

**Learning Probabilistic Relational Models in  
the context of reverse engineering genetic  
networks**

*Christoforos Anagnostopoulos*

Master of Science  
School of Informatics  
University of Edinburgh  
2004

# Abstract

This thesis places itself in the general direction of a probabilistic approach to systems biology. In specific we are concerned with applying the framework of Probabilistic Relational Models to the task of inferring clusters of transcriptionally coregulated genes on the basis of two distinct types of data: *gene expression data* and *promoter sequence data*. Such clustering is likely to reveal groups of interdependent genes that effectively form functional units. Identifying coregulation can therefore act as the first step enabling the smaller scale analysis required for the reverse engineering of genetic networks.

In particular, we have focused on the model of transcriptional coregulation suggested in Segal et al. (2003) and have thoroughly explored the issues of generalisation performance and interpretability of the model, finding them interlinked in several ways. Finally, we have constructed a biologically interpretable regularising scheme, which performed very successfully against synthetic data.

# Acknowledgements

I would like to express my gratitude to my supervisor Dirk Husmeier for his assistance throughout the project.

## **Declaration**

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

*(Christoforos Anagnostopoulos)*

# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Our Work in Brief . . . . .	1
1.2	Inference in Systems Biology . . . . .	2
1.3	Transcriptional co-regulation . . . . .	2
1.4	Probabilistic Relational Models . . . . .	3
1.5	Our Original Objective . . . . .	4
1.6	The structure of the Thesis . . . . .	5
<b>2</b>	<b>Introducing the Model</b>	<b>7</b>
2.1	In Brief: The Model . . . . .	7
2.2	In Brief: Learning the model via expectation-maximisation . . . . .	8
2.3	In Brief: Dynamically Adding and Deleting Motifs . . . . .	10
2.4	A Note on the Mathematics . . . . .	11
<b>3</b>	<b>Motif Model</b>	<b>12</b>
3.1	Formulating our Objective . . . . .	12
3.2	Representing Motifs . . . . .	13
3.3	Motivating and Formulating the Model . . . . .	14
3.4	Reparameterising the Model . . . . .	18
3.4.1	The Semantics of the Weights . . . . .	19
3.4.2	The Semantics of the Threshold . . . . .	21
3.5	Evaluation and Overfitting . . . . .	21
3.5.1	Testing the semantics of the threshold . . . . .	23
3.5.2	Assessing the Model . . . . .	25

3.5.3	Overfitting, Self-Similarity and Multiple Motifs . . . . .	26
<b>4</b>	<b>Regulation Model</b>	<b>27</b>
4.1	Formulation of the Model . . . . .	28
4.2	Geometrical Interpretation of the Model . . . . .	29
4.2.1	Binary Classification via Separating Hyperplanes . . . . .	29
4.2.2	Multiclass Classification via maximisation of the inner product	30
4.2.3	The Shape of the Multiclass Decision Boundary . . . . .	32
4.3	Maximum Likelihood Estimation of the parameters . . . . .	33
4.4	Directions of Invariance . . . . .	36
4.5	Interpreting inactivity . . . . .	37
4.5.1	The insignificance of zero . . . . .	38
4.5.2	Approximating inactivity . . . . .	40
<b>5</b>	<b>Regulation Model: Regularisation and Pruning</b>	<b>44</b>
5.1	Training vs Test Datasets . . . . .	45
5.2	The Data Augmentation Approach to Regularisation . . . . .	46
5.2.1	Divergence . . . . .	46
5.2.2	Linear Separability for Binary Classification . . . . .	48
5.2.3	Linear Separability for the Multilogit model . . . . .	48
5.3	The Smoothing Effect of the Data Augmentation Approach . . . . .	51
5.4	The Bayesian Approach to Regularisation . . . . .	54
5.4.1	Prior Distributions . . . . .	54
5.4.2	Motivating the Exponential Priors . . . . .	55
5.4.3	Eliminating the Hyperparameters . . . . .	57
5.5	A Bayesian approach to the Sparsity Constraint . . . . .	58
5.5.1	Pruning and Sparsity . . . . .	58
5.5.2	Pruning via the Laplacian Prior Without Mathematics . . . . .	60
5.6	Performance of Regularisers . . . . .	62
<b>6</b>	<b>Biological Data</b>	<b>72</b>
6.1	Comparing our model with the Segal model . . . . .	72
6.2	The Dataset . . . . .	73

6.3	Setting up our scheme . . . . .	74
6.4	Results . . . . .	75
<b>7</b>	<b>Conclusion</b>	<b>76</b>
7.1	Outcomes . . . . .	76
7.2	Regrets . . . . .	77
7.3	Future Work . . . . .	77
7.4	Outcomes . . . . .	77
<b>A</b>	<b>Conditions for Decomposability of the weights for the Motif Model</b>	<b>78</b>
	<b>Bibliography</b>	<b>82</b>

# Chapter 1

## Introduction

### 1.1 Our Work in Brief

This thesis places itself in the general direction of a probabilistic approach to systems biology. In specific we are concerned with applying the framework of Probabilistic Relational Models to the task of inferring clusters of transcriptionally coregulated genes on the basis of two distinct types of data: *gene expression data* and *promoter sequence data*.

Such clustering is likely to reveal groups of interdependent genes that effectively form functional units. Identifying coregulation can therefore act as the first step enabling the smaller scale analysis required for the reverse engineering of genetic networks.

The specific tasks of this dissertation have been the following:

- Implement in Matlab and investigate the behaviour of the model presented in Segal et al. (2003) to perform clustering on the basis of gene expression data and promoter sequence data.
- Examine possible solutions to shortcomings of the model, paying particular attention to the issues of *interpretability* and *overfitting*, which were not sufficiently addressed in Segal et al. (2003).

Our approach to the task of improving the model was driven by the general concern of promoting *interpretability*. Our work suggests that this can be done safely and systematically using Bayesian methods of incorporating prior knowledge.

## 1.2 Inference in Systems Biology

Systems biology understands cellular biology in an *integrated* manner, whereby, to understand the functionality of particular components, one needs to understand the relationships between all components and relate them to a global view of the system. Such understanding can only be given by analyses of global information about the state of the cell at different times and conditions. Therein lies the informational challenge. We are presented with the challenge of performing inference over data obtained severally from disparate data sources, each with a large amount of experimental noise and natural variation. Performing such inference solely by human intuition assisted by conventional statistical tests is impossible. The analysis must be fully recast on probabilistic grounds and be mostly automated.

It is preferable not to decompose this task further. The success of the model jointly depends on its biological plausibility, its statistical soundness and its computational demands. Therefore, the general problematic behind this thesis is how to *customise probabilistic inference for the purposes of systems biology* in all its aspects, from the biology, through the mathematics, to the implementation.

## 1.3 Transcriptional co-regulation

Some cellular processes are local and sustainable without assistance from other modules in the cell. Such processes are the hardest to know about, since information about them does not disseminate throughout the cell. This difficulty is an indication of a general principle: in trying to understand a complex system, it is at first best not to focus on localised behavior but rather on processes that occupy *informationally central* roles in its functioning. Transcriptional co-regulation is precisely such a process.

Many cellular processes either depend on or result in protein synthesis. Protein synthesis in turn requires the retrieval of the genetic code for the protein in question. This information is sent to the protein synthesis sites in the form of messenger-RNA. The creation of messenger-RNA for the purposes of protein synthesis is known as *gene expression* and is mostly initiated by the binding of certain proteins called transcription factors on specific DNA sites called promoter sequences that are located near, mostly

upstream of the gene in question. This latter process is known as *transcriptional regulation*.

The above system of control can be ‘intercepted’ experimentally at several points, two of which will concern us in this thesis. Firstly, the upstream regions of each gene can be read. Secondly, the abundance in the cell of mRNA for each gene at a certain time can be measured. Understanding such data properly is hence very useful, since transcriptional co-regulation indicates participation in the same cellular process and along with the time and condition-dependence of gene expression data it can give the biologist accurate insight into what genes are involved in each cellular process, without need for the largely unavailable knowledge of the physical nature of the process or the chemical processes of the proteins the genes code for.

