al.*, 2002) and the fruitfly (Orian *et al.*, 2003, van Steensel *et al.*, 2003). MotifRegressor (Conlon *et al.*, 2003) looks like the very same procedure, also applied to yeast genes, just using MDScan for the discovery step and applying some additional filtering to the list of genes before the analysis. (Bannai *et al.*, 2004) (Bannai *et al.*, 2004) is also doing a regression on words, but is speeding it up by constructing a suffix tree first. (Holmes

[25] The authors own algorithm "MobyDick" is not used for this purpose

& Bruno, 2000) is a rather different algorithm, by combining k-means clustering with a Gibbs sampler, the resulting algorithm is called KIMONO.

cis/TF (Birnbaum *et al.*, 2001) is the inverse of the previous approach, i.e. regression first, discovery next. It needs the whole expression data over a certain time, i.e. sets from several experiments conducted one after the other, and a set of already known transcription factors. Then it tries to find out whether any composite (added) expression level of any subset of genes explains the expression of the transcription factor. If a set is found, an exhaustive pattern-driven search for 7bp-long words is run on the upstream regions. On yeast, in 10%-40% of validation examples, the real binding sites for the transcription factors were found.

2.5. Searching for regulatory modules

As in motif scanning, motif discovery can also increase its specificity by searching for combinations of binding sites, commonly called (transcriptional) modules. Usually this is a postprocessing phase, mining the result of a discovery algorithm to find a set of motifs that occur preferentially close together or at a certain distance.

The ModuleSearcher (Aerts *et al.*, 2003) is an A*-like approach. As such, it iteratively searches all combinations of motifs. The starting point is one single motif. Then another one is added and the new score is calculated as the sum of all ICs of all currently selected motifs' instances that are present in a 200bp-window. If the score improves, the current motif is added to the set and the algorithm iterates.

Some other, similar algorithms are more targeted towards working with matrices taken from databases like Transfac. We don't know yet if they are also applicable for motif discovery results, with their many false positives. Therefore we just mention these briefly: CISTER (Frith *et al.*, 2001) is training an HMM and calculating P values of it to report clusters of binding sites. COMET (Frith *et al.*, 2002) is an improvement that takes account of the motifs' order and reports more meaningful scores by applying a much more elaborate statistical model. MCAST (Bailey & Noble, 2003) is also using an HMM, but with a different background model. Cis-analyst (Berman *et al.*, 2002) simply uses a sliding window and marks those regions that exhibit a high number of binding sites close together; it is quite certainly not usable for a high number of sites like in motif discovery. MSCAN (Johansson *et al.*, 2003) (Alkema *et al.*, 2004) is searching for a combination of binding sites that minimizes their joint P Value within a particular window of sequence. The reader if referred

to Saurabh Sinha's technical report from 2000 that describes various combinatiorial motif models (pair, set, HMM, algebra...) and their applications (Sinha, 2000).

2.6. Does it really work? Some success stories

This section was inspired by the many conversations that the author had with biologists that were rather sceptical about the applicability of motif scanning and – to a higher extent – discovery. However, especially motif scanning is already applied quite often by biologists. We suppose that there are many more examples of successful searches for binding sites than the ones listed in the following paragraph.

(Tullai *et al.*, 2004), for instance, scanned co-regulated human genes from the same pathway with Match from Transfac, kept only over-represented hits and filtered out ubiquitous binding sites. The remaining sites could be verified in part by ChIP-experiments. (Mayer *et al.*, 2004) applied MotifSampler and MotifScanner (using the Transfac database), run from the Toucan interface (Aerts *et al.*, 2003), and kept only positions where both results overlapped. One of the predictions really bound the protein.

(Ramirez-Parra *et al.*, 2003) scanned for all hits to the E2F-matrix – which is bound by a group of transcription factors – on the Arabidopsis-genome's 800bp-upstream regions (Tigr.org's "Patmatch" software). With transgenic plants, where the binding domain of the E2F transcription factor had been deleted, they could verify that most of the genes really were expressed much less and that 60% of those contained at least two binding sites for the E2F-factors.

(Lockwood & Frayling, 2003, Lockwood *et al.*, 2003) did the inverse: They scanned human 2kbp-upstream regions with their own software at `http://www.bindgene.org`. Then they produced mice where a certain transcription factor was knocked out. Cells from these mice where then used to compare expression levels with normal mice. They found 28 genes where levels differed, of which 8 contained the predicted binding site.

(Berman *et al.*, 2002) and (Berman *et al.*, 2004) scanned the whole *drosophila* genome with Cis-Analyist for some binding sites that they suspected to occur close together and could successfully validate the predictions with experiments. (Halfon *et al.*, 2002) did almost the same, but scanned only for one particular binding site combination, just like (Markstein *et al.*, 2002). All of them validated their findings with biological experiments.

Motif discovery, as opposed to motif scanning, is a rather new method, so we were surprised to see a successful application as early as 2002: (GuhaThakurta *et al.*,

2002) used co-regulated genes from *C. elegans* that were mined with Consensus and Ann-Spec which identified a new over-represented motif[26]. The resulting matrix was searched on the genome to determine a P value for the hits and the top scoring ones were validated by aligning orthologous genes from *C.briggsae* with an algorithm called GLASS (Batzoglou *et al.*, 2000). One of the best hits were verified by inserting a fluorescent protein into the genes next to them and then mutating the predicted binding sites. The result was a dramatical reduction in expression when multiple sites were deleted and rather small changes with only one site removed.

In (Marchal *et al.*, 2004) the *S. typhimurium* genome was scanned against a motif model built from validated sites. For the hits, orthologous genes from 7 different bacteria were assembled. MotifSampler was run on the resulting gene groups, keeping only motifs that showed one hit per sequence. Then, a multiple alignment – with the results of MotifSampler serving only as seeds – was constructed, and a motif was considered relevant if the region around it was "sufficiently" conserved: A special heuristic has been developed for this task and is documented. Four of the predictions were selected and their genes all showed significant change in expression when the transcription factors were suppressed. With the new motif model (sites from literature and newly verified sites) the genome was scanned again and other putative targets for the PmrAB transcription factor are postulated.

(Markstein *et al.*, 2004) used Mermaid to search known enhancer regions for recurrent motifs. Three binding sites were returned of which one was not known before. The resulting improved enhancer model was used to predict other target genes of the unknown transcription factor, of which one was verified with various deletion experiments.

The application of the Ahab algorithm, mentioned on page 35, to the genome of *drosophila* lead to 16 predictions of which 13 could be verified with experiments. (Schroeder *et al.*, 2004).

The most outstanding result of the phylogenetic approaches on yeast is a recent article (Harbison *et al.*, 2004) that did a complete prediction of *all* binding sites with high accuracy. They found weight matrices for all 204 transcription factors by applying five different motif discovery algorithms (MEME, MDScan, Converge, Kellis et al., AlignACE) to promoters of 12 microarray-data selected gene-groups and checked if matches to them were conserved among four *sensu stricto Saccharomyces* species. The result is a list of 3353 binding sites, the first complete collection for

[26] Note that the authors did not use their own motif discovery program, CoBind. It seems that the dyad-model was not applicable for the example.

a eucaryotic genome. From these data, they derive four specific types of promoter architecture:

1. One binding site, one copy: Mostly for general genes, expressed in many different, basal pathways. Standard transcriptional response.
2. One binding site, several copies: Needed for some rare TFs for binding or a graded transcriptional response.
3. Different binding sites, no order: Combinatorial regulation, response varies according to growth conditions.
4. Different binding sites, fixed combinations: Two TFs have related functions or interact physically.

They also note that more specific binding sites are usually *not* located in the region 0-100bp upstream of TSS, as this is bound by the transcription initiation apparatus.

2.7. The algorithm assessment in 2004

Although motif discovery benchmarks have been done before [27] (Lescot, 2002) (Pevzner & Sze, 2000) (Sinha & Tompa, 2003) (Sze *et al.*, 2002), they were either not realistic or not independent: Mr. Pevzner's comparison was done with artificial datasets, reflecting motif models that not even slightly tried to reflect known fundamental biological properties of binding sites but rather "difficult" sites from the point of view of computer science. The others were prepared by authors of some algorithm that, as a result, won the comparison. We don't want to suggest that all comparisons were unfair from the outset; the bias might be simply due to complex parameter-setting and insufficient experience with the other algorithms. So Prof. Tompa, whose students published quite a few papers on the motif discovery issue, started a benchmark in 2003 on realistic data from different organisms, covering different types of motifs.

There are three kinds of datasets:

1. Natural, real sequences with validated motifs from Transfac
2. HMM-generated sequences, trained on the species where the matrices originate, with inserted motifs, derived from real weight matrices.
3. Sequences generated from a simple multinomial model with inserted motifs, derived from a weight matrix

The 56 matrices come from yeast, drosophila, mouse and human organisms. For every species, there exist all three kinds of datasets. They have different lengths,

[27] One comparison has been done long ago (Hudak & Mcclure, 1999), but it only tested protein motifs which are longer and therefore easier to find than binding sites on DNA.

500 to 3000 bp. Every sequence contains about one motif (for the real sequences there might be more), whereas four negative sequences without any motif were added to test their influence on the performance. 45 binding sites have a length of 35-71 bps. Unfortunately, our analysis of the results is very limited, since the authors did not publish information about the hidden binding sites, we don't know how long they were in every set, how they were chosen from Transfac or anything about their nucleotide-composition.

The sequences were downloaded by the authors of the motif discovery algorithms who were supposed to run their tools and document the parameters used. Then they could return one single motif prediction per sequence group, in a common format, by specifying the alignment, i.e. the positions of the instances. This might seem rather strict, as a biologist would never look only at the number-one prediction. But since there were many sequence groups to analyze, roughly 50 per species, the final result should not depend too much on some single errors. So the setup is realistic enough to come close to a real application and certainly better than all other comparisons started before.

It was possible to cheat in this competition by scanning with Transfac's matrices to find out the true motif hits, but since all steps and parameters had to be documented and annotated, a special tuning to fit and find a certain motif should be visible from the supporting documentation that the participants had to supply.

The final scores were calculated as the number of overlapping positions between real and predicted sites. In a second type of scoring scheme, a putative site was considered a hit, if it overlapped the real site by 25% of its total length, reasoning that deleting this part would cause expression to change in a wetlab experiment. We focus on the position-based scores here, as they are not very different from the site-based calculations.

The following abbreviations are used to specify how the scores were calculated: tp (true positives) is the length of the overlap between real and predicted positions and fp (false positives) the number of predicted nucleotides that do not overlap the real ones, tn (true negatives) the number of non-predicted nucleotides that are also non-real and fn (false negatives) the number of non-predicted nucleotides that are real.

$$Sens = \frac{tp}{tp + fn}$$

$$Spec = \frac{tn}{tn + fp}$$

$$ppv = \frac{tp}{tp + fp}$$

$$CC = \frac{tp \cdot tn - fn \cdot fp}{\sqrt{(tp + fn)(tn + fp)(tp + fp)(tn + fn)}}$$

Sensitivity gives the fraction of known binding site nucleotides that are predicted and Specificity ppv, the positive predictive value, gives the fraction of predicted site nucleotides that are known. Specificity for all programs is in the range 97%-100%. Since the number of allowed predictions per program and the number of implanted sites are fixed, the three measures are highly correlated and one can deduced if the other's variables are given [28]. Therefore, we will focus on sensitivity, PPV and CC in the following. *CC* is the correlation coefficient, according to the authors, "the Pearson product-moment coefficient of correlation in the particular case of two binary variables" (Tompa *et al.*, 2005). CC has the advantage to be undefined (in our diagrams, we have put 0 instead) if a program does not predict any motif. PPV seems to favor programs that make rather no prediction instead of wrong ones. Tompa et al. prefer the CC measure in their report.

Given all this, we can easily draw the final results, by summing the scores over all sequences and calculating how many of the maximum number of positions every algorithm catched: See Figure 2.6, which seems to show Weeder as a clear winner. It leads the field in the two more comprehensive measures, CC and PPV. However, Figure 2.7, with the correlations split by organism, presents a different image when it comes to metazoans: Performance is much lower for higher eukaryotes, with the fly being the most extreme example, where many algorithms score less than zero: Their predictions are completely off the track.

Why is performance different between organisms? Is it due to motif IC, the number of sequences used or because of different motif lengths? To check this, we aligned 35 of the 56 *site-sequences*[29] with Meme[30].

This should give a good approximation of the real motif's weight matrix, as the sequences were very short, only 10-30 basepairs long and could be easily aligned by

[28] If the number of allowed predictions $tp + fp = p$ and the number of implanted positions $fn + tn = n$ are known, it follows: $Spec = \frac{n - fn}{(n - fn) + (p - tp)}$. Now $Spec$ does only depend on the variables in $Sens$ (tp, fn) and the constants p, n.

[29] The data of the assessment consists of excerpts from Transfac's SITE table, i.e. short sequences from real annotated promotors. They vary in length and often contain a couple of flanking bases. We call them site-sequences in the following, as they are more than just binding sites.

[30] Meme imposes a minimal length on the alignment, therefore, from every organism 3-5 sets could not be processed (exception: fruitfly, all could be aligned, note that this might skew the data for the fruitfly in our analysis). We also tried ClustalW, but it could not cope with motifs on the reverse-strand. This kind of exercise is *not* motif discovery as treated in the rest of this study: We merely used Meme as a replacement for ClustalW. There was almost no noise at all in the data, no flanking promotor, only pure binding sites with some additional bases.

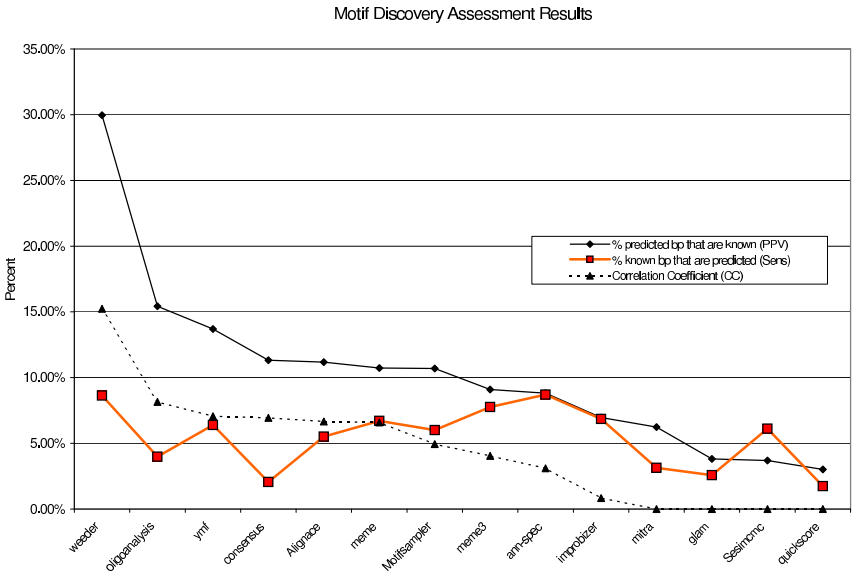


Figure 2.6.: Results of the motif discovery assessment over all sets

Meme. Figure 2.8 shows the results. They indicate that the motifs used from yeast were longer than those from the other organisms. Average motif width explains why the Information Content of the motifs is also lower for human/mouse/fly than for yeast. The fraction $\frac{AvgIC}{Avglength}$ is almost identical across all species, roughly 1.0-1.1 bits/bp (data not shown). The case of the human data set hints that giving more sequences to the algorithms does not change the results a lot. This is not a surprise, since noise and computational complexity also increase. Our analysis does not explain at all, however, why performance is so extremely poor for the fruit fly data set. We hypothesize that the structure (no core?) of the motifs might play a role here, but we lack a formula to put the structure of a motif's conserved positions into a good score.