## 1.4 Probabilistic Relational Models

Until now, the two forms of data mentioned above have been effectively treated in statistical separation, although conclusions about the biology had to comply with both of them. However, informationally speaking, to the extent that mRNA production is initiated by transcription factor binding, the two datasets ought to be *jointly redundant*. This suggests they should be *jointly modelled*.

Joint modelling of data coming from disparate data sources can be most appropriately expressed in a Bayesian context, mainly because it provides the framework to properly address the concepts of *conditional dependence* and *unobserved variables*. These two concepts allow the statistician to make the link between two disparate data sources explicit by linking them together indirectly via putative variables that represent the unobserved but presumed commonality in the two datasets.

Models based on conditional dependence or, equivalently, conditional independence assumptions, have lately received a popular graphical representation and become known as *Bayesian networks*. A largely equivalent but much richer representation called *Probabilistic Relational Models* (PRMs) has been recently suggested. The task of bringing under truly joint consideration mRNA expression data and promoter sequence data, as well as possibly data from other correlated data sources, can be done most naturally in

this framework.

## 1.5 Our Original Objective

The general objective of this thesis has been to investigate the ways in which PRMs can be most successfully applied in the context of reverse engineering genetic networks and in specific the task of identifying transcriptionally coregulated genes. This attempt was initiated by a series of papers produced by affiliates to Stanford University, USA, where Bayesian Networks and Probabilistic Relational Models were applied to the modelling task previously described.

When the proposal for this thesis was formulated, this thesis was envisaged to focus on assessing the effect on model performance and interpretability of the *transition from BNs to PRMs*. Although as we explain later the two modelling frameworks are in some sense equivalent, the transition from one to the other is far from inconsequential: it affects both the *probabilistic formulation of the modelling task* and the *learning schemes* used to relate the model to the data. The original objective was therefore to effectuate a comparison of the Bayesian Network implemented in Segal et al. (2003) with one or more extensions of this model that were presented in related papers and were using the framework of PRMs.

However, in trying to establish criteria of comparison between the two models, we realised that there was room for significant improvement with respect to such criteria *within* the model of Segal et al. (2003). We therefore focused on this model instead and investigated in great extents its behavior in the directions of interest. There were two main concerns driving our analysis: interpretability and overfitting. We attempted to view these problems in a Bayesian context and devise non-heuristic ways of dealing with them. We have tried to test the plausibility of our suggestions using synthetic data. Finally, the applicability of the model has been tested on real data, coming from the plant *Arabidopsis thaliana*.

It has been impossible to proceed to investigate the effects of the transition from Bayesian Networks to PRMs within the time constraints of an MSc thesis. However,

we view the work done in this thesis as a *prerequisite* to the work envisaged in the original project proposal, since the performance of PRMs relatively to BNs with respect to a given set of criteria cannot be properly discussed unless models in each class already perform their best.

The model we have investigated consists of three subcomponents: the *motif model* searches for common motifs in the upstream regions of the genes; the *regulation model* clusters the genes according to the motifs that are present in their upstream regions and the *expression model* which predicts the expression level of each gene in each experiment according to the cluster it has been assigned to. Finally, the three components are related by conditional dependence to provide a joint distribution over both the expression data and the upstream regions. During our investigation of this model, which was mainly concerned with the issues of generalisation performance and interpretability, we conclusively dealt with several points that were addressed inconclusively or misunderstood in the original paper and have suggested possible improvements on the basis of our theoretical analysis, which we support with testing on synthetic data.

## 1.6 The structure of the Thesis

The structure of the rest of the thesis is accurately suggested by the list of outcomes presented above. The organisation of this material in chapters occurs as follows:

**Chapter 3** The chapter immediately following this one is a short literature review, where we present in more depth the major research items that made the formulation of our model possible.

**Chapter 3** This chapter is brief and serves to summarily present the model of *Segal et al* in sufficient detail to prepare the reader for the main body of the thesis.

**Chapter 4** The *motif model* is presented. The semantics of the parameters and the strengths and shortcomings of the *discriminative nature* of the model are discussed. Arguments are backed by tests on synthetic data.

**Chapter 5** The *regulation model* is presented. We begin by formulating the model and offering a geometrical interpretation of the parameters. We briefly explain how to learn the Maximum Likelihood Estimates of the parameters and discuss the invariance properties of the parameter space. We discuss the ability of the model to accommodate two biologically plausible constraints offered by Segal et al.

**Chapter 6** We investigate the issue of the divergence of the parameters in the Regulation model relating it to the problem of *overfitting*. We then motivate two potential ways to ensure convergence, using our geometrical intuition as well as Bayesian arguments. We show how the sparsity constraint can be naturally enforced via the use of one of these regularisers prior. We conclude by assessing the performance of our regularisers in synthetic datasets and contrasting them to the commonly used Gaussian regulariser.

**Chapter 7** Here we are concerned with *training the model as a whole*. We present the *approximate EM algorithm* that we use following Segal et al. (2003). We investigate the soundness of the heuristic criteria for adding and deleting motifs. We hint towards *possible improvements* on this scheme.

**Chapter 8** We apply our final model to a real dataset and present the results.

**Chapter 9** This chapter is concerned with several theoretical issues that arised during the thesis and the directions they might hint towards for future work.

**Chapter 10** This is the conclusion of this thesis, where we summarise and assess the progress made during this MSc project in the direction of unifying disparate data sources in a coherent and interpretable way.

# Chapter 2

## Introducing the Model

### 2.1 In Brief: The Model

We will implement and investigate the model presented in Segal et al. (2003). In this model there is no relational structure to correlate one gene with another. Hence, genes are treated as samples from the joint distribution over a set of random variables, each representing an attribute of a gene. The Bayesian Network representing the factorisation of the joint probability distribution which we adhere to is presented in 2.1. The architecture is four-tier feed forward:

**Motif Model** Each variable in the first tier is discrete and represents a letter in the string representation of the promoter sequence of the gene. SO collectively the first tier represents the promoter sequence. Variables in the second tier are binary; each one represents the event of regulation of the gene by a certain motif. In other words, there will be as many variables in the second tier as motifs under consideration. The dependency of the 2nd tier on the 1st constitutes the *Motif Model*.

**Regulation Model** The third tier is composed of a single discrete latent variable. This variable depends on the collective assignment of all variables in the second tier and represents the module assignment of the gene. The dependency that is being modelled is that of the event of coregulation (which implies membership to the

same module) on the motif profile of the gene, i.e. the presence/absence of key motifs in the promoter sequence of the gene in question. This dependency is modelled by the *Regulation Model*.

**Expression Model** Each variable in the fourth tier represents one experiment, for instance one mRNA abundancy measurement on a microarray. There will be as many variables in the fourth tier as the expression measurement experiments under consideration. Since we assume that gene regulation is mediated by the binding of transcription factors, we correlate the expression levels with the module assignment, which in turn depends on the motif profile of the gene. The dependency between the third and fourth tiers is the *Gene Expression Model*.

## 2.2 In Brief: Learning the model via expectation-maximisation

The above is an example of *clustering via hidden variables*. It is easy to intuitively grasp the semantics of *g.M*. The underlying assumption is that the data is produced by one of several possible generating processes. The assignment of the hidden variable for each datapoint represents our beliefs about which one process this particular datapoint has most likely been generated from.

The task of learning the optimal clustering given such a model is difficult since on the one hand we need to learn which generating process each datapoint is most likely to come from and on the other we need to learn the parameters representing each generating process by considering only datapoints that have been generated by that particular process. These two subtasks can easily be seen to be mutually dependent and cannot be disentangled.

However, there exists an iterative scheme known as the *soft EM algorithm* which has been proven to eventually converge to a solution that is locally optimal (Dempster et al., 1977). The algorithm starts off at a certain initialisation of the hidden variable assignments and learns the optimal assignment of the parameters. It then proceeds to set the hidden variables to their expected values given the parameters and iterates,

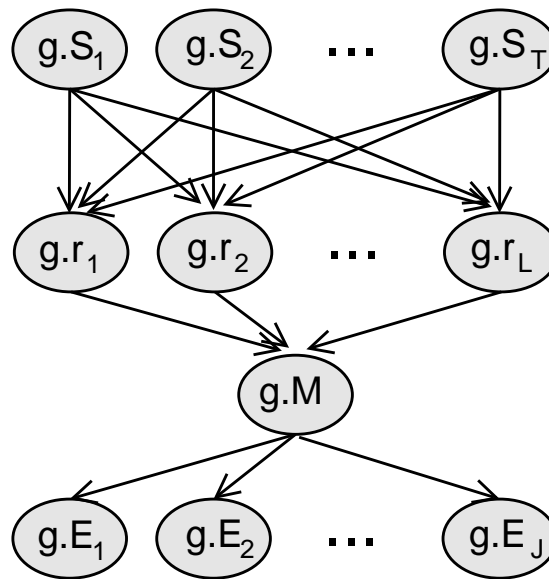


Figure 2.1: The graphical representation of our model. All variables are attributes of a gene  $g$ . The variables  $S_1, S_2, \dots, S_T$  represent the  $T$  letters in the promoter sequence of the gene. The variables  $r_1, r_2, \dots, r_L$  indicate whether each of the  $L$  motifs we are considering was present in the promoter sequence  $g.S$  of this gene. The variable  $g.M$  indicates the cluster assignment of this particular gene. Finally, variables  $g.E_1, g.E_2, \dots, g.E_J$  indicate whether  $g$  was upregulated, downregulated or unaffected in each one of the  $J$  experiments we have in our possession.

refining the setting of the parameters and the hidden variables so that the overall likelihood is improving every time.

As mentioned before, the crucial aspect of the EM algorithm is that it uses the joint probability distribution over *all* variables to evaluate the quality of the update. This gives our model its *unifying* character. However an exact implementation of a soft-EM scheme is not always possible. Evaluating the expected value of the hidden variables requires marginalisation over the joint probability distribution. In continuous datasets, marginalisation is integration and the joint probability distribution is often not integrable. In such cases, however, the expected value can still be successfully estimated using either variational methods (Jordan et al., 1999), although the computational cost can often be significant. In discrete datasets where integration is summation, the situation can be even more difficult since the number of summations is at least exponential in the number of discrete variables. In this thesis we therefore follow Segal et al. (2003) in implementing an approximate EM algorithm which is guaranteed to offer improvements on the basis of *all* variables still, but is not guaranteed to locate a locally optimal solution.

### **2.3 In Brief: Dynamically Adding and Deleting Motifs**

In Segal et al. (2003) the authors decide to give their training scheme the flexibility of adding or deleting Regulation motifs. The objective is to bring more motifs under consideration when the currently chosen set is poorly correlated with the expression data and also delete motifs that do not contribute sufficiently to the training algorithm. Changing the number of variables in a probabilistic model is a *structural* intervention, which cannot be readily incorporated in our training scheme, since it effectively changes the model, as opposed to changing its parameters. The authors of Segal et al. (2003) offer a heuristic manner to regulate the process of dynamically adding and deleting motifs. The validity and semantics of this method are explored at a later chapter.

## 2.4 A Note on the Mathematics

When possible we attempt to ground intuition and observation on firmer theoretical grounds. However, since this thesis concerns an academically mixed audience, we have been concerned with attempting to decouple the mathematics from our conclusions and observations. We chose not to separate the mathematical approach from the descriptive approach, so as to ensure the relevance of the former and the validity of the latter can be seen at all times. Instead, we have made every effort to ensure the mathematics are not necessary to follow the argumentation of this thesis. To this effect, we consciously attempt to accompany each paragraph of maths by a paragraph describing *exactly the same point* in words. This has been mostly possible, since our mathematical arguments are simple enough to be rephrased in terms of natural geometrical intuitions.

# Chapter 3

## Motif Model

### 3.1 Formulating our Objective

At the level of the motif model, we wish to predict coregulation on the basis of promoter sequence data. However, we are not particularly concerned with explicitly modelling the promoter sequence nor the transcription factor itself. Moreover, since the hidden variables are dealt with in a unified manner, they are considered *given* at the level of the motif model. These considerations reduce the problem to the following *supervised* classification task:

**Objective 1** *Given the promoter sequences of a set of genes  $G_1$  that are known (or assumed) to be regulated by a certain transcription factor  $T$ , a set of genes  $G_0$  that are known (or assumed) not to be regulated by  $T$  and a novel set of genes  $G'$ , classify each gene in set  $G'$  as 'regulated by  $T$ ' or 'not regulated by  $T$ '.*

The supervised nature of this task makes it significantly different than the standard approaches to motif finding. In general, the sets  $G_1$  and  $G_0$  are not explicitly given. The only option then is to concentrate on a set  $G'$  suspected of coregulation and find the motif that has the strongest presence there. In contrast, in our model the clustering of genes in coregulated groups is iteratively refined in a unified manner, involving more information than the sequences alone, and the motif model is always trained with respect to one of these tentative clusterings. For an overview of self contained approaches to motif finding the reader is invited to consult Jensen et al. (2004).

Our supervised classification task can be broken down as follows:

- Choose an efficient and versatile representation of a motif.
- Construct a way to decide on the presence or absence of a given motif in a given promoter sequence.
- Determine which binding site  $A$  is most likely to be present in most genes in  $G_1$  and absent in most genes in  $G_0$ .
- Classify each gene in  $G_1$  as 'regulated by  $T$ ' if  $A$  is most likely to be present in its promoter sequence of  $g$  or 'not regulated by  $T$ ' otherwise.

We proceed to demonstrate a probabilistically coherent way of dealing with this task.

## 3.2 Representing Motifs

Our representation of motifs cannot afford to be rigid. Transcription factors are often capable of binding to several variations on a motif.

Deterministic models understand this problem as one of deciding how much tolerance we ought to have with respect to deviations from a 'correct', *consensus* motif. Such approaches usually employ the *Hamming distance* as a measure of deviation between two motifs, which is simply a count of the mismatches between them. Only two mismatches are usually allowed in representations of this sort.

This approach models the biology very poorly. There seems to be no simple way of relating the mismatches between two motifs to the respective probabilities of binding: since the binding is a result of chemical interactions between the nucleotide sequence and the transcription factor, the role of a mismatch in inhibiting or promoting binding depends in a complicated manner to neighbouring nucleotides as well as the overall configuration of the transcription factor.

Probabilistic representations of motifs model this situation more aptly, acknowledging the uncertainty and attempting to model the probability of each possible configuration.

Position Specific Scoring Matrices (PSSMs) are the simplest probabilistic representation of motifs. They model the uncertainty at each position of the motif and ignore any possible interdependencies between the positions. Taking a step towards the direction of greater complexity, we can let the nucleotide at each position depend on the previous position. This model is known as 1st order Markov Chain. In general, an  $n$ 'th order Markov Chain model conditions the value at each position at the  $n$  immediately previous values. A PSSM therefore corresponds to a 0 order Markov Chain. Allowing for more sparse interactions, Bayesian Networks can be used along with Structural Learning algorithms to learn both the probability distributions and the interdependencies from the data. Finally, mixtures of Markov Chains that seek to model each motif on the assumption that it was sampled from one of several potential distributions can be very efficient in practice, since it optimally models local variations from a few global templates, which seems to correspond to the sensitivities of the chemical process. Graphical representations of these models can be seen in Figure 3.2.

For simplicity and to follow Segal et al. (2003) we make use of the PSSM representation in this thesis. We have not formally explored the effect of this choice on the performance of our model, which is regrettable since some characteristics of the model crucial to our discussion involve features of the motif model that certainly depend on the complexity of the representation of a motif. A general comparison of models of varying complexity is given in Barash et al. (2003).