Figure 2.8 indicates that motifs of yeast in the assessment were much longer than those from other species. Is this a bias of the test data or a general rule for yeast? To check this, we compute the matrix length in Transfac by organism. The result is plotted in Figure 2.9 and indicates that this is not a general property of yeast but more a bias in the assessment's data. It also shows a strange feature of yeast matrices in TRANSFAC: They contain either very long or very short sites, which can not be expressed by an average value. We have no explanation for this. Unfortunately,

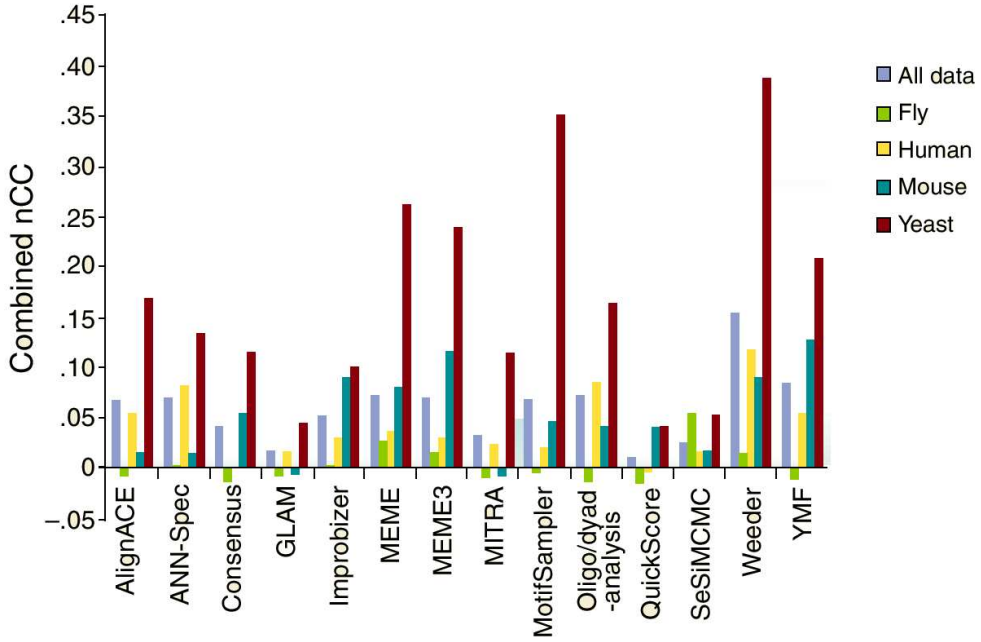


Figure 2.7.: Results of the assessment by organism
Diagram copied from (Tompa *et al.*, 2005)

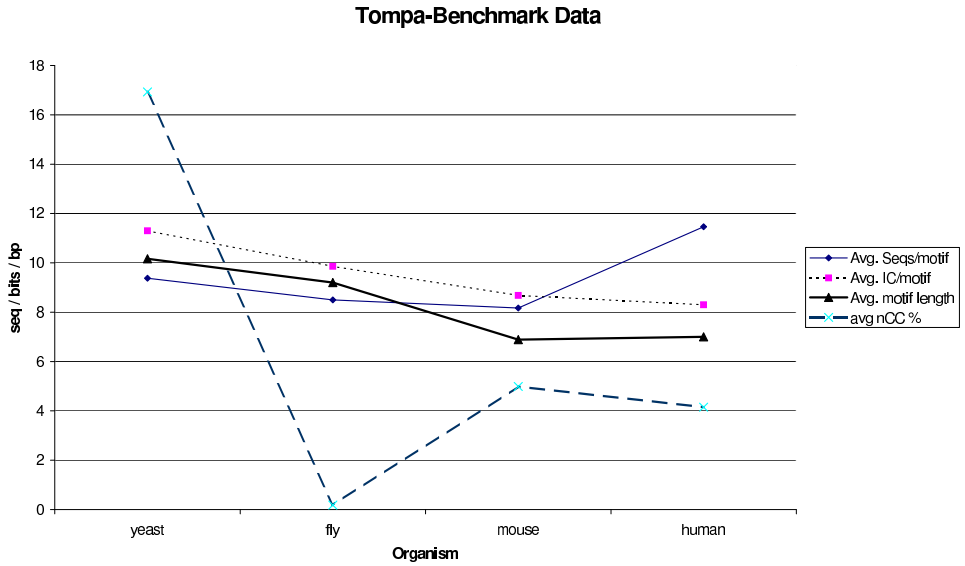


Figure 2.8.: Analysis of the motifs used in the assessment by organism

Avg. Seqs / motifs specifies the average number of sequences in data sets for this model organism from the assessment; *Avg. IC* is the average IC of the motifs from this model organism (see notes); *Avg. motif length* is these motifs' average length and *Avg. CC* is the averaged correlation coefficient in the assessment over all algorithms, multiplied by 100. IC and length were reported by Meme, CC and sequence counts were taken from the data in (Tompa *et al.*, 2005) (we always used data from assessment on basepair-level, not on the "per site"-level).

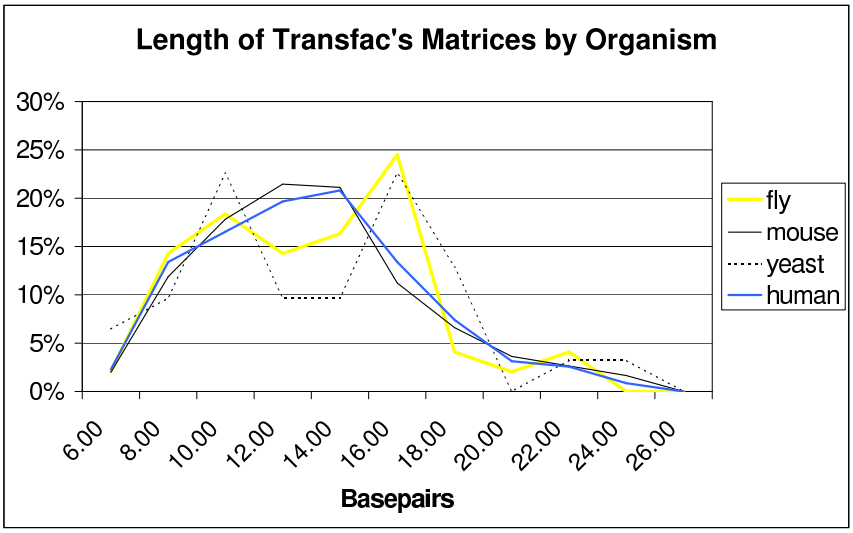


Figure 2.9.: Transfac Matrix Length by organism
See figure 1.2 for a general curve

Transfac's data on yeast is not very comprehensive, it contains only about 50 matrices (see page 16), so this might also be a flaw of the database which is specialized in eukaryotes.[31]

The benchmark's article (Tompa *et al.*, 2005) does not try any explanations *why* the results differ. It is a clear contribution of this thesis to compare the inner workings of the algorithms and we can therefore try to derive reasons for their performance. The basic setup of the algorithm is not very important. The best candidates are suffix trees, probabilistic and enumerative methods. They use various kinds of z-, significance- or IC-derived scores. We were surprised to see that algorithms tailored to yeast did not dominate the field for the yeast data set, see Figure 2.10. We can hypothesize two other important factors that might have made a difference, but only the first one is related to the algorithm itself:

Core Positions

The good scoring of Weeder is interesting, but not very surprising, as it is the only algorithm that takes into account conserved core positions of motifs, just like motif

[31] We did not have enough time to check if the 204 matrices from (Harbison *et al.*, 2004) show the same properties.

Ranking according to CC performance, by organism

on yeast	on human	on mouse	on fly
Weeder	Weeder	YMF	SesiMCMC
MotifSampler	oligodyadanalysis	Meme3	Meme
Meme	Ann-Spec	Improbizer	Meme3
Meme3	AlignACE	Weeder	Weeder
YMF	YMF	Meme	Ann-Spec
AlignACE	Meme	Consensus	Improbizer
oligodyadanalysis	Improbizer	MotifSampler	MotifSampler
Ann-Spec	Meme3	oligodyadanalysis	AlignACE
Consensus	MITRA	QuickScore	MITRA
MITRA	MotifSampler	AlignACE	GLAM
Improbizer	GLAM	SesiMCMC	Consensus
SesiMCMC	SesiMCMC	Ann-Spec	YMF
GLAM	QuickScore	GLAM	oligodyadanalysis
QuickScore	Consensus	MITRA	QuickScore

Figure 2.10.: Ranking of the algorithms by organism

scanners have done since quite a while (Quandt *et al.*, 1995). During the last few years, most authors neglected the important notion of a core in binding sites. By "core", we mean the property that usually some positions in the binding sites don't change at all and others can contain any nucleic acid. The core is usually packed together, so a motif like A.C.G.T.CC.G is not very probable, whereas AACGT....CGTAC seems biologically more realistic. Therefore, motif positions are not completely independant of each other, as in the binomial model. However, it's not the letter itself that influences its neighbours (like in an HMM) but the degree of conservation of a letter. The idea of a core was sometimes blankly ignored.[32]

Motif Length

The assessment's authors state that "many of the binding sites catalogued in TRANSFAC are unusually long: 31-71 bp in length" and that this "might have a detrimental effect on measured sensitivity". Looking at the assessment data as well as TRANFAC, we cannot confirm this finding and wonder why Tompa et al. regard TRANSFAC's

[32] Example: For Multiprofiler, an anonymous reviewer pointed out to the author Uri Keich that the positions of known sites from Transfac are often not independant, though this is an assumption of his software. Uri Keich replied that "the assumption of independence makes the motif-finding problem harder rather than easier, so it should not be considered a flaw in the algorithm." (Keich & Pevzner, 2002*a*). This is obviously quite the contrary. Ignoring an important part of the problem rather degrades the quality of the scoring function than constituting an advantage.

SITE table as *pure* data. The SITEs just collect data from publications, they merely consitute a list of binding sites with some flanking bases. We see no reason to believe that a couple of these bases could mislead algorithms very much, as about 2000 basepairs noise were added anyways to the binding sites during the assessment. TRANCSFAC's MATRIX table, that we used for our binding site length- and IC-analysis in Figures 1.2 and 1.8, stripped these and reflects the real lengths very well. When aligning the assessments sites with MEME, we find motifs with similar lengths as in TRANSFAC, as shown in Figure 2.8.

However, the data from yeast shows well that the motifs length plays a role when uncovering motifs. Longer motifs are easier to discriminate against the noise and every basepair counts.

Conclusions from the assessment

The authors of the assessment proposed some improvements for the next similar benchmark: Eliminate real datasets (they can include other, unknown motifs which might confuse some algorithms, but are not included in the score), consider not only the very best, but the first, say, three hits and eliminate negative sequence sets.

Some articles suggested before that most of the different search methods lead to very similar results [33]. It seems that even YMF with its simple enumerative approach can challenge sophisticated samplers (the mouse dataset). In contrast, a good scoring function obviously makes a big difference, as seen with Weeder. The scoring certainly should include the possibility to compare motifs of different lengths, as this is never known in a realistic setting. Further ideas for future work are given on page 99.

[33] ''improved search algorithms will not bring about fundamental improvements" (Frith *et al.*, 2004) but this seems to be deduced from running only one algorithm with different parameters.

3. Jannotatix - a tool simplifying the work on prediction algorithms

3.1. Choosing the best algorithm for motif discovery

As can be seen from the preceding chapter, approaches for motif discovery are quite different. And although there are not too many basic search algorithms, implementations are plentiful (see Table 2.1).

Having many slightly different methods at hand is a situation very common in bioinformatics, well known from protein structure prediction, for instance. There, the competition CASP/CAFASP (Fischer & Rychlewski, 2003) is held every two years to shed light on the performance of the approaches: Given a certain input set of data where the result is known and biologically validated, the algorithms are run by their authors and their output compared to the expected prediction. As it is obvious from the Tompa-assessment, setting up a competition is not a question of few hours:

- Test data: As there exist different types of data and model organisms, it takes considerable time to gather sets of test data that are sufficiently comprehensive. The assessment with 52 data sets was already rather small.
- Prepare sequences: Upstream sequences have to be downloaded, cleaned from repeats and tailored into sets, where known data has to be annotated. Random-data should be added as a cross-check. This takes time as well, if not automated by some kind of software. The assessment's data did still contain repeats. The organizers left this task to the participants' teams, which used different and sometimes no filtering at all, which makes comparing the results even more difficult.
- Prepare implementations: Algorithms have to be downloaded, compiled and run. For the comparisons, scripts have to be written, as every program returns results in a different format. Differences in format include: Truncated sequence names, coordinate changes (zero-based, one-based), coordinate on negative strand relative to start instead of end of sequence, different kinds of scores for one single binding site. In addition, some implementations are only available for a certain operating system, Windows, for example. The assessment asked authors to return their results in a fixed format, so avoided all this easily.

- Parameters: Every algorithm has different parameters. Setting these to sensible values involves knowledge of the inner working of the algorithm and some assumptions about the problem (e.g. biologists never know the strand to search, so fixing it to some value is usually unrealistic). Setting parameters has direct influence on the performance, some algorithms can go completely off the track if not tuned properly: (Poluliakh *et al.*, 2002) tried many different parameters for Gibbs and Meme. They found out that the default parameters of the Gibbs Sampler find 0% of the real motifs on their test set, whereas the best one found up to 80%. Meme's performance easily doubled when supplied with the best parameters. This illustrates how crucial the correct parameters are when running motif discovery programs. The assessment circumvented this problem by asking authors themselves to run algorithms and supply optimal parameters, which resembles the setup of the CASP-competitions.
- Comparison Measure: All the data has to be saved in one common format to make comparison possible. Some kind of score has to be created to be able to compare the results. The assessment used all scores that are known from the gene-scanner field at the moment.

Automatizing these four steps as well as possible is the aim of Jannotatix. As some of them need visual inspection of the data (sequence preparation, visualization of results) a graphical user interface is a necessity.

The assessment done by the Tompa group lead to interesting results but suffers from some drawbacks: Some groups used manual selection to pick out the best looking motif, the final ranking of the algorithms poses more questions than it answers and the limitation of only one single motif allowed in the results is not realistic. It would be interesting to see a comparison on other data sets, for other parameters and a relaxed notion of what is considered a "hit", taking into account the best 10 predictions, for instance. But repeating the assessment would mean a lot of time and work for all people involved.

Actually, every author that develops a new algorithm has to do the same to prove that his or her one is superior to its predecessors. This is similar to protein structure prediction or gene scanning. It is why in these fields, some solutions to the problem of comparing algorithms were found : The gene scanner community is using a common format (Gene Feature Format (Durbin & Haussler, Year)) and in structure prediction there is a ready-made set of scripts available that run algorithms, parse and compare the results automatically (Bujnicki *et al.*, 2001).

In motif discovery, many biologists would probably like to get a feeling for the algorithms themselves. They could do this by using their own data and comparing

predictions manually, like in (Lescot, 2002). But it would clearly be an advantage to have helper-tools like in the structure prediction community. The whole process of sequences preparation, sequence cleaning and running algorithms is the same for a biologist trying to find binding sites as for computer scientists trying to compare algorithms. Therefore, it seems obvious to automate the whole process, with a tool that ends tedious script-writing and makes complicated algorithm-execution – meant for computer scientists anyways – easier to use. It can be used by authors of algorithms to validate the performances, by everyone to compare algorithms in some sort of contest and by biologists while working on their own data.

As motif discovery is only one short step in unraveling potential binding sites, it would be nice to have this step not overcomplicated by rather tedious software and data format issues. As motif discovery is a field still changing every few months, the tool should not depend on a central server with a fixed set of algorithms, like Toucan (Aerts *et al.*, 2003) or SockEye (Montgomery *et al.*, 2004). Tompa et al. in (Tompa *et al.*, 2005) state the same: "Biologists would be well advised to use a few complimentary tools in combination rather than relying on a single one". It should be possible to extend the platform easily on the local machine, to ´´play" with the results. As discovery is computationally quite time-consuming, few people would have the resources to afford running a central server for other people in the long run. So the solution would be a program that is very flexible, multi-platform, but still running locally, which should boast some kind of simple interface so that it can be used by computer scientists and biologists alike.

3.2. Existing Motif Discovery Platforms

Since 1998, the RSA-Tools by Jacques van Helden (van Helden *et al.*, 2000*a*) (van Helden, 2003) are an invaluable resource that integrates several motif discovery programs, an upstream-region-database and postprocessing applications under one common web-interface. They are tested on biological samples and include a lot of verification data and interesting tutorials. Unfortunately, they only support yeast.

Imitating RSAT, in 2003/2004 a couple of new tools appeared that integrate several algorithms. Mentioned in the last section, Toucan (Aerts *et al.*, 2003) is a locally installed Java interface but works with a central server located at Leuven University and supports only the MotifSampler algorithm developed by this group. Melina (Poluliakh *et al.*, 2003) is a web interface and runs AlignAce, Gibbs Sampler, CoResearch and Meme, just like the more recent RGSMiner (Huang *et al.*, 2004), which supports AlignACE, Gibbs and Meme. RgsMiner and Toucan are more ad-

vanced as they can download sequences automatically and annotate putative sites from Transfac. Both also try to find recurrent combinations of sites (only pairs at the moment). Also a web interface and similar to Melina, but really much more comprehensive with a massive list of 22 supported algorithms is Mogul (Rust *et al.*, 2003). Unfortunately, it is not publicly accessible yet, due to licence restrictions. Almost identical to Jannotatix (excluding plugins) is HitPlotter [1] by the same group that also wrote Sockeye (see below). But HitPlotter is still in very early beta-stage.

Although started as a comparative 3D sequence viewer, Sockeye (Montgomery *et al.*, 2004), written in Java, now also includes support for motif discovery – AlignACE, Meme, Gibbs, ElPH, MotifSampler and Consensus at the time of writing – and Multiple Alignment for cross-species comparison as well – MLagan,ClustalW, DiAlign and more. Given its highly skilled, numerous authors and the very nice interface, Sockeye will in the long run surpass most other academic tools. Its powerful backend architecture called Chinook is a peer-to-peer based system that runs algorithms on a network of distributed machines, something that no other interface tried yet.

Sockeye and Toucan are now open-source, so they can in theory be tailored to the user's needs if he or she has a thorough knowledge of Java, object-oriented-development and Biojava. Source-code is naturally not intended to be modified by a user, so adaptation is profoundly difficult, as it is easy to get lost in some 20,000+ lines of code. Chinook is similar to Jannotatix, as it also supports many algorithms and tries to be flexible but to add a new one, Java programming is still needed.

The only commercial solutions that are known to the author are SRMS and Atragene's Pattern Explorer, both appeared at the end of 2003. SRMS (`http://www.silicoinformatics.com`) supports download and clearing of sequences and Gibbs and Meme. Atragene Pattern Explorer (`http://www.atragene.com`) seems to run some kind of Gibbs Sampler and Transfac but the company refused to send a demo version of its product.

3.3. Jannotatix — a flexible tool for working with feature prediction

Our program Jannotatix tries to combine flexibility and ease of use. It's main advantage is the concept of plugins, borrowed from very succesful open-source projects like IBM's Eclipse. Once the program is run, the user can select the algorithms he or she prefers. They are then downloaded and installed onto the user's harddisk and can be run from the interface. An algorithm can be almost anything: A com-

[1] see `http://www.cisred.org/content/software/index_html`, there does not seem to exist many publications on this software yet.

piled binary executable, a Java-class[2] or a website accepting user input as http-post requests.

Plugins are written in standard XML that specifies general info about the algorithm, how to run it, what parameters of which type it accepts and how to parse the results. They can be created using any texteditor and look very similar to a html-page. Packed into a zip-file together with any accompanying binaries and data, they represent what we call a *Jannotatix plugin*, are usually made accessible by putting them in a *plugin list* on some webserver and can be installed with the *plugin manager* from within the Jannotatix interface.