### 3.3 Motivating and Formulating the Model

Let  $M_i$  indicate the  $i$ 'th position of the motif and  $S_j$  indicate the  $j$ 'th position of the promoter sequence. Assume we are given the PSSM of a certain motif of length  $p$ . This would be given as a set of  $4p$  parameters, representing the probability of each nucleotide appearing in each position. Therefore, for each position  $j$ , we have a probability distribution over the nucleotides:

$$\psi_i(B) = P(M_i = B), \quad B \in \{A, C, G, T\} \quad (3.1)$$

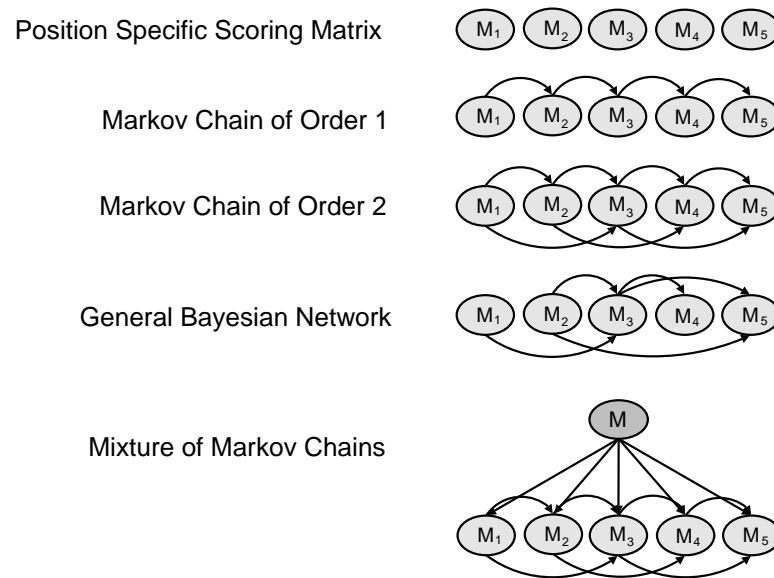


Figure 3.1: Graphical Representations of Motif Models. Each numbered node represents a motif position and the arrows indicate modelled dependencies between the positions. The extra node in the Mixture Model labelled as 'M' is also discrete and represents the choice of probability distribution each motif is considered a sample of.

Assuming independence between the positions, the probability of certain subsequence of length  $p$  being an instance of the motif in question cannot be evaluated using 3.1 only, since we need to contrast the PSSM to what a subsequence that is *not* a motif looks like. As yet, we can evaluate the probability of an instance of a motif looking like our subsequence, i.e. the probability of our subsequence *given* that it is a motif:

$$P((S_{i+j-1})_{j=1}^p \mid \text{subsequence is a motif}) = \prod_{j=1}^p \psi_j(S_{i+j-1}) \quad (3.2)$$

We now wish to evaluate the probability of the entire sequence  $g.S$  given that it is regulated by the respective transcription factor and hence contains a motif sampled from the PSSM  $\psi$  at an arbitrary position  $k$ . To do that, we need to assume a background model  $\theta$  for the subsequences that are *not* assumed to be instances of the motif. We assume a position-independent background model. We now have the expression we want:

$$P(S \mid R = \text{true}, \text{binding position} = k) = \left( \prod_{\substack{i=1 \\ i \notin \{k, \dots, k+p-1\}}}^n \theta(S_i) \right) \left( \prod_{j=1}^p \psi_j(S_{k+j-1}) \right) \quad (3.3)$$

which simplifies to the expression:

$$P(S \mid R = \text{true}, \text{binding position} = k) = \prod_{i=1}^n \theta(S_i) \prod_{j=1}^p \frac{\psi_j(S_{k+j-1})}{\theta(S_{k+j-1})} \quad (3.4)$$

Since we are unaware of the binding position we can obtain the conditional probability of the entire sequence by marginalising over all possible binding positions. This is easy since the parameters in the expression 3.4 are independent of the binding position  $k$ . The marginalisation we ought to make is given by:

$$P(S \mid R = \text{true}) = \sum_i P(S, \text{binding at } i \mid R = \text{true}) \quad (3.5)$$

By Bayes' theorem we can rewrite this as<sup>1</sup>:

$$P(S \mid R = \text{true}) = \sum_{i=1} P(S \mid R = \text{true}, \text{binding at } i) P(\text{binding at } i) \quad (3.6a)$$

---

<sup>1</sup>In fact, we are also assuming that the binding position is independent of the  $R$  variable, which is justified since the background model we will present shortly is position-independent

$$P(S_1, S_2, \dots, S_n | R = true) = \sum_{i=1}^{n-p+1} \frac{P(S | R = true, \text{binding at } i)}{n-p+1} \quad (3.6b)$$

Finally, we need to invert the direction of the conditioning, since we are interested in the probability that the gene is regulated by a certain transcription factor, given the sequence. To do that we use Bayes' theorem again:

$$\begin{aligned} P(R = true | S) &= \frac{P(S, R = true)}{P(S)} \\ &= \frac{P(S, R = true)}{P(S, R = true) + P(S, R = false)} \\ &= \frac{P(S | R = true)P(R = true)}{P(S | R = true)P(R = true) + P(S | R = false)(1 - P(R = true))} \end{aligned} \quad (3.7)$$

This can be recast to the simpler form:

$$P(R = true | S) = \text{logit}(x) \quad (3.8)$$

where  $x$  is given by:

$$\begin{aligned} x &= \log\left(\frac{P(S, R = true)}{P(S, R = false)}\right) \\ &= \log\left(\frac{P(R = true)}{P(R = false)} \frac{1}{n-p+1} \sum_{j=1}^{n-p+1} \prod_i^p \frac{\psi_i(S_{i+j-1})}{\theta(S_{i+j-1})}\right) \end{aligned} \quad (3.9)$$

and  $\text{logit}(x)$  is the function given by:

$$\text{logit}(x) = \frac{1}{1 + e^{-x}} \quad (3.10)$$

This completes our formulation of the probability distribution.

The inversion of the direction of conditioning is crucial in determining the discriminative nature of this model. The authors of Segal et al. (2003) express this succinctly in a previous related paper(?), where they contrast this approach to *generative* approaches:

[Other approaches are] *generative*, in the sense that they try to build a model of the promoter region sequence, and training succeeds when the model gives the given sequences high probability. However, these approaches can often be confused by repetitive motifs that occur in many promoter sequences. These motifs have to be filtered out by using an appropriate background distribution.

By focusing on the probability of regulation only, the only data that are relevant in determining the parameters of the model are the sets  $G_1$  and  $G_0$ . The model then will automatically *filter out* highly repetitive motifs since learning such motifs would have the undesirable effect of yielding equally high probabilities to genes in either set.

### 3.4 Reparameterising the Model

We are therefore able to perform the classification task outlined in the beginning by learning the setting of the parameters of the background distribution  $\theta$  and the PSSM  $\psi$  that maximally discriminates between the sets  $G_1$  and  $G_0$ . This corresponds simply to setting  $g.R$  to be *true* whenever  $g$  is in  $G_1$  or 0 otherwise and then computing the Maximum Likelihood Estimates of the parameters. The maximum likelihood estimates of  $P(R = \textit{true})$  and  $\theta$  are closed form, but the parameters of  $\psi$  attain no closed form and have therefore to be approximated by gradient descent.

However, we follow the authors of Segal et al. (2003) in adopting a different parameterisation than the one readily suggested by the model. First we code the nucleotides by the convention that  $A \equiv 1$ ,  $C \equiv 2$ ,  $G \equiv 3$ ,  $T \equiv 1$ . We introduce a  $4 \times p$  matrix of *weights* and a separate parameter  $\alpha$ , which we call the *threshold*. These are related to the model as follows:

$$w_{ij} = \log \left( \frac{\psi_j(M_j = i)}{\theta(M_j = i)} \right) \quad (3.11)$$

and

$$\alpha = \log \left( \frac{P(R = \textit{true})}{1 - P(R = \textit{true})} \right) \quad (3.12)$$

Observe that this is a *complete* parameterisation of the model. Also note that the authors of Segal et al. (2003) still use the  $w_{ij}$ 's as the parameters of the PSSM. In this paper we alternate between a strict understanding of a PSSM as a probability distribution and a loose one where no normalisation constraints are in place. It is always clear from the context which of the two is referred to.

### 3.4.1 The Semantics of the Weights

The benefits of switching to this parameterisation are directly verified in testing but are hard to investigate formally. In specific, the model seems to be more free than before, but not in a theoretically interpretable way. The authors of Segal et al. (2003) are not concerned with attempting this interpretation and therefore rightfully evade the issue altogether by describing their model as a *standard binary logistic model* and omitting the derivation we presented in the previous section. However, in the previously cited paper by the same group, an explanation was hinted at when the background distribution was first introduced:

For simplicity, we use a Markov process of order 0 for the background distribution. (As we will see, the choice of background model is not crucial in the discriminative model we develop).

Indeed it is true that this parameterisation is complete for all 0 order Markov background distributions, as we have demonstrated above. However, this passage seems to suggest that by reparameterising and learning the model we are effectively optimising over *all possible* background probability distributions. In other words, the reparameterised model withholds its probabilistic semantics without the need to explicitly define a functional form for the background distribution. This implication is in fact strictly false since the reparameterisation holds *only* for a 0 order background distribution. Any dependence on other positions would make the probability of a sequence containing a motif dependent on the binding position, hence causing the derivation from 3.6b to 3.7 to fail and the semantics to consequently collapse.

However, given any assignment of the  $w_{ij}$ 's that satisfies a simple criterion, it is possible to find a *0 order background distribution* and a PSSM that give rise to exactly the same model as our reparameterised version. The condition is the following:

**Statement 1** *Statement For the weights to be recastable to the form:*

$$w_j(i) = \log \frac{\Psi_j(i)}{\Theta(i)} \quad (3.13)$$

where  $\theta$  and  $\psi_j$ , for each  $j$ , are normalised probability distributions over  $i \in \{1, 2, 3, 4\}$ , it is necessary and sufficient that the following holds for all  $j$ :

$$\mathbf{either} \quad w_j(i) = 0 \quad \text{for all values of } i \quad (3.14)$$

$$\mathbf{or} \quad \min_i(w_j(i)) < 0 \quad \mathbf{AND} \quad \max_i(w_j(i)) > 0 \quad (3.15)$$

A short proof of this statement is included in the appendix. There is no natural way to enforce this constraint during training, but all tests indicated that the gradient descent method only searches portions of the parameter space where this condition holds.

The decomposition above is evidently *never* unique, unless a 0 order background prior is specified. It could be argued that this sheds doubt on the validity of the probabilistic semantics defended so far. We will not argue this point, for lack of time. We ought to mention however that such doubts could arguably be further supported by the fact that in a total of 20 complete tests of this model on synthetic data of varying specifications, the model never produced a weight matrix that was decomposable to the true background prior and the true PSSM.

In specific, a test we performed early in our investigation was to contrast the performance of the motif model with and without a normalisation constraint on the weights on each position corresponding to restricting the model in assuming a uniform prior. This constraint is given by:

$$\theta(k) = 1/4 \forall k \quad (3.16)$$

which implies that:

$$\frac{\psi_j(k)}{\theta(k)} = 4\psi_j(k) \quad \Rightarrow \quad (3.17)$$

$$\sum_{k=1}^4 e^{w_j(k)} = 4 \sum_{k=1}^4 \psi_j(k) \quad \forall j \quad (3.18)$$

$$\sum_{k=1}^4 e^{w_j(k)} = 4 \quad \forall j$$

This evidence is highly suggestive but inconclusive, since the likelihood surface is highly irregular with many local optima and there is also an overfitting effect, that is discussed in the next section. What *would* be conclusive evidence as to the validity or invalidity of the probabilistic semantics? We speculatively suggest that to establish

beyond doubt the status of the probabilistic semantics of the reparameterised version, one ought to prove that the point of highest likelihood for the reparameterised model is always decomposable to the highest likelihood background prior and the highest likelihood PSSM. This would mean the two models are identical in a maximum likelihood sense. Since most of these parameters cannot be estimated in closed form, no direct way exists of arguing either side and therefore settling this point falls beyond the scope of this thesis.

### 3.4.2 The Semantics of the Threshold

The final parameter of our model is the threshold,  $\alpha$ . This analytically corresponds to  $\log \frac{P(R=true)}{P(R=false)}$ . The above difficulties in justifying the probabilistic semantics of the reparameterised model do not directly apply on the threshold, since the parameter  $\alpha$  is not affected by the choice of the joint distribution over  $R$  and  $S$  in either account.

Testing on synthetic data seems to suggest the threshold is in fact in close relation to its probabilistic counterpart. Observe that, when the data is *truly* sampled from a 0 order background distribution and a PSSM, the Maximum Likelihood Estimate of  $\log \frac{P(R=true)}{P(R=false)}$  attains the closed form  $\frac{\text{Number of genes in } G_1}{\text{Number of genes in } G_2}$ . In synthetic datasets where that was the case, the learnt value of  $\alpha$  precisely coincided with this analytical estimate when the rest of the weights were assigned to their true values. In contrast, when there is noise or the generating process is higher order and cannot possibly be accurately captured by this model, as well as when the rest of the weights are given a locally optimal assignment, the threshold is generally expected to be significantly *lower* than the analytically computed value, indicating the lower confidence of the model in its ability to classify correctly. This was verified in all cases during testing.

## 3.5 Evaluation and Overfitting

We have performed a series of tests to establish the discriminative capacities of our model. We have used the netLab implementation of conjugate gradient descent to learn

the parameters on the following list of synthetic datasets. The motif length is always 6 and the default specifications otherwise are 150 datapoints per class, of length 100.

1. 150 datapoints of length 100 each in  $G_1$  and 150 datapoints of equal length in  $G_0$
2. As above but motifs are sampled from a different PSSM
3. As above but motifs are sampled from a different PSSM
4. The length of the sequences is increased to 1000. The PSSM is the same as in dataset 1.
5. An imbalanced dataset of otherwise default specifications, where  $G_1$  has 2 times more elements than  $G_0$ .
6. An imbalanced dataset of otherwise default specifications, where  $G_1$  has 5 times more elements than  $G_0$ .
7. A dataset of default specifications where the background distribution is 0 order non-uniform.
8. A dataset of default specifications with 20% of the bases randomly flipped.
9. A dataset of default specifications where sequences in  $G_0$  have motifs that are sampled from a PSSM similar<sup>2</sup> to the class 1 PSSM.
10. A dataset of default specifications where both the motif and the background data are generated from 1 order Markov Chains
11. The sequences here are the same as in dataset 1, but 30% of the datapoints in  $G_1$  are switched with datapoints from  $G_0$ .
12. The sequences here are the same as in dataset 1, but 20% of the datapoints that truly lie in  $G_1$  are misplaced in  $G_0$  instead.

---

<sup>2</sup>Similarity is measured by the KL divergence.