This interface is a graphical Java application that runs on most operating systems. It displays the sequences and all features added to it from any of the plugins. It acts as a central data storage for all predictions, saving and reading standard GFF-files[3]. It allows for easy navigation through the features, filtering, separating into tracks and showing properties associated to features by the plugins.

This GUI can be used independently, without any plugins, as a web-based viewer for GFF-files. It starts directly from the webbrowser as long as Java is installed.[4] The idea was to make it possible for the amino-acids group at Max-Plank-Institute, Golm, to use the viewer for their own motif annotation system which relies on data from the Agris-database (Davuluri *et al.*, 2003) of putative binding sites, but this is still in development.

The algorithm repository on the local machine is a directory which holds all installed plugins. We currently supply plugins for AlignACE, MotifSampler, PlantCare and Transfac.[5] These can be run from the interface; multi-processor machines are supported by executing several instances in parallel. For server side use, a single-processor command-line mode is envisaged and partly implemented, which supports

[2] Actually, this is not completely supported yet, as there are few examples written in Java, but the source code is designed to support it with an abstract interface being supplied.

[3] The author chose GFF inspired by Toucan, but at the moment most sequence viewers can read GFF, like Artemis (Rutherford *et al.*, 2000), Apollo (Lewis *et al.*, 2002) or Sockeye (Montgomery *et al.*, 2004), and popular web sites like the UCSD Genome Browser or Ensembl, export to GFF. The format is very simple; on every line, the following data is listed, separated by a tab-character: SequenceName, Source (=Software that generated the feature), FeatureType (e.g. binding site), Startposition, Endposition, Score, Strand, Frame (not applicable for binding sites), various semicolon-separated tags (optional)

[4] Java Webstart is needed, which is configured by default on Windows but not on Linux. In Linux, the user has to associate the filetype manually with the Webstart executable that comes with Sun's Java Runtime Environment

[5] In order to include other programs, someone has to write suitable XML-files. There is a short example on the Jannotatix-Website how to do this, but no comprehensive documentation of the file format yet.

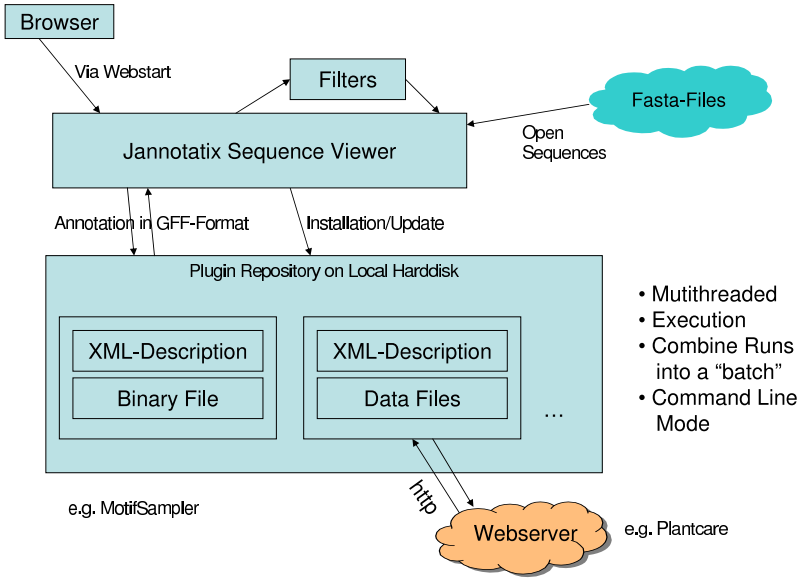


Figure 3.1.: General structure of Jannotatix

the basic functionality from the GUI. The idea here is to use Jannotatix as part of RGSMiner (Huang *et al.*, 2004) at National Central University, Taiwan, to simplify incorporation of new algorithms in the future.

The whole process is illustrated by Figure 3.1: Jannotatix is launched from the web browser. The application starts and displays available sequences with motifs on them, which can be filtered. When the user selects an algorithm from the repository, it is run (either locally or on a webserver via an http-request) as described by its XML-file. The results are again displayed and can be filtered. Automatic sequence download is envisaged but not implemented yet.

3.4. Structure of the source code

The source code consists of roughly 170 classes. The relationships between the most important ones are depicted in Figure 3.6 with the UML. The main class is Aligned-Sequences. It uses SequenceDB (from BioJava) that stores sequences, which can be im- and exported to various fileformats, like FastA or EMBL. Every Sequence has annotations and an ID associated with it and can store features. The concrete Sequence in this case is TrackedSequence, derived from BioJavas Sequence-class, which adds

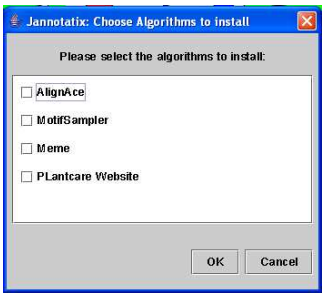


Figure 3.2.: Screenshot: Installation of discovery algorithms made simple
When the program is started for the first time, the user can select the algorithms he wants to install

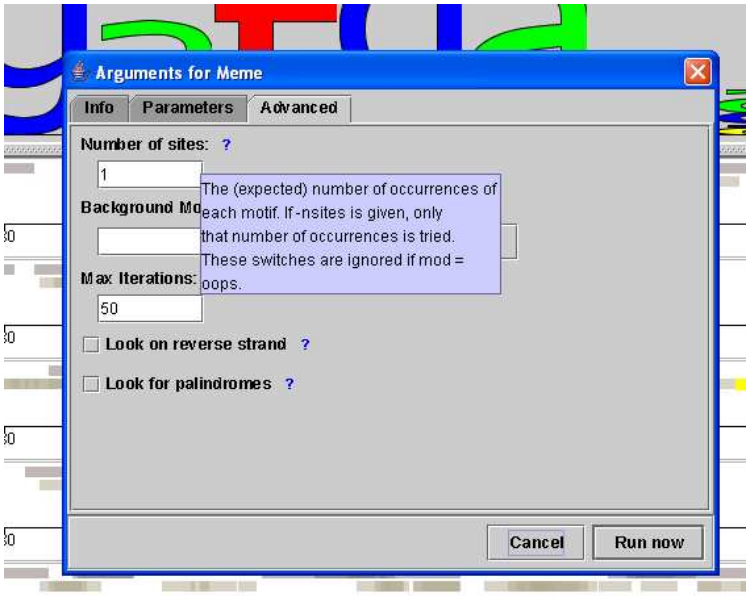


Figure 3.3.: Screenshot: Running MEME
Parameters are explained by tool tips

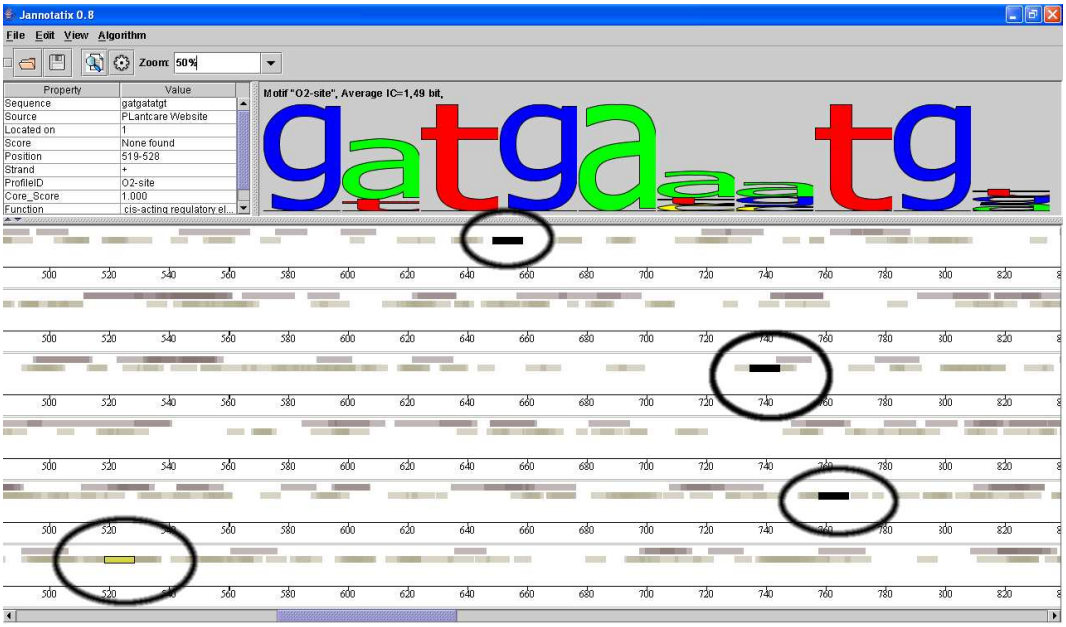


Figure 3.4.: Screenshot: Sequence View

After AlignACE and PlantCARE are run, the results can be explored with the Sequence Viewer, which shows the sequences and motifs on them. There is a separate track (line) for each program's results. In this example, an implanted O2-site (selected and highlighted with a circle) was found by PlantCARE, but not by AlignACE (the track above).

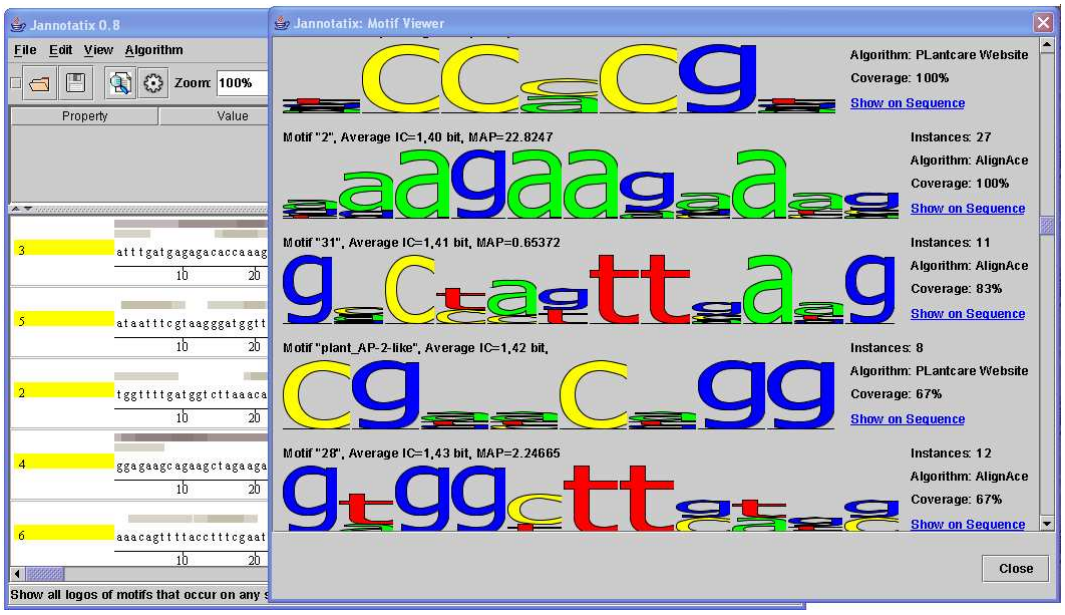


Figure 3.5.: Screenshot: Motif View
The same data as in the preceding figure, but the motifs are sorted by total Information Content, with some additional data.

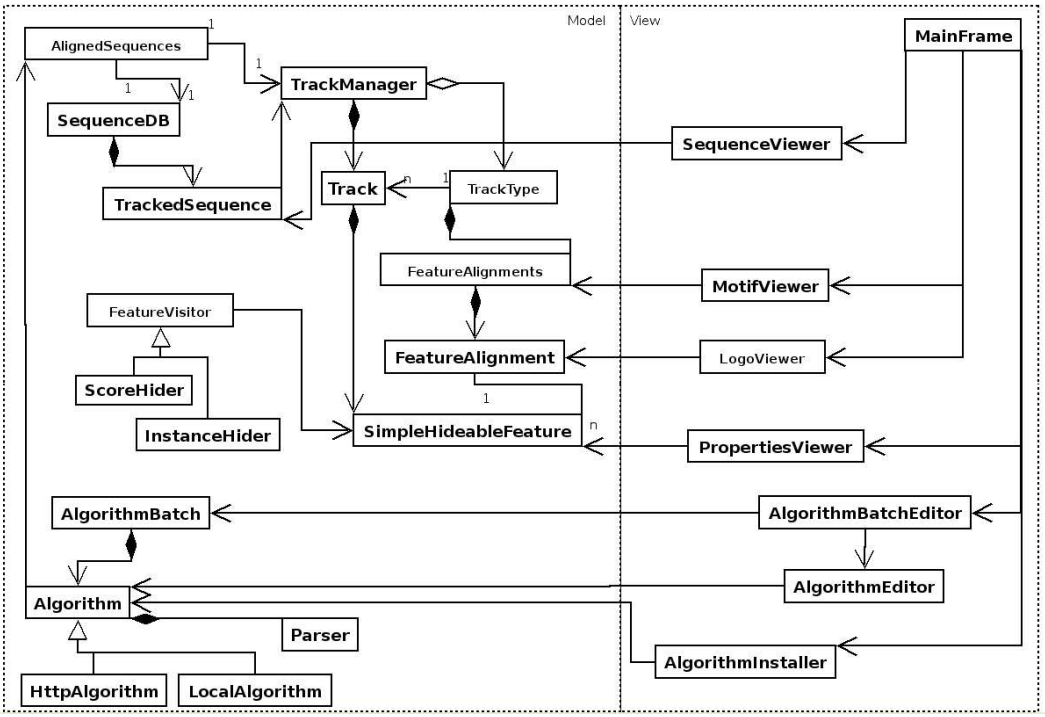


Figure 3.6.: UML Diagram of Jannotatix' Main Classes

the notion of a *Track*: Whenever a feature is added to a TrackedSequence, it is put into a *Track*, i.e. a collection of features that originates from the same source (i.e. program/annotator/algorithm). Every AlignedSequences-object has one TrackManager that is used whener a new track needs to be created or removed. It assigns every track to a TrackType (we are still searching for a better name): A TrackType references all tracks that share the same source from different sequences and contains attributes like color and visibility. Whereas a *track* relates to only one sequence and contains many features, a *tracktype* relates a only one AlignedSequences-Object and contains many tracks.

SimpleHideableFeature is a highly extended version of BioJava's SimpleFeature that contains various attributes, the most important being the *Hidden* attribute. We do not change the visibility of features on the screen with BioJava's Filters, though they originally were designed for this purpose: They proved to be way too slow. Instead, a FeatureVisitor is configured with appropriate parameters (e.g. filter out all features with score < 1.2) and iterates over a collection of features in the background. Features that share the same ID are grouped into a FeatureAlignment object. It is in turn assigned to the FeatureAlignments object, one for every TrackType.

An Algorithm is either an external program or run via HTTP, retrieves sequences from the AlignedSequences object and adds features to its TrackedSequences.

Example:

The user opens a sequence file (AlignedSequences.openFasta). This will create many TrackedSequences with no features and no track. He or she opens a GFF-file from the harddisk (AlignedSequences.openGff), where all features have a "Transfac"-source field: There is no track yet for "Transfac", so the TrackManager is asked to create a new TrackType and a Track on every TrackedSequence. A feature consists of the following data: start-/endposition, strand, sequence, source, various kinds of scores (depending on source-program) and an alignment-ID. All features with the same alignment ID constitute a motif.

The features are created, filled with the right attributes from the file, added to the Tracks and at the same time sorted into the current alignment, according to their alignment ID. To improve the speed of this process, FeatureAlignments internally uses a hashed map, to look up very quickly the correct FeatureAligment for a given alignment ID.

A SequenceViewer is created on the screen. It paints all sequences and features in their colors as specified by their attributes. When a user clicks onto a feature, a PropertyViewer is created for it and the alignment is transformed internally into a

probility distribution from which the LogoViewer calculates and draws the Logo for this alignment. If the user selects the menu View - Motifs, the MotifViewer will display all available alignments, sorted by some criterion (Currently Total Information Content but this can be changed very easily). If the user selects View - Filter, a FeatureVisitor is created and run (A filter on the alignment-level is not yet implemented but would be more useful than the current feature-based one).

All algorithms are initialized from their XML-files, together with their respective parsers. If one of them is selected, the AlignedSequences are written to a file (or to a temporary string, to spool them via HTTP to the algorithm). The Algorithm is run (key words from its output are used to update the progress bar on the screen). The parser transforms the final output into a GFF-file which is read again by the AlignedSequences object.

3.5. Open-source libraries

Jannotatix' roughly 12,750[6] lines of code would have been never possible without the help of the people from the Biojava mailing lists and heavily rely on a couple of other open-source projects. Reading and writing XML files is done via Castor (`http://www.castor.org`). The format of the plugin files was specified in an abstract, grammar-like way with XML-schema (`http://www.w3.org/XML/Schema`), from which Castor generates classes that are used to read/write the files. Features and annotations are stored, displayed read and written with modified versions of Biojava's (Mangalam, 2002) classes. The whole program makes use of many of the Commons (`http://jakarta.apache.org/commons`) libraries from the Apache Jakarta project, like the command-line mode, the http-client, the logging system or the configuration-file parsing. Multiprocessor support depends on the libraries from a well-known book by Doug Lea about Java concurrency (Doug, 1999). The GUI-part uses the XML-Actions framework to simplify coding all kinds of menus, icons and buttons (Davidson, 2003).

[6] 19,000 if the XML-parsers are counted, but these are generated from a grammar by Castor.