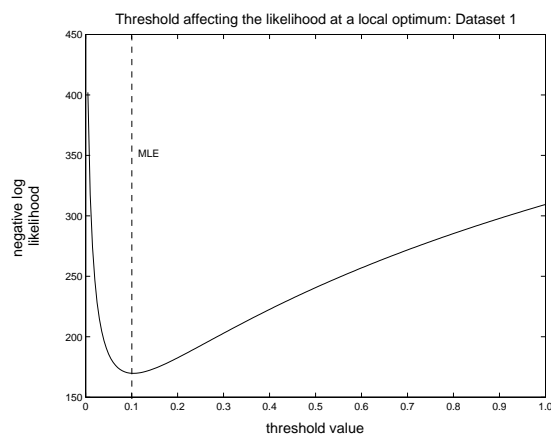


Figure 3.2: Dataset 1

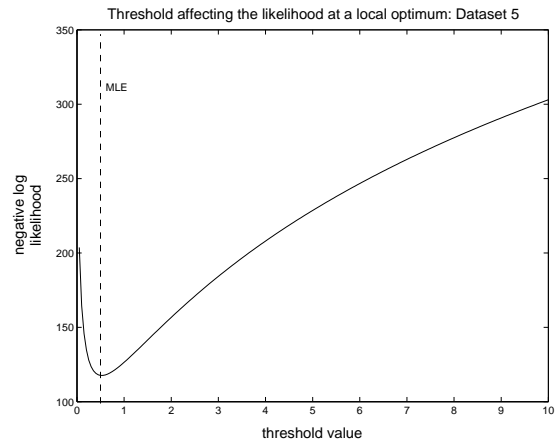


Figure 3.3: Dataset 6

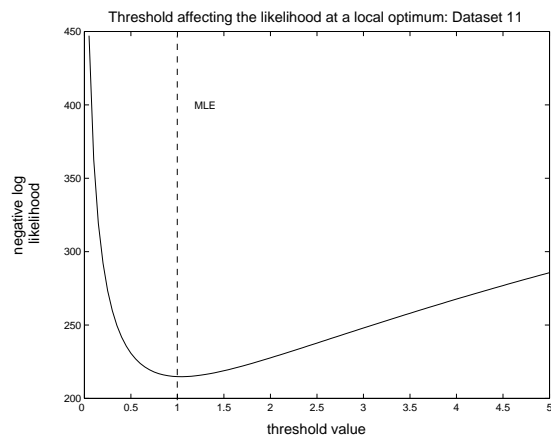


Figure 3.4: Dataset 11

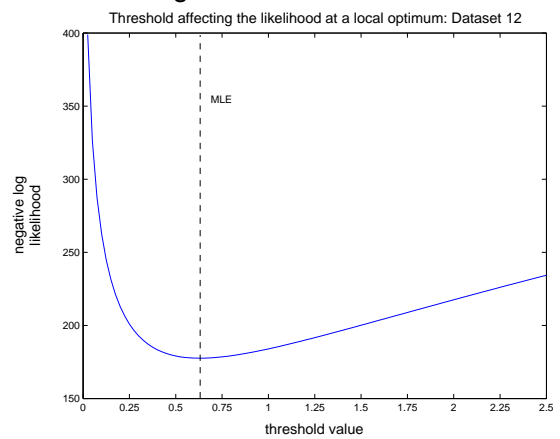


Figure 3.5: Dataset 12

### 3.5.1 Testing the semantics of the threshold

We start by testing the coincidence of the analytically computed MLE for  $\alpha$  with the numerically estimated ML value (Figure ??) at the *true assignment of the weights*, which is computed using our knowledge of the generating background distribution and the generating PSSM. We test neighbouring values to the learnt one to ensure the solution is locally unimodal so that the coincidence can be deemed significant.

Dataset	$\frac{P(R=true)}{P(R=false)}$	$\alpha$ learnt	$\alpha$ learnt
		true weights	local optimum
1	1	1	0.0319
2	1	1	0.156
3	1	1	0.0990
4	1	1	0.1353
5	0.5	0.5	0.012
6	0.2	0.2	0.018
7	1	1.1	0.0683
8	N/A	1.349	0.0638
9	N/A	1	0.0647
10	N/A	N/A	0.2831
11	1	1	0.165
12	0.66	0.60	0.0471

Table 3.1: We compare the learnt value of the threshold  $\alpha$  and the analytically computed MLE of its probabilistic counterpart  $\frac{P(R=true)}{P(R=false)}$ . The values coincide as expected for these datasets, since the true generating process is within the capacity of our model. Moreover, the coincidence is significant since it occurs at the only mode of the likelihood surface with respect to the threshold at the neighborhood of the true assignment of the weights. Comparison is made for all datasets where the true value is known.

Dataset	LL of learnt $\mathbf{w}$	LL of true $\mathbf{w}$
1	163.59	169.74
2	194.19	171.74
3	183.92	198.34
4	592	599
5	66.15	67.81
6	113.36	117.8
7	120.44	139.57
8	187.7662	204.82
9	168.45	174.29
10	N/A	106.44
11	194.02	214.82
12	170.93	177.65

Table 3.2: Ability of the model to learn motifs that are as likely as the true structure in synthetic data.

### 3.5.2 Assessing the Model

We now turn our attention to the learning of the rest of the weights. For now we concentrate on the ‘base’ generating model, which is represented by the first 4 datasets. We contrast in table 3.2 the negative log likelihood of the learnt solution with the likelihood of the true assignment. In several of the datasets above we observe that the learnt weights yield even lower likelihoods than the true assignment of the weights. In datasets 8 and 9 such a difference is expected since the true weights are no longer strictly true, since the noise and the background motifs are not modelled by our probability distribution. However, in the remaining datasets, this feature indicate an overfitting effect. Since, as we have seen, the threshold indicates the classification confidence, it is appropriate to investigate overfitting effects using *ROC curves*.

A ROC curve measures training performance, but in a different way to the training error or the likelihood. The ROC curve is a plot of the *sensitivity* against *1-specificity* for several different values of the threshold (the classification confidence parameter, in general). The sensitivity and specificity are defined in terms of classification hits and

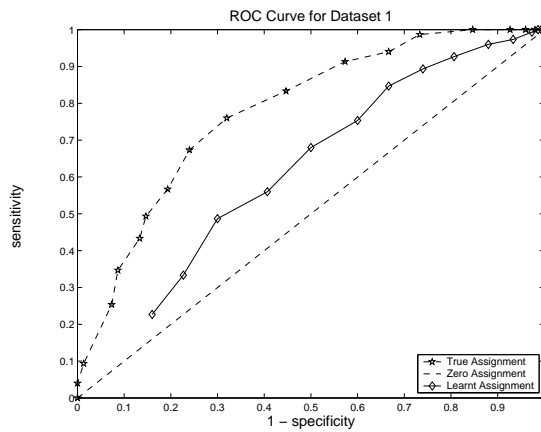


Figure 3.6: ROC curve for Dataset 1

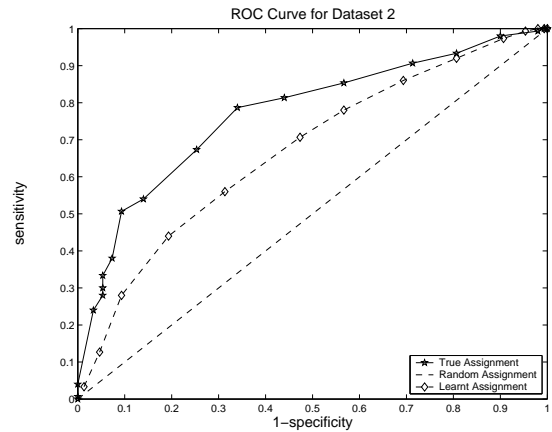


Figure 3.7: ROC curve for Dataset 2

misses as follows:

$$\text{sensitivity} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{specificity} = \frac{\text{True Negatives}}{\text{True Negatives} + \text{False Positives}}$$

The interesting feature of this graph is the *area under the ROC curve*, which is thought to be a good predictor of test performance (Hanley and McNeil, 1982). For instance, given two different assignments of the weights that have the same likelihood, the one that encloses the largest area under its ROC curve is probably going to perform best in test datasets.

We present the ROC curves with respect to datasets 1 and 2 (Figures 3.6 and 3.7 respectively). We observe that indeed the true assignment of the weights encloses the largest area in both cases, which indicates that it will probably be the best predictor of previously unseen sequences.

### 3.5.3 Overfitting, Self-Similarity and Multiple Motifs

An interesting feature of 3.6 and 3.7 is

# Chapter 4

## Regulation Model

The objective of this model is to cluster the data on the basis of the information obtained from the previous layer about the absence or presence of motifs for each gene. The clusters are called "transcriptional modules" and are mutually exclusive and exhaustive.

Genes belonging to the same module are said to have *similar motif profiles*. This roughly means that they are meant to have most motifs in common (which is evidence of coregulation). In making this measure of similarity precise the main challenge is *interpretability*. On the one hand, there are decisions we need to make concerning the hypotheses of our model with respect to the biology. For instance, can the upregulation of a gene be dependent on the absence of a certain motif from its promoter sequence? Can the simultaneous presence of two motifs be strong evidence of coregulation when the individual presence of either motif is insignificant? It is then necessary to construct a mathematical model that can codify our hypotheses along with the questions we are asking and relate them in a *non-biased, uniquely interpretable* way to the data, remaining computationally tractable. This is rarely if ever fully possible. It is then proper to explicitly relax the hypothesis in accordance with the discriminative capacity of the model, so as to ensure the results remain interpretable. For this to happen, both the hypothesis and the model must be understood explicitly and to the detail.

The reader will observe that this section is the richest and longest of the thesis. This reflects its relative significance. The main import of the model under scrutiny is not

the individual modelling of promoter sequences and expression data but rather their joint consideration in a coherent manner. This 'bringing together' mainly occurs at the regulation level.

## 4.1 Formulation of the Model

The model we are using belongs to the class of General Linear Models and is known as the multinomial or polychotomous logistic model in statistics. It corresponds to the multivariate softmax as a choice of transfer function in neural networks. Let  $L$  be the number of motifs we wish to consider. We view the multivariate softmax as a probability distribution over a discrete variable  $M \in \{1, \dots, K\}$ , which represents the module assignment, conditional on a set of continuous parameters given in the form of a  $K \times L$  matrix  $\mathbf{u}$ , and a multidimensional variable  $\mathbf{R} \in \mathbb{R}^L$ , which represents the input. Under this model, the probability of a datapoint  $\mathbf{r}'$  belonging to module  $m'$  is given by:

$$P(M = m' | \mathbf{R} = \mathbf{r}') = \frac{\exp\{\sum_{i=1}^L u_{m'i}r'_i\}}{\sum_{m=1}^M \exp\{\sum_{i=1}^L u_{mi}r'_i\}} \quad (4.1)$$

The input  $r$  in [1] is taken to be strictly binary, with  $r_i = 1$  when a motif is present or 0 otherwise. However, each component of our input variable  $\mathbf{R}$  can in principle take any real value. This yields some flexibility in choosing how to feed in the information from the previous layer.

This information naturally comes as a vector of probabilities  $\mathbf{p} = [p_1, p_2, \dots, p_L]^T$  where  $L$  is the number of motifs under consideration at that point in the algorithm and  $p_i = P(\text{motif } i \text{ is present})$ . Therefore, we could have alternatively used the vector  $\mathbf{p}$  of probabilities itself as input, corresponding to taking the expected value of the  $\mathbf{r}_i$ 's with respect to the probability distribution of the previous layer. Since this hints at the EM training framework, we call this format a *soft* assignment of the variables. For the purposes of the thesis, we keep the values discrete for testing in this and the following chapter, but during unified training we switch to soft input.

Another consideration is *symmetry*. Rescaling the soft binary input values by 2 and translating them by -1 yields a symmetric input format, where  $r_i = -1$  indicates certain absence of the  $i$ 'th motif,  $r_i = 1$  certain presence and  $r_i = 0$  represents ignorance.

Although it appears as mere relabelling, the choice between asymmetric and symmetric input affects the performance of the model drastically as can be seen in the results section appended at the end of the next chapter.

## 4.2 Geometrical Interpretation of the Model

A firm geometrical interpretation of a statistical model is a very important tool, for two reasons. Firstly, the geometrical interpretation yields an intuitive grasp of the model to the non-statistician, who can then bypass the mathematics and decide whether the model fits their understanding of the domain in much friendlier terms. Secondly, to the statistician, the geometrical interpretation makes available a wealth of mathematical methods, results and intuitions which can be brought to bear to the training of the model and hence better explain its performance. We proceed to outline the geometrical counterpart of the multilogit classification model.

### 4.2.1 Binary Classification via Separating Hyperplanes

It is general practice in supervised clustering techniques to embed the data in a Euclidean<sup>1</sup> space of  $L$  dimensions (where  $L$  is the number of attributes/variables) and attempt to partition  $\mathbb{R}^L$  into clusters by means of hyperplanes, so that each cluster contains datapoints of the same class only.

What is a hyperplane? In 2 dimensions, a hyperplane is a line, whereas in 3 dimensions it is a plane. The generalisation to higher dimensions is obtained by generalising the concept of a plane in three dimensions or a line in two. A particularly simple way to do it is to specify a vector  $\mathbf{w}_H$  perpendicular to the hyperplane and a point  $\mathbf{b}_H$  contained in it (fig 4.1). To understand this, consider any point  $\mathbf{x}$  contained in the hyperplane  $H$ . The difference  $\mathbf{x} - \mathbf{b}_H$  geometrically corresponds to the hypotenuse of the triangle formed by the two points and the origin, which can be easily seen to be entirely contained in  $H$ . As a result, it has to be perpendicular to  $\mathbf{w}_H$ , which means that their inner

---

<sup>1</sup>Specifying the metric is in fact not necessary so long as our space is an inner product space. In Support Vector Machines, for instance, the classification occurs in a non-Euclidean space.

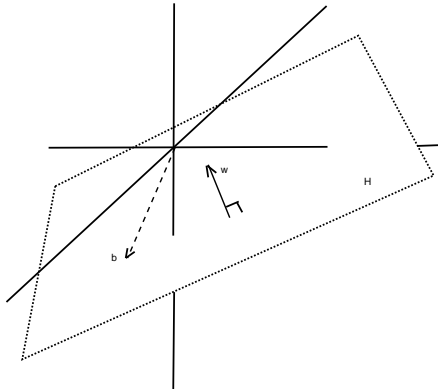
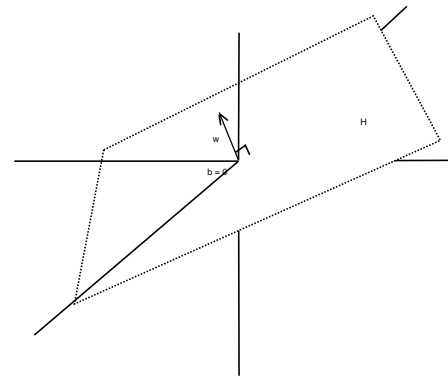


Figure 4.1: A general plane

Figure 4.2: The case  $\mathbf{b} = \mathbf{0}$ 

product must be 0, and hence:

$$\mathbf{x} \text{ belongs to the hyperplane } \mathbf{H} \text{ if and only if } \mathbf{w}_{\mathbf{H}} \cdot (\mathbf{x} - \mathbf{b}_{\mathbf{H}}) = 0 \quad (4.2)$$

Conversely, then, any points not contained in  $\mathbf{H}$  can be classified according to the sign of the inner product  $\mathbf{w}_{\mathbf{H}} \cdot (\mathbf{x} - \mathbf{b}_{\mathbf{H}})$ . In this sense, a high dimensional hyperplane still separates the hyperspace in two "sides", similarly to a line in two dimensions or a plane in three.

In binary classification tasks, the objective is to find a hyperplane such that most datapoints of class 0 are located on one side of the hyperplane and most datapoints of class 1 in the other.