4. Directions for future work

From the more than 250 references in this work, a few ideas emerge that are summarized in the following. We go along more or less with (Helden *et al.*, 1998): "Our feeling is that in the future the emphasis should be put on incorporation of most available biological information rather than on the development of elaborated statistical methods." As obvious from the list of motif discovery algorithms and the assessment results, this advice was not really acted on.

4.1. Future work

- Does the search method in motif discovery really make a difference? Looking at the assessment one could be inclined to share the opinion of (Frith *et al.*, 2004) that, indeed, it does not. This leads to the question why more than 50 articles were published that focus a lot on different search methods and very few authors tried to improve only the score function and just the score function using biological examples. Running algorithms with completely different search methods and putting one common score function for all of them into, for example, Jannotatix could be used to verify if the search method makes a difference and put hopefully an end to the torrent of different motif discovery algorithm implementations.
- Is there really a difference between matrix-based and string-based search? If all possible IUPAC-symbols are used and information content is calculated on the strings, the results should be similar. And if strings are applied as seeds to scan for a profile, results should become nearly identical (Eskin & Pevzner, 2002). Very recent research indicates that the difference is not very high, probably close to a few percent (Philippakis *et al.*, 2005). In addition, some of the best scoring methods in the motif discovery assessment were not weight-matrix based. To verify this, one could scan all of the human genome's upstream sequences (downloadable from UCSD) with Transfac's MatInspector with matrices and afterwards with the respective consensus sequences. If there are no differences, this could at last end the year-long struggle between matrix and string approaches. Consensus sequences could incorporate the notion of uppercase/lowercase letters as a way to visualize additional information, as

proposed by (Eskin, 2004). This could lead to a way to store binding sites as strings which would be much simpler to handle than matrices but more expressive than traditional consensi.

- Cross-species data should be read, saved and visualized in a common interface and data format for Jannotatix. Multiple alignment programs should be integrated as plugins, just like motif discovery programs now. This would necessitate adaptation of the XML plugin file format for a new type, SequenceAlignment. The visualisation of multiple alignments on multiple sequences is, however, not trivial.
- The question on how to find a TSS on the human genome remains difficult. Jannotatix could be extended to aid here, displaying all information that is relevant: TSS-Predictions (like those given by FirstEF/Eponine (see Appendix B) and ESTs from databases can be shown on a preliminary 20kb upstream-5 kb downstream region that could then further be trimmed by the user.
- With the advent of the chimpanzee genome, a good score function is needed to improve binding site detection in human-chimpanzee, human-mouse, mouse-rat alignments. With test data from Transfac, motif discovery and alignment done with one of the algorithms available in Jannotatix: How can the detection rate be improved by mixing comparative genomics, discovery and motif? Using Jannotatix, all these data would be available in one data format, and various scores could be tested on a combination of different algorithms.

4.2. Ideas for Another Motif Discovery Algorithm

From the vast literature and the assessment, many ideas can be extracted that could guide the design of a future algorithm. Most of this is not necessarily difficult to implement, as few statistical methods are used and even suffix trees are constructed easily with one of the many C-libraries available.

- Search Method: It should use complete enumeration, with a suffix tree, to allow long motifs, possibly speeding up the search for trinucleotides as seeds that are extended in two directions until a good score is found. Obviously, no kind of spacers are needed for single motifs. The search method should be flexible, overall design should facilitate its change.
- No parameters: It should be possible to get good results with the default parameters. Motif length should be tried between two limits.
- Testing the scoring function: The score can be easily tested on motifs that were obtained from random sequences and on motifs that are directly taken from

Transfac. Any function that separates the two well is a good candidate for a scoring function.

- Score Limit: "Too good" motifs are not very realistic, they rather indicate repeated regions that weren't filtered out. The score should eliminate too well conserved motifs, like DMOTIF. One way to achieve this would be the filtering out motifs whose IC exceeds a certain threshold.
- Negative Score: It should be possible that the algorithm returns negative results, indicating that no good motif has been found, similar to MOPAC. This makes interpretation by biologists easier and improved their confidence on the results. It should return as few motifs as possible on random sequences (the author acknowledges that this looks very difficult at the moment).
- The Conserved Core: As indicated by the motif discovery assessment, the score has to take into account motifs that show some kind of conserved core, i.e. some positions next to each other that stay the same in many sites. Very probable motifs fullfill this condition, as proven by Weeder's good performance. (Sandelin & Wasserman, 2004) (shown with the Gibbs and AnnSpec-implementations) already proved that incorporation of typical transcription factor's profiles (similar to the matrix families from Genomatix' Database) does indeed improve the performance of motif scanning .
- Counting on the Genome: Using the whole set of upstream sequences as the background to calculate p-values makes much sense, as used by the algorithms by Jacques van Helden, oligo-/dyad-analysis. With suffix trees, this can be done very quickly, like in SMILE. It should be possible to score against and possibly see all instances of a found motif on the genome, like in AlignACE or the MotifScanner.
- Group specificity (optional): Some kind of group specifity should be part of a score, similar to ScanACE or dyad-analysis. However, it seems unlikely that doing this for a single binding site makes much sense for mammalian promoters.
- Comparative Genomics (optional): Conservation in close species should receive a high weight and is probably the most important signal, as shown by the recent successful predictions on yeast. Close sequences should be aligned, blocks of 6-9bp-long stretches highlighted. How to put this into a score is difficult to estimate at the moment but should be part of an improved version of the algorithm.
- Performance Validation: With an automated test and validation against the leading programs of the assessment — Weeder, oligoanalysis, Consensus, YMF

and MEME — different scores could be easily tried and the best combination identified. Test sets could be those from the assessment for a start or others, extracted from Transfac. Like in the assessment, the algorithm should also be tried on sets with some negative examples, e.g. sequences that don't contain the motif, as it is usually the case when working on real biological data.

The result would be a motif discovery algorithm that has been designed from the ground up along meaningful biological examples. It could be more useful to biologists than the current implementations and therefore should show a superior performance in the next assessment.

Appendix A.

Learning about promotor analysis

General Reviews The best and most up-to-date review is (Wasserman & Sandelin, 2004). It is a broad overview over most of the general methods and directions of the field, with obvious practical experiences by the author. Another general one and very similar to the latter is (Bulyk, 2003). Bearing a slightly misleading title, (Zhang, 2002) treats roughly the same issues, with more focus on algorithmical details and statistical background. (Hoglund & Kohlbacher, 2004) presents the topic from a more protein-structural-view. Covering just biological topics, (Levine & Tjian, 2003) gives some good real-world examples of promotor structure in various species. (Wray *et al.*, 2003) is a massive 40-pages review with everything about eukaryotic promoters that can be possibly collected at the moment.

Reviews: Motif scanning and discovery A basic and readable article about weight matrices is (Stormo, 2000) or (Stormo, 1990). The problems of consensus sequences are mentioned in all publications by Tom Schneider, "googling" his name should give ample amounts of websites and references. The general problems when dealing with composite eurcarytic promotors are well explained in the somewhat older review (Werner, 1999). Practical results from promotor analysis are presented in (Rombauts *et al.*, 2003) (plants) and (Vanet *et al.*, 1999) (bacteria), where the latter's focus lies on more outdated but easy-to-grasp motif discovery methods.

Hands-on-experience For streamlined tutorials, refer to the RSA-Tools Tutorials section. They include test data and web-interfaces, integrating different tools that can be tried. Having completed one tutorial gives a feeling for the general process of extraction and analysis while not overcomplicating issues, as the examples are well-studied yeast genes.

The company Genomatix `http://www.genomatix.com` also has tutorials that focus on their own software applied to human genes . They are still interesting, since they highlight the search for combinations of multiple factors.

To see hundreds of links for further work and tutorials that explain most pitfalls of the complete analysis on higher eukaryotic species' promotors, Bioinformatics

World (`http://homepage.univie.ac.at/herbert.mayer/` is the one and only site on the web. Permanent updates, experiences for most programs referenced and fast feedback by the author to any questions makes it an invaluable though little-known resource when exploring the full width of promotor analysis tools.

(Frazer *et al.*, 2003) is a practical review that explains how to work with human-mouse comparative analysis tools and what sequences to expect next from the comparative genomics program at the National Institutes of Health.

Appendix B.

Extracting eukaryotic promoters

Motif discovery and scanning alike necessitate the availability of sequences. However, when working on real examples, we soon found out that this task if far from trivial. Our experiences are presented in this excursus, since they are not needed for the discussion of motif discovery algorithms. They might be valuable, though, for readers with a biological background that want to apply promoter analysis in the near future.

The definition of "promoter" is rather vague: A region which is full of binding sites that are needed to express the gene. As long as the binding sites are unknown, it is hard to delineate the promotor region. Therefore, most promoter analyses rather concentrate on "upstream" regions, i.e. , the region 5' of the start of transcription (Transcription Start Site, TSS). As the start of transcription is often not available due

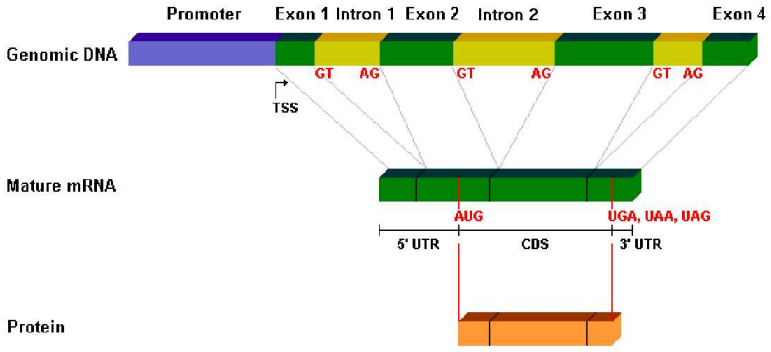


Figure B.1.: Structure of a gene
Legend: TSS = Transcription Start Site, Exon = gene sequence part that is used for coding the protein, Intron = gene sequence part that is removed during splicing and does not code for a protein, UTR=Untranslated Region, CDS=Coding Sequence, AUG and UGA/UAA/UAG typical start/stop-codons of the coding sequence

to the lack of full mRNA-sequences in the databases, we have to rely on the region 5' of the start of translation, which – for the case of yeast, bacteria or plants – is rather close, separated from the TSS only by a usually short "untranslated region" (UTR). The UTR is a section in front of the coding sequence which is not translated. The start of translation is then much easier to determine, as we simply can take the protein sequence for this gene, re-convert the amino acids to nucleic acids, align them onto the genome and search the beginning. These data is all readily available in most genome databases and can be simply viewed on genome browser as a graphic and accessed via special APIs (see Figure B.1, taken from the Genomatix.de-tutorials, for an illustration of all these terms).

However, the UTR can get very long for high eukaryotes, like humans. In addition, evidence suggests that in 40% of human genes, the first exon is non-coding and in 60% the distance to TSS can be up to 1000 bps. (Davuluri *et al.*, 2001) So for more complex species, we can align the protein sequences, but when taking the upstream 2kb-region, will extract mostly Exon 1 and UTR. They might contain binding sites sometimes, but it is not the region we are interested in.

Therefore, for high eukaryotes, one really should predict the promoter by different means. Traditional gene scanners won't be a help here, as they only try to separate coding from non-coding regions. There are various other options but they are all far from being very accurate:

- First Exon Predictions: It seems that first exons can be predicted by similar means as genes, exploiting composition and special features attributed to them in a statistical way, like counting certain words, calculating GC-content and the like. FirstEF can find roughly 86% of the first exons (Davuluri *et al.*, 2001). The problem of the 5'-UTR still persits, however.
- TSS and Promoter Predictions: *Promotor Inspector* uses a heurisic based on combinations of basal binding sites and will find about 50% of human promoters. (Scherf *et al.*, 2000) Similar programs are available that try to predict TSS. *Improbizer* (Scherf *et al.*, 2000) has about the same sensitivity as Promoter Inspector, just like McPromoter (Ohler *et al.*, 2002). The newer *Dragon Gene Start Finder* can improve sensitivity to 65%. (Bajic & Seah, 2003) Using sequences from multiple species, one can increase this figure by some degree, like the *PromH*-program (Solovyev & Shahmuradov, 2003). See (Werner, 2003) for a comparison and review of these and other methods.
- Aligning ESTs, manually: Expressed Sequence Tags (ESTs) are short oligonucleotides that result from experiments where the biologist simply wants to identify a certain RNA-sequence. They are the quick-and-dirty assembly of the

sequences at both ends of a given RNA. ESTs are stored in the usual genome databases in high numbers and are displayed in most genome browsers. By inspecting a gene's upstream region and displaying all ESTs below the sequence, biologists can try to guess where the gene really starts. It will usually be a part upstream of the start of translation, where a couple of similar ESTs align well. Many other ones are often scattered over a long stretch of DNA, since the original RNA is very fragile: broken RNA will lead to an EST that was not located at the end of the RNA but somewhere in the middle of it. A biologist's intuition is needed to find out, which ESTs are valuable and which should rather be ignored. So this method is time-intensive and requires experience and luck.

- Aligning ESTs, automatic: The whole process can of course be automated. Special programs exist that extract all necessary data from Genbank, align all ESTs and try to find a position which is either most 5' of many ESTs or which regroups the highest number of them close to it. They then return the upstream part of the predicted position up to a certain length. However, luck is still part of the game and the fact that many ESTs align well at a certain position can also mean that the RNA tends to break at certain places that are more fragile than the surrounding parts. It does not necessarily mean that the TSS is more likely to start here than at some other place.
  Tools in this field either align EST-data on the human genome, like FIE2 (Chong *et al.*, 2003) and Promoser (Halees *et al.*, 2003), or on genbank-records, like PEG (Zhang & Zhang, 2001).
- Annotated TSS: The RefSeq list of genes (Pruitt & Maglott, 2001) is a curated database that lists a "standard"-sequence for most human genes of interest. By collecting evidence from literature, the RefSeq-team tries to annotate genes and therefore also their start, the TSS. All upstream sequences from all human RefSeq genes are available as one big file, updated every few months, from the UCSC genome browser website. Another option is Ensmart (Kasprzyk *et al.*, 2004) , which extracts batches of sequences, tailored to the users needs, or EZ-Retrieve (Zhang, 2002). As can be seen by the next paragraph though, RefSeq is not very accurate in many cases. The company Genomatix (`http://www.genomatix.de`) combines RefSeq, PromoterInspector- or proprietary-comparative predictions and aligned ESTs for its Promoter-Database, to improve accuracy.
- Full mRNA Database: DBTSS (Suzuki *et al.*, 2002) (Suzuki *et al.*, 2001) is a database that stores mRNA sequences obtained by a new method which is supposed to be more reliable than its predecessors. In addition, being conducted

centrally and almost in the same way, the experiments are more comparable than ESTs, originated from all around the world. DBTSS is very likely the most reliable way to determine a TSS: 34% of RefSeq's entries could be corrected by it. However, at a size of 9000 records, DBTSS is far from covering all 25.000 human genes, so other methods will still be used for a while. We don't know how much of DBTSS's corrections have already been incorporated into the public databases at the moment. MGC is also a database that contains full length mRNA sequences but seems to be slighly smaller and, to the best of our knowledge, does not offer to download upstream regions directly. (Strausberg *et al.*, 2002)

- Promoter Databases: If one is very lucky, the respective promoter has already been analyzed and verified with experiments. The Eukaryotic Promoter Database (Périer *et al.*, 1998) comprises some 2500 human and 200 plant promoters collected from literature and – since very recently (Schmid *et al.*, 2004) – also derived from MCG/DBTSS. PRESTA (Mach, 2002) extends older promoters from genbank with ESTs. All these sources should be consulted, as their results do not necessarily overlap and are of similar quality. (Schmid *et al.*, 2003)

The whole process gets more complicated by the fact that many genes are supposed to have multiple possible TSS and promoters. Genomatix, e.g., claims that 28% of human genes have alternative promotors (Genomatix, 2004). The many different ESTs, visible by small lines in the genome browsers, might reflect this situation better than thought before. However, working on this *in silico* on a large-scale is rather new; there are no benchmarks and little experience with it. We therefore have the impression that the prospects for applying motif discovery successfully to human or mouse sequences look dim at the moment, but the recent and ongoing improvements of promoter databases might change this very soon, though mostly for human and mouse genes. Not as much for the fruitfly, where the author assumes that recent sequencing efforts will soon pave the way for comparative promotor prediction.

Bibliography

[Adebiyi & Kaufmann, 2002] Adebiyi, E. F. & Kaufmann, M. (2002) Extracting common motifs under the levenshtein measure: theory and experimentation. In *Proceedings of the Second International Workshop on Algorithms in Bioinformatics* pp. 140–156 Springer-Verlag.

[Aerts *et al.*, 2003] Aerts, S., Thijs, G., Coessens, B., Staes, M., Moreau, Y. & Moor, B. D. (2003) Toucan: deciphering the cis-regulatory logic of coregulated genes. *Nucleic Acids Res,* **31** (6), 1753–64.

[Alberts *et al.*, 1994] Alberts, B., Bray, D., Lewis, J., Raff, M., Roberts, K. & Watson, J. (1994) *Molecular Biology of the Cell*. 3. edition,, Garland Science.

[Alkema *et al.*, 2004] Alkema, W. B. L., Johansson, O., Lagergren, J. & Wasserman, W. W. (2004) MSCAN: identification of functional clusters of transcription factor binding sites. *Nucleic Acids Res,* **32** (Web Server issue), W195–8.

[Anderson & Parker, 2000] Anderson, J. J. & Parker, R. (2000) Computational identification of cis-acting elements affecting post-transcriptional control of gene expression in Saccharomyces cerevisiae. *Nucleic Acids Res,* **28** (7), 1604–17.

[Apostolico *et al.*, 2000] Apostolico, A., Bock, M., Lonardi, S. & Xu, X. (2000) Efficient detection of unusual words. *J Comput Biol,* **7** (1-2), 71–94.

[Apostolico *et al.*, 2003] Apostolico, A., Gong, F. & Lonardi, S. (2003) Verbumculus and the Discovery of Unusual Words. *Journal of Computer Science and Technology,* **19** (1), 22–41.

[Bailey & Elkan, 1994] Bailey, T. & Elkan, C. (1994) Fitting a mixture model by expectation maximization to discover motifs in biopolymers. *Proc Int Conf Intell Syst Mol Biol,* **2**, 28–36.

[Bailey & Gribskov, 1998] Bailey, T. & Gribskov, M. (1998) Methods and statistics for combining motif match scores. *J Comput Biol,* **5** (2), 211–21.

[Bailey, 1995] Bailey, T. L. (1995). *Discovering motifs in DNA and protein sequences: The approximate common substring problem*. PhD thesis, University of California, San Diego.

[Bailey & Noble, 2003] Bailey, T. L. & Noble, W. S. (2003) Searching for statistically significant regulatory modules. *Bioinformatics,* **19 Suppl 2**, II16–II25.

[Bajic & Seah, 2003] Bajic, V. B. & Seah, S. H. (2003) Dragon Gene Start Finder identifies approximate locations of the 5' ends of genes. *Nucleic Acids Res,* **31** (13), 3560–3.

[Bannai *et al.*, 2004] Bannai, H., Inenaga, S., Shinohara, A., Takeda, M. & Miyano, S. (2004) Efficiently finding regulatory elements using correlation with gene expression. *J Bioinform Comput Biol,* **2** (2), 273–88.

[Barash *et al.*, 2003] Barash, Y., Elidan, G., Friedman, N. & Kaplan, T. (2003) Modeling dependencies in protein-dna binding sites. In *Proceedings of the seventh annual international conference on Computational molecular biology* pp. 28–37 ACM Press.

[Basehoar *et al.*, 2004] Basehoar, A. D., Zanton, S. J. & Pugh, B. F. (2004) Identification and distinct regulation of yeast TATA box-containing genes. *Cell,* **116** (5), 699–709.

[Batzoglou *et al.*, 2000] Batzoglou, S., Pachter, L., Mesirov, J., Berger, B. & Lander, E. (2000) Human and mouse gene structure: comparative analysis and application to exon prediction. *Genome Res,* **10** (7), 950–8.

[Baxevanis, 2002] Baxevanis, A. (2002) *Current Protocols in Bioinformatics, Chapter 5*. Wiley.

[Benos *et al.*, 2002] Benos, P. V., Bulyk, M. L. & Stormo, G. D. (2002) Additivity in protein-DNA interactions: how good an approximation is it? *Nucleic Acids Res,* **30** (20), 4442–51.

[Berezikov *et al.*, 2004] Berezikov, E., Guryev, V., Plasterk, R. H. A. & Cuppen, E. (2004) CONREAL: conserved regulatory elements anchored alignment algorithm for identification of transcription factor binding sites by phylogenetic footprinting. *Genome Res,* **14** (1), 170–8.

[Bergman *et al.*, 2002] Bergman, C. M., Pfeiffer, B. D., n Limas, D. E., Hoskins, R. A., Gnirke, A., Mungall, C. J., Wang, A. M., Kronmiller, B., Pacleb, J., Park, S., Stapleton, M., Wan, K., George, R. A., de Jong, P. J., Botas, J., Rubin, G. M. & Celniker, S. E. (2002) Assessing the impact of comparative genomic sequence data on the functional annotation of the Drosophila genome. *Genome Biol,* **3** (12), RESEARCH0086.

[Berman *et al.*, 2002] Berman, B. P., Nibu, Y., Pfeiffer, B. D., Tomancak, P., Celniker, S. E., Levine, M., Rubin, G. M. & Eisen, M. B. (2002) Exploiting transcription factor binding site clustering to identify cis-regulatory modules involved in pattern formation in the Drosophila genome. *Proc Natl Acad Sci U S A,* **99** (2), 757–62.

[Berman *et al.*, 2004] Berman, B. P., Pfeiffer, B. D., Laverty, T. R., Salzberg, S. L., Rubin, G. M., Eisen, M. B. & Celniker, S. E. (2004) Computational identification of developmental enhancers: conservation and function of transcription factor binding-site clusters in Drosophila melanogaster and Drosophila pseudoobscura. *Genome Biol,* **5** (9), R61.

[Bi & Rogan, 2004] Bi, C. & Rogan, P. K. (2004) Bipartite pattern discovery by entropy minimization-based multiple local alignment. *Nucleic Acids Res,* **32** (17), 4979–91.

[Birnbaum *et al.*, 2001] Birnbaum, K., Benfey, P. & Shasha, D. (2001) cis element/transcription factor analysis (cis/TF): a method for discovering transcription factor/cis element relationships. *Genome Res,* **11** (9), 1567–73.

[Blanchette *et al.*, 2003] Blanchette, M., Kwong, S. & Tompa, M. (2003) An empirical comparison of tools for phylogenetic footprinting. In *Third IEEE Symposium on Bioinformatics and Bioengineering* p. pp. 6978 IEEE Computer Society, Los Alamitos, CA.

[Blanchette *et al.*, 2000] Blanchette, M., Schwikowski, B. & Tompa, M. (2000) An exact algorithm to identify motifs in orthologous sequences from multiple species. *Proc Int Conf Intell Syst Mol Biol,* **8**, 37–45.

[Blanchette & Sinha, 2001] Blanchette, M. & Sinha, S. (2001) Separating real motifs from their artifacts. *Bioinformatics,* **17 Suppl 1**, S30–8.

[Blanchette & Tompa, 2002] Blanchette, M. & Tompa, M. (2002) Discovery of regulatory elements by a computational method for phylogenetic footprinting. *Genome Res,* **12** (5), 739–48.

[Blanchette & Tompa, 2003] Blanchette, M. & Tompa, M. (2003) FootPrinter: A program designed for phylogenetic footprinting. *Nucleic Acids Res,* **31** (13), 3840–2.

[Boffelli *et al.*, 2003] Boffelli, D., McAuliffe, J., Ovcharenko, D., Lewis, K. D., Ovcharenko, I., Pachter, L. & Rubin, E. M. (2003) Phylogenetic shadowing of primate sequences to find functional regions of the human genome. *Science,* **299** (5611), 1391–4.

[Brazma *et al.*, 1998*a*] Brazma, A., Jonassen, I., Eidhammer, I. & Gilbert, D. (1998*a*) Approaches to the automatic discovery of patterns in biosequences. *J Comput Biol,* **5** (2), 279–305.

[Brazma *et al.*, 1998*b*] Brazma, A., Jonassen, I., Vilo, J. & Ukkonen, E. (1998*b*) Predicting gene regulatory elements in silico on a genomic scale. *Genome Res,* **8** (11), 1202–15.

[Brejova *et al.*, 2000] Brejova, B., Di Marco, C., Vinar, T., SR., H., Holguin, G. & Patten, C. Finding patterns in biological sequences, project report, university of waterloo. Brejova2000.

[Brudno *et al.*, 2003] Brudno, M., Do, C. B., Cooper, G. M., Kim, M. F., Davydov, E., Green, E. D., Sidow, A., Batzoglou, S. & Batzoglou, N. C. S. P. (2003) LAGAN and Multi-LAGAN: efficient tools for large-scale multiple alignment of genomic DNA. *Genome Res,* **13** (4), 721–31.

[Buhler & Tompa, 2002] Buhler, J. & Tompa, M. (2002) Finding motifs using random projections. *J Comput Biol,* **9** (2), 225–42.

[Bujnicki *et al.*, 2001] Bujnicki, J., Elofsson, A., Fischer, D. & Rychlewski, L. (2001) LiveBench-2: large-scale automated evaluation of protein structure prediction servers. *Proteins,* **Suppl 5**, 184–91.

[Bulyk *et al.*, 2001] Bulyk, M., Huang, X., Choo, Y. & Church, G. (2001) Exploring the DNA-binding specificities of zinc fingers with DNA microarrays. *Proc Natl Acad Sci U S A,* **98** (13), 7158–63.

[Bulyk, 2003] Bulyk, M. L. (2003) Computational prediction of transcription-factor binding site locations. *Genome Biol,* **5** (1), 201.

[Bussemaker *et al.*, 2000*a*] Bussemaker, H., Li, H. & Siggia, E. (2000*a*) Regulatory element detection using a probabilistic segmentation model. *Proc Int Conf Intell Syst Mol Biol,* **8**, 67–74.

[Bussemaker *et al.*, 2000*b*] Bussemaker, H., Li, H. & Siggia, E. (2000*b*) Building a dictionary for genomes: identification of presumptive regulatory sites by statistical analysis. *Proc Natl Acad Sci U S A,* **97** (18), 10096–100.

[Bussemaker *et al.*, 2001] Bussemaker, H., Li, H. & Siggia, E. (2001) Regulatory element detection using correlation with expression. *Nat Genet,* **27** (2), 167–71.

[Chiang *et al.*, 2003] Chiang, D. Y., Moses, A. M., Kellis, M., Lander, E. S. & Eisen, M. B. (2003) Phylogenetically and spatially conserved word pairs associated with gene-expression changes in yeasts. *Genome Biol,* **4** (7), R43.

[Chong *et al.*, 2003] Chong, A., Zhang, G. & Bajic, V. B. (2003) FIE2: A program for the extraction of genomic DNA sequences around the start and translation initiation site of human genes. *Nucleic Acids Res,* **31** (13), 3546–53.

[Cliften *et al.*, 2003] Cliften, P., Sudarsanam, P., Desikan, A., Fulton, L., Fulton, B., Majors, J., Waterston, R., Cohen, B. A. & Johnston, M. (2003) Finding functional features in Saccharomyces genomes by phylogenetic footprinting. *Science,* **301** (5629), 71–6.

[Coessens *et al.*, 2003] Coessens, B., Thijs, G., Aerts, S., Marchal, K., Smet, F. D., Engelen, K., Glenisson, P., Moreau, Y., Mathys, J. & Moor, B. D. (2003) INCLUSive: A web portal and service registry for microarray and regulatory sequence analysis. *Nucleic Acids Res,* **31** (13), 3468–70.

[Conlon *et al.*, 2003] Conlon, E. M., Liu, X. S., Lieb, J. D. & Liu, J. S. (2003) Integrating regulatory motif discovery and genome-wide expression analysis. *Proc Natl Acad Sci U S A,* **100** (6), 3339–44.

[Danilova *et al.*, 2003] Danilova, L. V., Lyubetsky, V. A. & Gelfand, M. S. (2003) An algorithm for identification of regulatory signals in unaligned DNA sequences, its testing and parallel implementation. *In Silico Biol,* **3** (1-2), 33–47.

[Davidson, 2003] Davidson, M. (2003). An easy architecture for managing actions. http://www.javadesktop.org/articles/actions/.

[Davuluri *et al.*, 2001] Davuluri, R., Grosse, I. & Zhang, M. (2001) Computational identification of promoters and first exons in the human genome. *Nat Genet,* **29** (4), 412–7.

[Davuluri *et al.*, 2003] Davuluri, R. V., Sun, H., Palaniswamy, S. K., Matthews, N., Molina, C., Kurtz, M. & Grotewold, E. (2003) AGRIS: Arabidopsis gene regulatory information server, an information resource of Arabidopsis cis-regulatory elements and transcription factors. *BMC Bioinformatics,* **4** (1), 25.

[Dieterich *et al.*, 2003] Dieterich, C., Wang, H., Rateitschak, K., Luz, H. & Vingron, M. (2003) CORG: a database for COmparative Regulatory Genomics. *Nucleic Acids Res,* **31** (1), 55–7.

[Djordjevic *et al.*, 2003] Djordjevic, M., Sengupta, A. M. & Shraiman, B. I. (2003) A biophysical approach to transcription factor binding site discovery. *Genome Res,* **13** (11), 2381–90.

[Doug, 1999] Doug, L. (1999) *Concurrent Programming in Java: Design Principles and Patterns*. 2. edition,, Addison Wessley.

[Down & Hubbard, 2005] Down, T. A. & Hubbard, T. J. P. (2005) NestedMICA: sensitive inference of over-represented motifs in nucleic acid sequence. *Nucleic Acids Res,* **33** (5), 1445–53.

[Durbin & Haussler, Year] Durbin, R. & Haussler, D. (UnknownYear). GFF: an Exchange Format for Feature Description. http://www.sanger.ac.uk/Software/formats/GFF/.

[Edwards *et al.*, 2003] Edwards, Y. J. K., Carver, T. J., Vavouri, T., Frith, M., Bishop, M. J. & Elgar, G. (2003) Theatre: A software tool for detailed comparative analysis and visualization of genomic sequence. *Nucleic Acids Res,* **31** (13), 3510–7.

[Elkon *et al.*, 2003] Elkon, R., Linhart, C., Sharan, R., Shamir, R. & Shiloh, Y. (2003) Genome-wide in silico identification of transcriptional regulators controlling the cell cycle in human cells. *Genome Res,* **13** (5), 773–80.

[Eskin, 2004] Eskin, E. (2004) From profiles to patterns and back again: a branch and bound algorithm for finding near optimal motif profiles. In *Proceedings of the eighth annual international conference on Computational molecular biology* pp. 115–124 ACM Press.

[Eskin & Pevzner, 2002] Eskin, E. & Pevzner, P. A. (2002) Finding composite regulatory patterns in DNA sequences. *Bioinformatics,* **18 Suppl 1**, S354–63.

[Favorov *et al.*, 2002] Favorov, A., Gelfand, M., Mironov, A. & Makeev, V. (2002) Yet another digging for dna motifs gibbs sampler. In *Proceedings of the Third International Conference on Bioinformatics of Genome Regulation and Structure, BGRS 2002, Novosibirsk, Russia* vol. 1, pp. 31–33.

[Felsenfeld & Groudine, 2003] Felsenfeld, G. & Groudine, M. (2003) Controlling the double helix. *Nature,* **421** (6921), 448–53.

[Fischer & Rychlewski, 2003] Fischer, D. & Rychlewski, L. (2003) The 2002 Olympic Games of protein structure prediction. *Protein Eng,* **16** (3), 157–60.

[Fogel *et al.*, 2004] Fogel, G. B., Weekes, D. G., Varga, G., Dow, E. R., Harlow, H. B., Onyia, J. E. & Su, C. (2004) Discovery of sequence motifs related to coexpression of genes using evolutionary computation. *Nucleic Acids Res,* **32** (13), 3826–35.

[Forget *et al.*, 1997] Forget, D., Robert, F., Grondin, G., Burton, Z., Greenblatt, J. & Coulombe, B. (1997) RAP74 induces promoter contacts by RNA polymerase II upstream and downstream of a DNA bend centered on the TATA box. *Proc Natl Acad Sci U S A,* **94** (14), 7150–5.

[Frazer *et al.*, 2003] Frazer, K. A., Elnitski, L., Church, D. M., Dubchak, I. & Hardison, R. C. (2003) Cross-species sequence comparisons: a review of methods and available resources. *Genome Res,* **13** (1), 1–12.

[Frazer *et al.*, 2004] Frazer, K. A., Tao, H., Osoegawa, K., de Jong, P. J., Chen, X., Doherty, M. F. & Cox, D. R. (2004) Noncoding sequences conserved in a limited number of mammals in the SIM2 interval are frequently functional. *Genome Res,* **14** (3), 367–72.

[Frith *et al.*, 2004] Frith, M., Hansen, U., Spouge, J. & Weng, Z. (2004) Finding functional sequence elements by multiple local alignment. *Nucleic Acids Res,* **32** (1), 189–200.

[Frith *et al.*, 2001] Frith, M., Hansen, U. & Weng, Z. (2001) Detection of cis-element clusters in higher eukaryotic DNA. *Bioinformatics,* **17** (10), 878–89.

[Frith *et al.*, 2004] Frith, M. C., Fu, Y., Yu, L., Chen, J.-F., Hansen, U. & Weng, Z. (2004) Detection of functional DNA motifs via statistical over-representation. *Nucleic Acids Res,* **32** (4), 1372–81.

[Frith *et al.*, 2002] Frith, M. C., Spouge, J. L., Hansen, U. & Weng, Z. (2002) Statistical significance of clusters of motifs represented by position specific scoring matrices in nucleotide sequences. *Nucleic Acids Res,* **30** (14), 3214–24.

[Fu *et al.*, 2004] Fu, Y., Frith, M. C., Haverty, P. M. & Weng, Z. (2004) MotifViz: an analysis and visualization tool for motif discovery. *Nucleic Acids Res,* **32** (Web Server issue), W420–3.

[Fukue *et al.*, 2004] Fukue, Y., Sumida, N., ichi Nishikawa, J. & Ohyama, T. (2004) Core promoter elements of eukaryotic genes have a highly distinctive mechanical property. *Nucleic Acids Res,* **32** (19), 5834–40.

[Galas *et al.*, 1985] Galas, D., Eggert, M. & Waterman, M. (1985) Rigorous pattern-recognition methods for DNA sequences. Analysis of promoter sequences from Escherichia coli. *J Mol Biol,* **186** (1), 117–28.

[Ganesh *et al.*, 2003] Ganesh, R., Siegele, D. A. & Ioerger, T. R. (2003) MOPAC: motif finding by preprocessing and agglomerative clustering from microarrays. *Pac Symp Biocomput,* , 41–52.

[Gelfand *et al.*, 2000] Gelfand, M., Koonin, E. & Mironov, A. (2000) Prediction of transcription regulatory sites in Archaea by a comparative genomic approach. *Nucleic Acids Res,* **28** (3), 695–705.

[Genomatix, 2004] Genomatix (2004). Demands on Microarray Design for the Evaluation of Gene Regulation. Whitepaper http://www.genomatix.de/download/documents/AltPromoterMicroarrays.pdf.

[Georgi, 2002] Georgi, B. (2002). Der expectation-maximization algorithmus, slides for the seminar "markovketten"at freie universitat berlin, winter 2002. http://biocomputing.mi.fu-berlin.de/Lehre/MarkovKetten%5FWS02/seminar/Vortraege/Georgi.ppt.

[Ghosh, 1998] Ghosh, D. (1998) OOTFD (Object-Oriented Transcription Factors Database): an object-oriented successor to TFD. *Nucleic Acids Res,* **26** (1), 360–2.

[Giegerich & Kurtz, 1995] Giegerich, R. & Kurtz, S. (1995) A comparison of imperative and purely functional suffix tree constructions. *Science of Computer Programming,* **25** (2-3), 187–218.

[Grad *et al.*, 2004] Grad, Y. H., Roth, F. P., Halfon, M. S. & Church, G. M. (2004) Prediction of similarly-acting cis-regulatory modules by subsequence profiling and comparative genomics in D. melanogaster and D. pseudoobscura. *Bioinformatics,* **20** (16), 2738–50.

[Grundy *et al.*, 1997] Grundy, W., Bailey, T., Elkan, C. & Baker, M. (1997) Meta-MEME: motif-based hidden Markov models of protein families. *Comput Appl Biosci,* **13** (4), 397–406.

[GuhaThakurta *et al.*, 2002] GuhaThakurta, D., Palomar, L., Stormo, G. D., Tedesco, P., Johnson, T. E., Walker, D. W., Lithgow, G., Kim, S. & Link, C. D. (2002) Identification of a novel cis-regulatory element involved in the heat shock response in Caenorhabditis elegans using microarray gene expression and computational methods. *Genome Res,* **12** (5), 701–12.

[GuhaThakurta & Stormo, 2001] GuhaThakurta, D. & Stormo, G. (2001) Identifying target sites for cooperatively binding factors. *Bioinformatics,* **17** (7), 608–21.

[Gupta & Liu, 2003] Gupta, M. & Liu, J. S. (2003) Discovery of conserved sequence patterns using a stochastic dictionary model. *Journal of the American Statistical Association,* **98** (461), 55–66.

[Halees *et al.*, 2003] Halees, A. S., Leyfer, D. & Weng, Z. (2003) PromoSer: A large-scale mammalian promoter and transcription start site identification service. *Nucleic Acids Res,* **31** (13), 3554–9.

[Halfon *et al.*, 2002] Halfon, M. S., Grad, Y., Church, G. M. & Michelson, A. M. (2002) Computation-based discovery of related transcriptional regulatory modules and motifs using an experimentally validated combinatorial model. *Genome Res,* **12** (7), 1019–28.

[Harbison *et al.*, 2004] Harbison, C. T., Gordon, D. B., Lee, T. I., Rinaldi, N. J., Macisaac, K. D., Danford, T. W., Hannett, N. M., Tagne, J.-B., Reynolds, D. B., Yoo, J., Jennings, E. G., Zeitlinger, J., Pokholok, D. K., Kellis, M., Rolfe, P. A., Takusagawa, K. T., Lander, E. S., Gifford, D. K., Fraenkel, E. & Young, R. A. (2004) Transcriptional regulatory code of a eukaryotic genome. *Nature,* **431** (7004), 99–104.

[Helden *et al.*, 1998] Helden, J. v., Andre, B. & Collado-Vides, J. (1998) Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. *J Mol Biol,* **281** (5), 827–42.

[Hengen *et al.*, 2003] Hengen, P. N., Lyakhov, I. G., Stewart, L. E. & Schneider, T. D. (2003) Molecular flip-flops formed by overlapping Fis sites. *Nucleic Acids Res,* **31** (22), 6663–73.

[Hernandez *et al.*, 2004] Hernandez, D., Gras, R. & Appel, R. (2004) MoDEL: an efficient strategy for ungapped local multiple alignment. *Comput Biol Chem,* **28** (2), 119–28.

[Hertz & Stormo, 1999] Hertz, G. & Stormo, G. (1999) Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics,* **15** (7-8), 563–77.

[Higo *et al.*, 1998] Higo, K., Ugawa, Y., Iwamoto, M. & Higo, H. (1998) PLACE: a database of plant cis-acting regulatory DNA elements. *Nucleic Acids Res,* **26** (1), 358–9.

[Hoglund & Kohlbacher, 2004] Hoglund, A. & Kohlbacher, O. (2004) From sequence to structure and back again: approaches for predicting protein-DNA binding. *Proteome Sci,* **2** (1), 3.

[Holmes & Bruno, 2000] Holmes, I. & Bruno, W. (2000) Finding regulatory elements using joint likelihoods for sequence and expression profile data. *Proc Int Conf Intell Syst Mol Biol,* **8**, 202–10.

[Horton, 2001] Horton, P. (2001) Tsukuba BB: a branch and bound algorithm for local multiple alignment of DNA and protein sequences. *J Comput Biol,* **8** (3), 283–303.

[Hu, 2003] Hu, Y.-J. (2003) Finding subtle motifs with variable gaps in unaligned DNA sequences. *Comput Methods Programs Biomed,* **70** (1), 11–20.

[Huang *et al.*, 2004] Huang, H.-D., Horng, J.-T., Sun, Y.-M., Tsou, A.-P. & Huang, S.-L. (2004) Identifying transcriptional regulatory sites in the human genome using an integrated system. *Nucleic Acids Res,* **32** (6), 1948–56.

[Hudak & Mcclure, 1999] Hudak, J. & Mcclure, M. (1999) A comparative analysis of computational motif-detection methods. *Pac Symp Biocomput,* **4**, 138–49.

[Hughes *et al.*, 2000] Hughes, J., Estep, P., Tavazoie, S. & Church, G. (2000) Computational identification of cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae. *J Mol Biol,* **296** (5), 1205–14.

[Jegga *et al.*, 2002] Jegga, A. G., Sherwood, S. P., Carman, J. W., Pinski, A. T., Phillips, J. L., Pestian, J. P. & Aronow, B. J. (2002) Detection and visualization of compositionally similar cis-regulatory element clusters in orthologous and coordinately controlled genes. *Genome Res,* **12** (9), 1408–17.

[Jensen & Liu, 2004] Jensen, S. T. & Liu, J. S. (2004) BioOptimizer: a Bayesian scoring function approach to motif discovery. *Bioinformatics,* **20** (10), 1557–64.

[Johansson *et al.*, 2003] Johansson, O., Alkema, W., Wasserman, W. & Lagergren, J. (2003) Identification of functional clusters of transcription factor binding motifs in genome sequences: the MSCAN algorithm. *Bioinformatics,* **19 Suppl 1**, i169–76.

[Kamvysselis *et al.*, 2003] Kamvysselis, M., Patterso, N., Birren, B., Berger, B. & Lander, E. (2003) Whole-genome comparative annotation and regulatory motif discovery in multiple yeast species. In *Proceedings of the seventh annual international conference on Computational molecular biology* pp. 157–166 ACM Press.

[Kankainen & Holm, 2004] Kankainen, M. & Holm, L. (2004) POBO, transcription factor binding site verification with bootstrapping. *Nucleic Acids Res,* **32** (Web Server issue), W222–9.

[Karanam & Moreno, 2004] Karanam, S. & Moreno, C. S. (2004) CONFAC: automated application of comparative genomic promoter analysis to DNA microarray datasets. *Nucleic Acids Res,* **32** (Web Server issue), W475–84.

[Kasprzyk *et al.*, 2004] Kasprzyk, A., Keefe, D., Smedley, D., London, D., Spooner, W., Melsopp, C., Hammond, M., Rocca-Serra, P., Cox, T. & Birney, E. (2004) EnsMart: a generic system for fast and flexible access to biological data. *Genome Res,* **14** (1), 160–9.

[Keich & Pevzner, 2002*a*] Keich, U. & Pevzner, P. (2002*a*) Finding motifs in the twilight zone. *Bioinformatics,* **18** (10), 1374–81.

[Keich & Pevzner, 2002*b*] Keich, U. & Pevzner, P. (2002*b*) Subtle motifs: defining the limits of motif finding algorithms. *Bioinformatics,* **18** (10), 1382–90.

[Kel *et al.*, 2003] Kel, A., ssling, E., Reuter, I., Cheremushkin, E., Kel-Margoulis, O. & Wingender, E. (2003) MATCH: A tool for searching transcription factor binding sites in DNA sequences. *Nucleic Acids Res,* **31** (13), 3576–9.

[Kel-Margoulis *et al.*, 2000] Kel-Margoulis, O., Romashchenko, A., Kolchanov, N., Wingender, E. & Kel, A. (2000) COMPEL: a database on composite regulatory elements providing combinatorial transcriptional regulation. *Nucleic Acids Res,* **28** (1), 311–5.

[Kellis *et al.*, 2004] Kellis, M., Patterson, N., Birren, B., Berger, B. & Lander, E. S. (2004) Methods in comparative genomics: genome correspondence, gene identification and regulatory motif discovery. *J Comput Biol,* **11** (2-3), 319–55.

[Kellis *et al.*, 2003] Kellis, M., Patterson, N., Endrizzi, M., Birren, B. & Lander, E. S. (2003) Sequencing and comparison of yeast species to identify genes and regulatory elements. *Nature,* **423** (6937), 241–54.

[Kielbasa *et al.*, 2004] Kielbasa, S., Blüthgen, N., Sers, C., Schäfer, R. & H, H. (2004) Prediction of cis-regulatory elements of coregulated genes. In *Fourth International Workshop on Bioinformatics and Systems Biology*, (Mamitsuka, H., Smith, T., Holzhütter, H.-G., Kanehisa, M., DeLisi, C., Heinrich, R. & Miyano, S., eds), vol. 15, of *Genome Informatics* pp. 117–124 Japanese Society for Bioinformatics Universial Academy Press, Tokyo.

[Kielbasa *et al.*, 2001] Kielbasa, S., Korbel, J., Beule, D., Schuchhardt, J. & Herzel, H. (2001) Combining frequency and positional information to predict transcription factor binding sites. *Bioinformatics,* **17** (11), 1019–26.

[King & Roth, 2003] King, O. D. & Roth, F. P. (2003) A non-parametric model for transcription factor binding sites. *Nucleic Acids Res,* **31** (19), e116.

[Koerkamp *et al.*, 2002] Koerkamp, M. G., Rep, M., Bussemaker, H. J., Hardy, G. P. M. A., Mul, A., Piekarska, K., Szigyarto, C. A.-K., Mattos, J. M. T. D. & Tabak, H. F. (2002) Dissection of transient oxidative stress response in Saccharomyces cerevisiae by using DNA microarrays. *Mol Biol Cell,* **13** (8), 2783–94.

[Kolchanov *et al.*, 2002] Kolchanov, N., Ignatieva, E., Ananko, E., Podkolodnaya, O., Stepanenko, I., Merkulova, T., Pozdnyakov, M., Podkolodny, N., Naumochkin, A. & Romashchenko, A. (2002) Transcription Regulatory Regions Database (TRRD): its status in 2002. *Nucleic Acids Res,* **30** (1), 312–7.

[Lawrence *et al.*, 1993] Lawrence, C., Altschul, S., Boguski, M., Liu, J., Neuwald, A. & Wootton, J. (1993) Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science,* **262** (5131), 208–14.

[Lenhard *et al.*, 2003] Lenhard, B., Sandelin, A., Mendoza, L., r m, Jareborg, N. & Wasserman, W. W. (2003) Identification of conserved regulatory elements by comparative genome analysis. *J Biol,* **2** (2), 13.

[Lenhard & Wasserman, 2002] Lenhard, B. & Wasserman, W. W. (2002) TFBS: Computational framework for transcription factor binding site analysis. *Bioinformatics,* **18** (8), 1135–6.

[Lescot, 2002] Lescot, M. (2002). *Analyse bioinformatique des regions regulatrices d'arabidopsis thaliana et de drosophila melanogaster*. PhD thesis, Faculte des Sciences de Luminy.

[Lescot *et al.*, 2002] Lescot, M., Dehais, P., Thijs, G., Marchal, K., Moreau, Y., Peer, Y. V. d., Rouze, P. & Rombauts, S. (2002) PlantCARE, a database of plant cis-acting regulatory elements and a portal to tools for in silico analysis of promoter sequences. *Nucleic Acids Res,* **30** (1), 325–7.

[Levine & Tjian, 2003] Levine, M. & Tjian, R. (2003) Transcription regulation and animal diversity. *Nature,* **424** (6945), 147–51.

[Lewis *et al.*, 2002] Lewis, S., Searle, S. M. J., Harris, N., Gibson, M., Lyer, V., Richter, J., Wiel, C., Bayraktaroglir, L., Birney, E., Crosby, M., Kaminker, J., Matthews, B., Prochnik, S., Smithy, C., Tupy, J., Rubin, G., Misra, S., Mungall, C. & Clamp, M. (2002) Apollo: a sequence annotation editor. *Genome Biol,* **3** (12), RESEARCH0082.

[Li *et al.*, 2002] Li, M., Ma, B. & Wang, L. (2002) Finding similar regions in many sequences. *J. Comput. Syst. Sci.,* **65** (1), 73–96.

[Liang *et al.*, 2004] Liang, S., Samanta, M. & Biegel, B. (2004) cWINNOWER algorithm for finding fuzzy dna motifs. *J Bioinform Comput Biol,* **2** (1), 47–60.

[Liu *et al.*, 2004] Liu, F. F. M., Tsai, J. J. P., Chen, R.-M., Chen, S. N. & Shih, S. H. (2004) Fmga: finding motifs by genetic algorithm. In *4th IEEE International Symposium on BioInformatics and BioEngineering (BIBE 2004), 19-21 March 2004, Taichung, Taiwan* pp. 459–466.

[Liu, 1994] Liu, J. S. (1994) The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. *Journal of the American Statistical Association,* **29** (427), 958–967.

[Liu, 1995] Liu, J. S. (1995) Bayesian models for multiple local sequence alignment and gibbs sampling strategies. *J. Amer. Statist. Assoc.,* **90** (432), 1156–1170.

[Liu, 2001] Liu, J. S. (2001) *Monte Carlo Strategies in Scientific Computing*. Springer Series in Statistics, 1. edition,, Springer Verlag.

[Liu *et al.*, 2001] Liu, X., Brutlag, D. & Liu, J. (2001) BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pac Symp Biocomput,* **7**, 127–38.

[Liu *et al.*, 2002] Liu, X. S., Brutlag, D. L. & Liu, J. S. (2002) An algorithm for finding protein-DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nat Biotechnol,* **20** (8), 835–9.

[Liu *et al.*, 2004*a*] Liu, Y., Liu, X. S., Wei, L., Altman, R. B. & Batzoglou, S. (2004*a*) Eukaryotic regulatory element conservation analysis and identification using comparative genomics. *Genome Res,* **14** (3), 451–8.

[Liu *et al.*, 2004*b*] Liu, Y., Wei, L., Batzoglou, S., Brutlag, D. L., Liu, J. S. & Liu, X. S. (2004*b*) A suite of web-based programs to search for transcriptional regulatory motifs. *Nucleic Acids Res,* **32** (Web Server issue), W204–7.

[Lockwood *et al.*, 2003] Lockwood, C. R., Bingham, C. & Frayling, T. M. (2003) In silico searching of human and mouse genome data identifies known and unknown HNF1 binding sites upstream of beta-cell genes. *Mol Genet Metab,* **78** (2), 145–51.

[Lockwood & Frayling, 2003] Lockwood, C. R. & Frayling, T. M. (2003) Combining genome and mouse knockout expression data to highlight binding sites for the transcription factor HNF1alpha. *In Silico Biol,* **3** (1-2), 57–70.

[Lodish *et al.*, 2000] Lodish, H., Berk, A., Zipursky, S., Matsudaira, P., Baltimore, D. & Darnell, J. (2000) *Molecular Cell Biology, Chapter 10:Regulation of Transcription Initiation*. 4. edition,.

[Loots *et al.*, 2002] Loots, G. G., Ovcharenko, I., Pachter, L., Dubchak, I. & Rubin, E. M. (2002) rVista for comparative sequence-based discovery of functional transcription factor binding sites. *Genome Res,* **12** (5), 832–9.

[Mach, 2002] Mach, V. (2002) PRESTA: associating promoter sequences with information on gene expression. *Genome Biol,* **3** (9), research0050.

[Mancheron & Rusu, 2003] Mancheron, A. & Rusu, I. (2003) Pattern discovery allowing gaps, substitution matrices and multiple score functions. In *Algorithms in Bioinformatics. Proceedings of the 3rd International Workshop on Algorithms in BioInformatics (WABI)*, (Benson, G. & Page, R., eds), vol. 2812, of *Lecture Notes in Bioinformatics* pp. 129–145 Springer-Verlag.

[Mangalam, 2002] Mangalam, H. (2002) The Bio* toolkits–a brief overview. *Brief Bioinform,* **3** (3), 296–302.

[Marchal *et al.*, 2004] Marchal, K., Keersmaecker, S. D., Monsieurs, P., van Boxel, N., Lemmens, K., Thijs, G., Vanderleyden, J. & Moor, B. D. (2004) In silico identification and experimental validation of PmrAB targets in Salmonella typhimurium by regulatory motif detection. *Genome Biol,* **5** (2), R9.

[Marchal *et al.*, 2003] Marchal, K., Thijs, G., Keersmaecker, S. D., Monsieurs, P., Moor, B. D. & Vanderleyden, J. (2003) Genome-specific higher-order background models to improve motif detection. *Trends Microbiol,* **11** (2), 61–6.

[Marino-Ramirez *et al.*, 2004] Marino-Ramirez, L., Spouge, J. L., Kanga, G. C. & Landsman, D. (2004) Statistical analysis of over-represented words in human promoter sequences. *Nucleic Acids Res,* **32** (3), 949–58.

[Markstein *et al.*, 2002] Markstein, M., Markstein, P., Markstein, V. & Levine, M. S. (2002) Genome-wide analysis of clustered Dorsal binding sites identifies putative target genes in the Drosophila embryo. *Proc Natl Acad Sci U S A,* **99** (2), 763–8.

[Markstein *et al.*, 2004] Markstein, M., Zinzen, R., Markstein, P., Yee, K.-P., Erives, A., Stathopoulos, A. & Levine, M. (2004) A regulatory code for neurogenic gene expression in the Drosophila embryo. *Development,* **131** (10), 2387–94.

[Marsan, 2002] Marsan, L. (2002). *Inference de motifs structures: algorithmes et outils appliquees a la detection de sites de fixation dans des sequences genomiques*. PhD thesis, Universite Marne-la-Vallee.

[Marsan & Sagot, 2000] Marsan, L. & Sagot, M. (2000) Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification. *J Comput Biol,* **7** (3-4), 345–62.

[Matys *et al.*, 2003] Matys, V., Fricke, E., Geffers, R., G?ssling, E., Haubrock, M., Hehl, R., Hornischer, K., Karas, D., Kel, A., Kel-Margoulis, O., Kloos, D.-U., Land, S., Lewicki-Potapov, B., Michael, H., M?nch, R., Reuter, I., Rotert, S., Saxel, H.,

Scheer, M., Thiele, S. & Wingender, E. (2003) TRANSFAC: transcriptional regulation, from patterns to profiles. *Nucleic Acids Res,* **31** (1), 374–8.

[Mayer *et al.*, 2004] Mayer, H., Bilban, M., Kurtev, V., Gruber, F., Wagner, O., Binder, B. R. & de Martin, R. (2004) Deciphering regulatory patterns of inflammatory gene expression from interleukin-1-stimulated human endothelial cells. *Arterioscler Thromb Vasc Biol,* **24** (7), 1192–8.

[McCue *et al.*, 2002] McCue, L. A., Thompson, W., Carmack, C. S. & Lawrence, C. E. (2002) Factors influencing the identification of transcription factor binding sites by cross-species comparison. *Genome Res,* **12** (10), 1523–32.

[McGuire & Church, 1999] McGuire, A. & Church, G. (1999). The first international bioinformatics summer school ibss'99: discovery of dna regulatory motifs (workshop handout). Workshop Handout.

[McGuire *et al.*, 2000] McGuire, A., Hughes, J. & Church, G. (2000) Conservation of DNA regulatory motifs and discovery of new motifs in microbial genomes. *Genome Res,* **10** (6), 744–57.

[McGuire & Church, 2000] McGuire, A. M. & Church, G. (2000) Predicting regulons and their cis-regulatory motifs by comparative genomics. *Nucleic Acids Res,* **28** (22), 4523–30.

[Mengeritsky & Smith, 1987] Mengeritsky, G. & Smith, T. (1987) Recognition of characteristic patterns in sets of functionally equivalent DNA sequences. *Comput Appl Biosci,* **3** (3), 223–7.

[Mohr *et al.*, 1998] Mohr, E., Horn, F., Janody, F., Sanchez, C., Pillet, V., Bellon, B., Röder, L. & Jacq, B. (1998) FlyNets and GIF-DB, two internet databases for molecular interactions in Drosophila melanogaster. *Nucleic Acids Res,* **26** (1), 89–93.

[Montgomery *et al.*, 2004] Montgomery, S. B., Astakhova, T., Bilenky, M., Birney, E., Fu, T., Hassel, M., Melsopp, C., Rak, M., Robertson, A. G., Sleumer, M., Siddiqui, A. S. & Jones, S. J. M. (2004) Sockeye: a 3D environment for comparative genomics. *Genome Res,* **14** (5), 956–62.

[Moses *et al.*, 2004] Moses, A., Chiang, D. & Eisen, M. (2004) Phylogenetic motif detection by expectation-maximization on evolutionary mixtures. *Pac Symp Biocomput,* **10**, 324–35.

[Moses *et al.*, 2003] Moses, A. M., Chiang, D. Y., Kellis, M., Lander, E. S. & Eisen, M. B. (2003) Position specific variation in the rate of evolution in transcription factor binding sites. *BMC Evol Biol,* **3** (1), 19.

[Murphy, 2002] Murphy, D. (2002) Gene expression studies using microarrays: principles, problems, and prospects. *Adv Physiol Educ,* **26** (1-4), 256–70.

[Neuwald *et al.*, 1995] Neuwald, A., Liu, J. & Lawrence, C. (1995) Gibbs motif sampling: detection of bacterial outer membrane protein repeats. *Protein Sci,* **4** (8), 1618–32.

[Ohler *et al.*, 2002] Ohler, U., chun Liao, G., Niemann, H. & Rubin, G. M. (2002) Computational analysis of core promoters in the Drosophila genome. *Genome Biol,* **3** (12), RESEARCH0087.

[Olman *et al.*, 2003] Olman, V., Xu, D. & Xu, Y. (2003) CUBIC: identification of regulatory binding sites through data clustering. *J Bioinform Comput Biol,* **1** (1), 21–40.

[Olsen et al, 2002] Olsen et al, R. e. a. (2002). Chimpanzee sequencing whitepaper, proposals at the national human genome research initiative. Whitepaper.

[Orian *et al.*, 2003] Orian, A., van Steensel, B., Delrow, J., Bussemaker, H. J., Li, L., Sawado, T., Williams, E., Loo, L. W. M., Cowley, S. M., Yost, C., Pierce, S., Edgar, B. A., Parkhurst, S. M. & Eisenman, R. N. (2003) Genomic binding by the Drosophila Myc, Max, Mad/Mnt transcription factor network. *Genes Dev,* **17** (9), 1101–14.

[Owen & Zelent, 2000] Owen, G. & Zelent, A. (2000) Origins and evolutionary diversification of the nuclear receptor superfamily. *Cell Mol Life Sci,* **57** (5), 809–27.

[Papatsenko *et al.*, 2002] Papatsenko, D. A., Makeev, V. J., Lifanov, A. P., gnier, M., Nazina, A. G. & Desplan, C. (2002) Extraction of functional binding sites from unique regulatory regions: the Drosophila early developmental enhancers. *Genome Res,* **12** (3), 470–81.

[Pavesi *et al.*, 2001] Pavesi, G., Mauri, G. & Pesole, G. (2001) An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics,* **17 Suppl 1**, S207–14.

[Pavesi *et al.*, 2004*a*] Pavesi, G., Mauri, G. & Pesole, G. (2004*a*) In silico representation and discovery of transcription factor binding sites. *Brief Bioinform,* **5** (3), 217–36.

[Pavesi *et al.*, 2004*b*] Pavesi, G., Mereghetti, P., Mauri, G. & Pesole, G. (2004*b*) Weeder Web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucleic Acids Res,* **32** (Web Server issue), W199–203.

[Pedersen *et al.*, 1999] Pedersen, A., Baldi, P., Chauvin, Y. & Brunak, S. (1999) The biology of eukaryotic promoter prediction–a review. *Comput Chem,* **23** (3-4), 191–207.

[Pennacchio & Rubin, 2003] Pennacchio, L. A. & Rubin, E. M. (2003) Comparative genomic tools and databases: providing insights into the human genome. *J Clin Invest,* **111** (8), 1099–106.

[Pesole *et al.*, 1992] Pesole, N., Prunella, S., Liuni, M., Attimonelli & Saccon, C. (1992) WORDUP: an efficient algorithm for discovering statistically significant patterns in DNA sequences. *Nucleic Acids Res,* **20** (11), 2871–5.

[Pevzner *et al.*, 1989] Pevzner, P., MYu, P. B. & Mironov, A. (1989) Linguistics of nucleotide sequences. I: The significance of deviations from mean statistical characteristics and prediction of the frequencies of occurrence of words. *J Biomol Struct Dyn,* **6** (5), 1013–26.

[Pevzner & Sze, 2000] Pevzner, P. & Sze, S. (2000) Combinatorial approaches to finding subtle signals in DNA sequences. *Proc Int Conf Intell Syst Mol Biol,* **8**, 269–78.

[Philippakis *et al.*, 2005] Philippakis, A., He, F. & Bulyk, M. (2005) Modulefinder: a tool for computational discovery of cis regulatory modules. In *Proceeding of the Pacific Symposium on Biocomputing 2005*.

[Phuong *et al.*, 2003] Phuong, T. M., Lee, D. & Lee, K.-H. (2003) Regulatory element discovery using tree-structured models. In *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003), 19-22 December 2003, Melbourne, Florida, USA* pp. 739–742.

[Pollard *et al.*, 2004] Pollard, D. A., Bergman, C. M., Stoye, J., Celniker, S. E. & Eisen, M. B. (2004) Benchmarking tools for the alignment of functional noncoding DNA. *BMC Bioinformatics,* **5** (1), 6.

[Poluliakh *et al.*, 2002] Poluliakh, N., Konno, M., Takagi, T. & Nakai, K. (2002). Parameter landscape analysis for improving the performance of common motif detection algorithms. Poster at the 13th International Conference on Genome Informatics, Tokyo, Japan.

[Poluliakh *et al.*, 2003] Poluliakh, N., Takagi, T. & Nakai, K. (2003) Melina: motif extraction from promoter regions of potentially co-regulated genes. *Bioinformatics,* **19** (3), 423–4.

[Prakash *et al.*, 2004] Prakash, A., Blanchette, M., Sinha, S. & Tompa, M. (2004) Motif discovery in heterogeneous sequence data. *Pac Symp Biocomput,* **10**, 348–59.

[Praz *et al.*, 2002] Praz, V., Perier, R., Bonnard, C. & Bucher, P. (2002) The Eukaryotic Promoter Database, EPD: new entry types and links to gene expression data. *Nucleic Acids Res,* **30** (1), 322–4.

[Prestridge, 1991] Prestridge, D. (1991) SIGNAL SCAN: a computer program that scans DNA sequences for eukaryotic transcriptional elements. *Comput Appl Biosci,* **7** (2), 203–6.

[Pribnow, 1975] Pribnow, D. (1975) Nucleotide sequence of an RNA polymerase binding site at an early T7 promoter. *Proc Natl Acad Sci U S A,* **72** (3), 784–8.

[Price *et al.*, 2003] Price, A., Ramabhadran, S. & Pevzner, P. A. (2003) Finding subtle motifs by branching from sample strings. *Bioinformatics,* **19 Suppl 2**, II149–II155.

[Pritsker *et al.*, 2004] Pritsker, M., Liu, Y.-C., Beer, M. A. & Tavazoie, S. (2004) Whole-genome discovery of transcription factor binding sites by network-level conservation. *Genome Res,* **14** (1), 99–108.

[Prohaska *et al.*, 2004] Prohaska, S. J., Fried, C., Flamm, C., nter P Wagner & Stadler, P. F. (2004) Surveying phylogenetic footprints in large gene clusters: applications to Hox cluster duplications. *Mol Phylogenet Evol,* **31** (2), 581–604.

[Pruitt & Maglott, 2001] Pruitt, K. & Maglott, D. (2001) RefSeq and LocusLink: NCBI gene-centered resources. *Nucleic Acids Res,* **29** (1), 137–40.

[Purves *et al.*, 1998] Purves, W. K., Orians, G., Heller, H. & Sadava, D. (1998) *Life: The Science of Biology, Chapter 14*. 5. edition,, W H Freeman & Co.

[Périer *et al.*, 1998] Périer, R. C., Junier, T. & Bucher, P. (1998) The Eukaryotic Promoter Database EPD. *Nucleic Acids Res,* **26** (1), 353–7.

[Quandt *et al.*, 1995] Quandt, K., Frech, K., Karas, H., Wingender, E. & Werner, T. (1995) MatInd and MatInspector: new fast and versatile tools for detection of consensus matches in nucleotide sequence data. *Nucleic Acids Res,* **23** (23), 4878–84.

[Rajewsky *et al.*, 2002*a*] Rajewsky, N., Socci, N. D., Zapotocky, M. & Siggia, E. D. (2002*a*) The evolution of DNA regulatory regions for proteo-gamma bacteria by interspecies comparisons. *Genome Res,* **12** (2), 298–308.

[Rajewsky *et al.*, 2002*b*] Rajewsky, N., Vergassola, M., Gaul, U. & Siggia, E. D. (2002*b*) Computational detection of genomic cis-regulatory modules applied to body patterning in the early drosophila embryo. *BMC Bioinformatics,* **3** (1), 30.

[Ramirez-Parra *et al.*, 2003] Ramirez-Parra, E., Fründt, C. & Gutierrez, C. (2003) A genome-wide identification of E2F-regulated genes in Arabidopsis. *Plant J,* **33** (4), 801–11.

[Riechmann *et al.*, 2000] Riechmann, J., Heard, J., Martin, G., Reuber, L., Jiang, C., Keddie, J., Adam, L., Pineda, O., Ratcliffe, O., Samaha, R., Creelman, R., Pilgrim, M., Broun, P., Zhang, J., Ghandehari, D., Sherman, B. & Yu, G. (2000) Arabidopsis transcription factors: genome-wide comparative analysis among eukaryotes. *Science,* **290** (5499), 2105–10.

[Rigoutsos & Floratos, 1998] Rigoutsos, I. & Floratos, A. (1998) Combinatorial pattern discovery in biological sequences: The TEIRESIAS algorithm. *Bioinformatics,* **14** (1), 55–67.

[Robison *et al.*, 1998*a*] Robison, K., McGuire, A. & Church, G. (1998*a*) A comprehensive library of DNA-binding site matrices for 55 proteins applied to the complete Escherichia coli K-12 genome. *J Mol Biol,* **284** (2), 241–54.

[Robison *et al.*, 1998*b*] Robison, K., McGuire, A. & Church, G. (1998*b*) A comprehensive library of DNA-binding site matrices for 55 proteins applied to the complete Escherichia coli K-12 genome. *J Mol Biol,* **284** (2), 241–54.

[Rombauts *et al.*, 2003] Rombauts, S., Florquin, K., Lescot, M., Marchal, K., Rouz?, P. & Peer, Y. v. d. (2003) Computational approaches to identify promoters and cis-regulatory elements in plant genomes. *Plant Physiol,* **132** (3), 1162–76.

[Roth *et al.*, 1998] Roth, F., Hughes, J., Estep, P. & Church, G. (1998) Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat Biotechnol,* **16** (10), 939–45.

[Rust *et al.*, 2003] Rust, A., Ramsey, S., Robinson, M. & Bolouri, H. (2003) Reconstructing transcriptional regulatory networks via the integration and optimisation of multiple binding site prediction algorithms (poster). In *Proceedings of the Int. Conf. on Systems Biology (ISMB2003)*.

[Rutherford *et al.*, 2000] Rutherford, K., Parkhill, J., Crook, J., Horsnell, T., Rice, P., Rajandream, M. & Barrell, B. (2000) Artemis: sequence visualization and annotation. *Bioinformatics,* **16** (10), 944–5.

[Sachs, 1999] Sachs, L. (1999) *Angewandte Statistik. Anwendung statistischer Methoden*. 9. edition„ Springer Verlag.

[Sagot, 1998] Sagot, M.-F. (1998) Spelling approximate repeated or common motifs using a suffix tree. In *Proceedings of the Third Latin American Symposium on Theoretical Informatics* pp. 374–390 Springer-Verlag.

[Sandelin *et al.*, 2004] Sandelin, A., Alkema, W., r m, Wasserman, W. W. & Lenhard, B. (2004) JASPAR: an open-access database for eukaryotic transcription factor binding profiles. *Nucleic Acids Res,* **32 Database issue**, D91–4.

[Sandelin & Wasserman, 2004] Sandelin, A. & Wasserman, W. W. (2004) Constrained binding site diversity within families of transcription factors enhances pattern discovery bioinformatics. *J Mol Biol,* **338** (2), 207–15.

[Sandelin *et al.*, 2004] Sandelin, A., Wasserman, W. W. & Lenhard, B. (2004) ConSite: web-based prediction of regulatory elements using cross-species comparison. *Nucleic Acids Res,* **32** (Web Server issue), W249–52.

[Schbath, 1997] Schbath, S. (1997) An efficient statistic to detect over- and under-represented words in DNA sequences. *J Comput Biol,* **4** (2), 189–92.

[Scherf *et al.*, 2000] Scherf, M., Klingenhoff, A. & Werner, T. (2000) Highly specific localization of promoter regions in large genomic sequences by PromoterInspector: a novel context analysis approach. *J Mol Biol,* **297** (3), 599–606.

[Schmid *et al.*, 2003] Schmid, C. D., Praz, V., Delorenzi, M., Perier, R. & Bucher, P. (2003) Automatic procedures for compilation of promoter sequences and their

evaluation based on signal content and positional distributions. In *Proceedings of the European Conference on Computational Biology 2003, Poster Abstracts 7*.

[Schmid *et al.*, 2004] Schmid, C. D., Praz, V., Delorenzi, M., Périer, R. & Bucher, P. (2004) The Eukaryotic Promoter Database EPD: the impact of in silico primer extension. *Nucleic Acids Res,* **32 Database issue**, D82–5.

[Schneider *et al.*, 1986] Schneider, T., Stormo, G., Gold, L. & Ehrenfeucht, A. (1986) Information content of binding sites on nucleotide sequences. *J Mol Biol,* **188** (3), 415–31.

[Schneider, 2002] Schneider, T. D. (2002) Consensus sequence Zen. *Appl Bioinformatics,* **1** (3), 111–9.

[Schroeder *et al.*, 2004] Schroeder, M. D., Pearce, M., Fak, J., Fan, H., Unnerstall, U., Emberly, E., Rajewsky, N., Siggia, E. D. & Gaul, U. (2004) Transcriptional control in the segmentation gene network of Drosophila. *PLoS Biol,* **2** (9), E271.

[Sharan *et al.*, 2004] Sharan, R., Ben-Hur, A., Loots, G. G. & Ovcharenko, I. (2004) CREME: Cis-Regulatory Module Explorer for the human genome. *Nucleic Acids Res,* **32** (Web Server issue), W253–6.

[Shinozaki *et al.*, 2003] Shinozaki, D., Akutsu, T. & Maruyama, O. (2003) Finding optimal degenerate patterns in DNA sequences. *Bioinformatics,* **19 Suppl 2**, II206–II214.

[Shultzaberger & Schneider, 1999] Shultzaberger, R. & Schneider, T. (1999) Using sequence logos and information analysis of Lrp DNA binding sites to investigate discrepanciesbetween natural selection and SELEX. *Nucleic Acids Res,* **27** (3), 882–7.

[Sinha, 2000] Sinha, S. (2000). Composite motifs in promoter regions of genes: models and algorithms. Generals report Washington University http://uqbar.rockefeller.edu/ saurabh/papers/generals.pdf.

[Sinha, 2003] Sinha, S. (2003) Discriminative motifs. *J Comput Biol,* **10** (3-4), 599–615.

[Sinha *et al.*, 2004] Sinha, S., Schroeder, M. D., Unnerstall, U., Gaul, U. & Siggia, E. D. (2004) Cross-species comparison significantly improves genome-wide prediction of cis-regulatory modules in Drosophila. *BMC Bioinformatics,* **5** (1), 129.

[Sinha & Tompa, 2000] Sinha, S. & Tompa, M. (2000) A statistical method for finding transcription factor binding sites. *Proc Int Conf Intell Syst Mol Biol,* **8**, 344–54.

[Sinha & Tompa, 2002] Sinha, S. & Tompa, M. (2002) Discovery of novel transcription factor binding sites by statistical overrepresentation. *Nucleic Acids Res,* **30** (24), 5549–60.

[Sinha & Tompa, 2003] Sinha, S. & Tompa, M. (2003) Performance comparison of algorithms for finding transcription factor binding sites. In *Proceedings of the Third IEEE Symposium on BioInformatics and BioEngineering (BIBE'03)* vol. 3, pp. 214–223 IEEE IEEE Computer Society.

[Sinha *et al.*, 2003] Sinha, S., van Nimwegen, E. & Siggia, E. D. (2003) A probabilistic method to detect regulatory modules. *Bioinformatics,* **19 Suppl 1**, i292–301.

[Solovyev & Shahmuradov, 2003] Solovyev, V. & Shahmuradov, I. (2003) PromH: Promoters identification using orthologous genomic sequences. *Nucleic Acids Res,* **31** (13), 3540–5.

[Spellman *et al.*, 1998] Spellman, P., Sherlock, G., Zhang, M., Iyer, V., Anders, K., Eisen, M., Brown, P., Botstein, D. & Futcher, B. (1998) Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell,* **9** (12), 3273–97.

[Staden, 1984] Staden, R. (1984) Computer methods to locate signals in nucleic acid sequences. *Nucleic Acids Res,* **12** (1 Pt 2), 505–19.

[Staden, 1989] Staden, R. (1989) Methods for discovering novel motifs in nucleic acid sequences. *Comput Appl Biosci,* **5** (4), 293–8.

[Stojanovic *et al.*, 1999] Stojanovic, N., Florea, L., Riemer, C., Gumucio, D., Slightom, J., Goodman, M., Miller, W. & Hardison, R. (1999) Comparison of five methods for finding conserved sequences in multiple alignments of gene regulatory regions. *Nucleic Acids Res,* **27** (19), 3899–910.

[Stone & Wray, 2001] Stone, J. & Wray, G. (2001) Rapid evolution of cis-regulatory sequences via local point mutations. *Mol Biol Evol,* **18** (9), 1764–70.

[Stormo, 1990] Stormo, G. (1990) Consensus patterns in DNA. *Methods Enzymol,* **183**, 211–21.

[Stormo, 2000] Stormo, G. (2000) DNA binding sites: representation and discovery. *Bioinformatics,* **16** (1), 16–23.

[Stormo & Hartzell, 1989] Stormo, G. & Hartzell, G. (1989) Identifying protein-binding sites from unaligned DNA fragments. *Proc Natl Acad Sci U S A,* **86** (4), 1183–7.

[Stormo *et al.*, 1982] Stormo, G., Schneider, T. & Gold, L. (1982) Characterization of translational initiation sites in E. coli. *Nucleic Acids Res,* **10** (9), 2971–96.

[Strausberg *et al.*, 2002] Strausberg, R. L., Feingold, E. A., Grouse, L. H., Derge, J. G., Klausner, R. D., Collins, F. S., Wagner, L., Shenmen, C. M., Schuler, G. D., Altschul, S. F., Zeeberg, B., Buetow, K. H., Schaefer, C. F., Bhat, N. K., Hopkins, R. F., Jordan, H., Moore, T., Max, S. I., Wang, J., Hsieh, F., Diatchenko, L., Marusina, K., Farmer, A. A., Rubin, G. M., Hong, L., Stapleton, M., Soares, M. B., Bonaldo, M. F., Casavant, T. L., Scheetz, T. E., Brownstein, M. J., Usdin, T. B., Toshiyuki, S., Carninci, P., Prange, C., Raha, S. S., Loquellano, N. A., Peters, G. J., Abramson, R. D., Mullahy, S. J., Bosak, S. A., McEwan, P. J., McKernan, K. J., Malek, J. A., Gunaratne, P. H., Richards, S., Worley, K. C., Hale, S., Garcia, A. M., Gay, L. J., Hulyk, S. W., Villalon, D. K., Muzny, D. M., Sodergren, E. J., Lu, X., Gibbs, R. A., Fahey, J., Helton, E., Ketteman, M., Madan, A., Rodrigues, S., Sanchez, A., Whiting, M., Madan, A., Young, A. C., Shevchenko, Y., Bouffard, G. G., Blakesley, R. W., Touchman, J. W., Green, E. D., Dickson, M. C., Rodriguez, A. C., Grimwood, J., Schmutz, J., Myers, R. M., Butterfield, Y. S. N., Krzywinski, M. I., Skalska, U., Smailus, D. E., Schnerch, A., Schein, J. E., Jones, S. J. M., Marra, M. A. & Marra, M. G. C. P. T. (2002) Generation and initial analysis of more than 15,000 full-length human and mouse cDNA sequences. *Proc Natl Acad Sci U S A,* **99** (26), 16899–903.

[Sumazin *et al.*, 2005] Sumazin, P., Chen, G., Hata, N., Smith, A. D., Zhang, T. & Zhang, M. Q. (2005) DWE: discriminating word enumerator. *Bioinformatics,* **21** (1), 31–8.

[Sun *et al.*, 2004] Sun, Z., Yang, J. & Deogun, J. S. (2004) Misae: a new approach for regulatory motif extraction. In *3rd International IEEE Computer Society Computational Systems Bioinformatics Conference (CSB 2004), 16-19 August 2004, Stanford, CA, USA* pp. 173–181.

[Suzuki *et al.*, 2001] Suzuki, Y., Tsunoda, T., Sese, J., Taira, H., Mizushima-Sugano, J., Hata, H., Ota, T., Isogai, T., Tanaka, T., Nakamura, Y., Suyama, A., Sakaki, Y.,

Morishita, S., Okubo, K. & Sugano, S. (2001) Identification and characterization of the potential promoter regions of 1031 kinds of human genes. *Genome Res,* **11** (5), 677–84.

[Suzuki *et al.*, 2002] Suzuki, Y., Yamashita, R., Nakai, K. & Sugano, S. (2002) DBTSS: DataBase of human Transcriptional Start Sites and full-length cDNAs. *Nucleic Acids Res,* **30** (1), 328–31.

[Sze *et al.*, 2002] Sze, S., Gelfand, M. & Pevzner, P. (2002) Finding weak motifs in DNA sequences. *Pac Symp Biocomput,* **7**, 235–46.

[Takusagawa & Gifford, 2004] Takusagawa, K. & Gifford, D. (2004) Negative information for motif discovery. *Pac Symp Biocomput,* **9**, 360–71.

[Tavazoie *et al.*, 1999] Tavazoie, S., Hughes, J., Campbell, M., Cho, R. & Church, G. (1999) Systematic determination of genetic network architecture. *Nat Genet,* **22** (3), 281–5.

[Thieffry *et al.*, 1998] Thieffry, D., Salgado, H., Huerta, A. & Collado-Vides, J. (1998) Prediction of transcriptional regulatory sites in the complete genome sequence of Escherichia coli K-12. *Bioinformatics,* **14** (5), 391–400.

[Thijs *et al.*, 2001] Thijs, G., Lescot, M., Marchal, K., Rombauts, S., Moor, B. D., é, P. & Moreau, Y. (2001) A higher-order background model improves the detection of promoter regulatory elements by Gibbs sampling. *Bioinformatics,* **17** (12), 1113–22.

[Tompa *et al.*, 2005] Tompa, M., Li, N., Bailey, T. L., Church, G. M., Moor, B. D., Eskin, E., Favorov, A. V., Frith, M. C., Fu, Y., Kent, W. J., Makeev, V. J., Mironov, A. A., Noble, W. S., Pavesi, G., Pesole, G., Régnier, M., Simonis, N., Sinha, S., Thijs, G., van Helden, J., Vandenbogaert, M., Weng, Z., Workman, C., Ye, C. & Zhu, Z. (2005) Assessing computational tools for the discovery of transcription factor binding sites. *Nat Biotechnol,* **23** (1), 137–44.

[Tronche *et al.*, 1997] Tronche, F., Ringeisen, F., Blumenfeld, M., Yaniv, M. & Pontoglio, M. (1997) Analysis of the distribution of binding sites for a tissue-specific transcription factor in the vertebrate genome. *J Mol Biol,* **266** (2), 231–45.

[Tullai *et al.*, 2004] Tullai, J. W., Schaffer, M. E., Mullenbrock, S., Kasif, S. & Cooper, G. M. (2004) Identification of transcription factor binding sites upstream of human genes regulated by the phosphatidylinositol 3-kinase and MEK/ERK signaling pathways. *J Biol Chem,* **279** (19), 20167–77.

[van Helden, 2003] van Helden, J. (2003) Regulatory sequence analysis tools. *Nucleic Acids Res,* **31** (13), 3593–6.

[van Helden *et al.*, 2000*a*] van Helden, J., André, B. & Collado-Vides, J. (2000*a*) A web site for the computational analysis of yeast regulatory sequences. *Yeast,* **16** (2), 177–87.

[van Helden *et al.*, 2000*b*] van Helden, J., del Olmo, M. & Pérez-Ortín, J. (2000*b*) Statistical analysis of yeast genomic downstream sequences reveals putative polyadenylation signals. *Nucleic Acids Res,* **28** (4), 1000–10.

[van Helden *et al.*, 2000*c*] van Helden, J., Rios, A. & Collado-Vides, J. (2000*c*) Discovering regulatory elements in non-coding sequences by analysis of spaced dyads. *Nucleic Acids Res,* **28** (8), 1808–18.

[van Steensel *et al.*, 2003] van Steensel, B., Delrow, J. & Bussemaker, H. J. (2003) Genomewide analysis of Drosophila GAGA factor target genes reveals context-dependent DNA binding. *Proc Natl Acad Sci U S A,* **100** (5), 2580–5.

[Vanet *et al.*, 2000] Vanet, A., Marsan, L., Labigne, A. & Sagot, M. (2000) Inferring regulatory elements from a whole genome. An analysis of Helicobacter pylori sigma(80) family of promoter signals. *J Mol Biol,* **297** (2), 335–53.

[Vanet *et al.*, 1999] Vanet, A., Marsan, L. & Sagot, M. (1999) Promoter sequences and algorithmical methods for identifying them. *Res Microbiol,* **150** (9-10), 779–99.

[VijayaSatya & Mukherjee, 2004] VijayaSatya, R. & Mukherjee, A. (2004) Pruner: algorithms for finding monad patterns in dna sequences. In *3rd International IEEE Computer Society Computational Systems Bioinformatics Conference (CSB 2004), 16-19 August 2004, Stanford, CA, USA* IEEE Computer Society.

[Vilo, 2002] Vilo, J. (2002). *Pattern Disocovery from Biosequences*. PhD thesis, University of Helsinki.

[Wang & Stormo, 2003] Wang, T. & Stormo, G. D. (2003) Combining phylogenetic data with co-regulated genes to identify regulatory motifs. *Bioinformatics,* **19** (18), 2369–80.

[Washio *et al.*, 1998] Washio, T., Sasayama, J. & Tomita, M. (1998) Analysis of complete genomes suggests that many prokaryotes do not rely on hairpin formation in transcription termination. *Nucleic Acids Res,* **26** (23), 5456–63.

[Wasserman & Sandelin, 2004] Wasserman, W. W. & Sandelin, A. (2004) Applied bioinformatics for the identification of regulatory elements. *Nat Rev Genet,* **5** (4), 276–87.

[Waterman *et al.*, 1984] Waterman, M., Arratia, R. & Galas, D. (1984) Pattern recognition in several sequences: consensus and alignment. *Bull Math Biol,* **46** (4), 515–27.

[Welling, nown] Welling, M. (YearUnknown). The em-algorithm (lecture notes). http://www.ics.uci.edu/∼welling/classnotes/classnotes.html.

[Werner, 1999] Werner, T. (1999) Models for prediction and recognition of eukaryotic promoters. *Mamm Genome,* **10** (2), 168–75.

[Werner, 2003] Werner, T. (2003) The state of the art of mammalian promoter recognition. *Brief Bioinform,* **4** (1), 22–30.

[Wolfertstetter *et al.*, 1996] Wolfertstetter, F., Frech, K., Herrmann, G. & Werner, T. (1996) Identification of functional elements in unaligned nucleic acid sequences by a novel tuple search algorithm. *Comput Appl Biosci,* **12** (1), 71–80.

[Workman & Stormo, 2000] Workman, C. & Stormo, G. (2000) 7048258. *Pac Symp Biocomput,* **5**, 467–78.

[Wray *et al.*, 2003] Wray, G. A., Hahn, M. W., Abouheif, E., Balhoff, J. P., Pizer, M., Rockman, M. V. & Romano, L. A. (2003) The evolution of transcriptional regulation in eukaryotes. *Mol Biol Evol,* **20** (9), 1377–419.

[Wu *et al.*, 2004] Wu, X., Wang, B., Song, C. & Cheng, J. (2004) A combined model and a varied gibbs sampling algorithm used for motif discovery. In *Proceedings of the second conference on Asia-Pacific bioinformatics* pp. 99–104 Australian Computer Society, Inc.

[Xing & Karp, 2004] Xing, E. P. & Karp, R. M. (2004) MotifPrototyper: a Bayesian profile model for motif families. *Proc Natl Acad Sci U S A,* **101** (29), 10523–8.

[Xing *et al.*, 2004] Xing, E. P., Wu, W., Jordan, M. I. & Karp, R. M. (2004) Logos: a modular bayesian model for de novo motif detection. *J Bioinform Comput Biol,* **2** (1), 127–54.

[Yada *et al.*, 1998] Yada, T., Totoki, Y., Ishikawa, M., Asai, K. & Nakai, K. (1998) Automatic extraction of motifs represented in the hidden Markov model from a number of DNA sequences. *Bioinformatics,* **14** (4), 317–25.

[Yper *et al.*, 2003] Yper, S. V., Thas, O., Ottoy, J.-P. & Criekinge, W. V. (2003). Phylogenetic footprinting of co-expressed genes by tree-gibbs sampling. Poster at Europen Conference on Computational Biology 2003.

[Zavolan *et al.*, 2003] Zavolan, M., Socci, N., Rajewsky, N. & Gaasterland, T. (2003) SMASHing regulatory sites in DNA by human-mouse sequence comparisons. In *2nd IEEE Computer Society Bioinformatics Conference (CSB 2003), 11-14 August 2003, Stanford, CA, USA* IEEE Computer Society.

[Zhang, 1999] Zhang, M. (1999) Promoter analysis of co-regulated genes in the yeast genome. *Comput Chem,* **23** (3-4), 233–50.

[Zhang, 2002] Zhang, M. Q. (2002) Computational methods for promotor recognition. In *Current Topics in Computational Biology*, (Jiang, T., Xu, Y. & Zhang, M. Q., eds),. The MIT Press.

[Zhang & Zhang, 2001] Zhang, T. & Zhang, M. (2001) Promoter Extraction from GenBank (PEG): automatic extraction of eukaryotic promoter sequences in large sets of genes. *Bioinformatics,* **17** (12), 1232–3.

[Zheng *et al.*, 2003] Zheng, J., Wu, J. & Sun, Z. (2003) An approach to identify over-represented cis-elements in related sequences. *Nucleic Acids Res,* **31** (7), 1995–2005.

[Zhou & Liu, 2004] Zhou, Q. & Liu, J. S. (2004) Modeling within-motif dependence for transcription factor binding site predictions. *Bioinformatics,* **20** (6), 909–16.

[Zhu & Zhang, 1999] Zhu, J. & Zhang, M. (1999) SCPD: a promoter database of the yeast Saccharomyces cerevisiae. *Bioinformatics,* **15** (7-8), 607–11.

[Zhu *et al.*, 2002] Zhu, Z., Pilpel, Y. & Church, G. M. (2002) Computational identification of transcription factor binding sites via a transcription-factor-centric clustering (TFCC) algorithm. *J Mol Biol,* **318** (1), 71–81.

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399