#### 4.2.2 Multiclass Classification via maximisation of the inner product

In our case, the classification criterion is more complex and involves not merely the sign of the inner product, but also its modulus. We can readily recover it from the probability distribution. Assume we want to classify a novel point  $\mathbf{r}$ . We observe that, if  $m'$  is the most probable module assignment, then for all  $m \neq m'$  we have:

$$P(M = m' | \mathbf{R} = \mathbf{r}, \mathbf{u}) > P(M = m | \mathbf{R} = \mathbf{r}, \mathbf{u}) \quad (4.3)$$

Substituting in the expression 4.1 for the probability distribution and cancelling out the normalisation constraint which is independent of  $m$  we get:

$$\exp\left\{\sum_{i=1}^L u_{m'i}r_i\right\} > \exp\left\{\sum_{i=1}^L u_{mi}r_i\right\} \quad (4.4)$$

Since the exponential is a strictly monotonically increasing function it can be ignored from both sides of the inequality. Letting  $\mathbf{w}^{(k)}$  denote the  $k$ 'th row of  $\mathbf{u}$  (not to be confused with the parameter set of the motif model) we can write this as:

$$\mathbf{w}^{(m')} \cdot \mathbf{r} > \mathbf{w}^{(m)} \cdot \mathbf{r} \quad (4.5)$$

Therefore, the most probable module assignment is given by:

$$m' = \underset{m \in \{1, \dots, K\}}{\operatorname{argmax}} \mathbf{w}^{(m)} \cdot \mathbf{r} \quad (4.6)$$

To summarise, given a novel datapoint  $\mathbf{r}$ , we need to find the row of  $\mathbf{u}$  whose inner product with  $\mathbf{r}$  is biggest.

We can consider the rows of  $\mathbf{u}$  as specifying a set of hyperplanes  $H_1, H_2, \dots, H_K$ , all containing the origin (fig 4.2)<sup>2</sup>. It is then evident that if a datapoint lies at the positive side of one hyperplane only, then it is assigned the label of that hyperplane. The criterion then is the sign of the inner product only. However, the sign of the inner product is not sufficient when the datapoint lies at the negative side of all hyperplanes or at the positive side of more than one hyperplanes. The modulus itself then comes into play. The modulus of the inner product  $\mathbf{r} \cdot \mathbf{w}^{(m)}$  is closely related to the perpendicular distance between  $\mathbf{H}_m$  and  $\mathbf{r}$ . In specific, if the hyperplane  $H$  is given by the unit modulus vector  $\mathbf{e}$ , the perpendicular distance between  $\mathbf{H}$  and  $\mathbf{r}$  is equal to  $(\mathbf{r} \cdot \mathbf{e})^2$ . Therefore:

**Statement 2** *Assuming that all rows of  $\mathbf{u}$  have modulus 1, the choice of hyperplane that lies closest to a given datapoint  $\mathbf{r}$  is also the one that minimises the square of the inner product  $\mathbf{r} \cdot \mathbf{w}^{(m)}$  and vice versa.*

There are two differences in our case: firstly, the rows of  $\mathbf{u}$  do not necessarily have unit modulus and, secondly, we are minimising the inner product itself, as opposed to its

---

<sup>2</sup>Following the terminology of the previous section, let  $\mathbf{w}_H = \mathbf{w}^{(m)}$  and  $\mathbf{b}_H = \mathbf{0}$ .

square.

We deal with the latter difference first. Statement 2 directly implies that minimising the inner product corresponds to *minimising* the distance when  $\mathbf{r}$  lies at the *positive* side of  $\mathbf{H}_m$  and *maximising* the distance when  $\mathbf{r}$  lies at the *negative* side of  $\mathbf{H}_m$ . This is in fact easily interpretable. When a datapoint lies at the positive side of several hyperplanes, the model gives it the label of the hyperplane whose negative side is furthest away from the datapoint. Conversely, when a datapoint lies at the negative side of all hyperplanes, the model will "do its best" and give it the label of the hyperplane whose positive side is nearest.

The former difference can also be explained away. Indeed, the rows of  $\mathbf{u}$  do not in general have unit modulus. However, this only means that the decision boundary formed by any two hyperplanes  $\mathbf{H}$  and  $\mathbf{H}'$ , rather than being the set of points equidistant to both hyperplanes, is instead the set of points whose distance from  $\mathbf{H}$  is  $q$  times the distance from  $\mathbf{H}'$ , where  $q$  is the ratio  $\frac{|w|}{|w'|}$ . Crucially, the decision boundary remains linear (ie also a hyperplane) and is merely pivoted through the intersection of the two hyperplanes by an appropriate angle (figure 4.3).

### 4.2.3 The Shape of the Multiclass Decision Boundary

In the binary case, the overall decision boundary forms a hyperplane containing the origin, which acts as a single separating hyperplane as described in the previous section (Figure 4.4). In learning a binary logistic classifier, we can then bypass the modulus of the inner product and train the model to learn the separating hyperplane immediately, without resort to two hyperplanes as shown here<sup>3</sup>

In the multiclass case, the overall decision boundary is formed by appropriately linking together segments of the "pairwise decision boundaries" - ie the hyperplanes forming the decision boundaries between any pair of hyperplanes (Figure 4.5). Crucially, *no* segment of the hyperplanes represented by  $\mathbf{u}$  themselves *ever* forms part of the decision

---

<sup>3</sup>In fact, due to the invariance properties that are discussed later, we are free to fix one of the two hyperplanes independently of the data. By setting it to be equal to the trivial hyperplane, which is the origin itself, the two methods are brought to literally coincide.

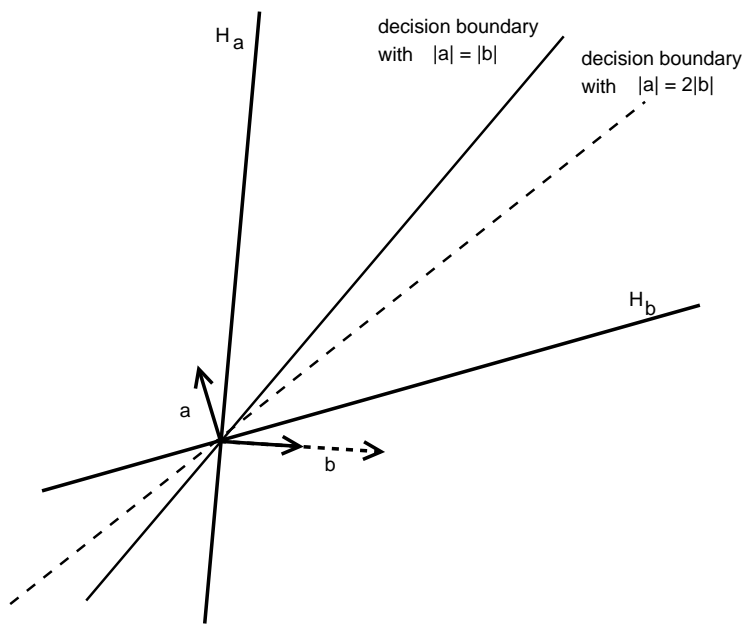


Figure 4.3: How the modulus of the rows of  $\mathbf{u}$  affects the decision boundary in 2 dimensions

boundary unless it coincides with a segment of a pairwise decision boundary.<sup>4</sup>

### 4.3 Maximum Likelihood Estimation of the parameters

The complexity of figure 4.5 clarifies that the task of locating the optimally positioned hyperplane with respect to a given training dataset is geometrically very difficult. The strength of the probabilistic model lies precisely in that it provides a way to numerically learn the optimal position.

The optimal setting of the parameters  $\mathbf{u}$  with respect to a given dataset and no prior knowledge is given by maximising the joint probability of the data given the parameters, which, when viewed as a function of the parameters, is called the *likelihood of the parameters with respect to the data*. Since we are assuming that the datapoints are

<sup>4</sup>This is immediate. Consider the hyperplane  $\mathbf{H}_1$  represented by  $\mathbf{w}^{(1)}$ . Now assume its segment  $S$  forms part of the decision boundary. Then all datapoints lying *near enough*  $S$  and at the positive side of  $\mathbf{H}_1$  will have a certain label and all datapoints lying near enough  $S$  at the other side will have another. So  $S$  is also a segment of one of the pairwise decision boundaries as claimed.

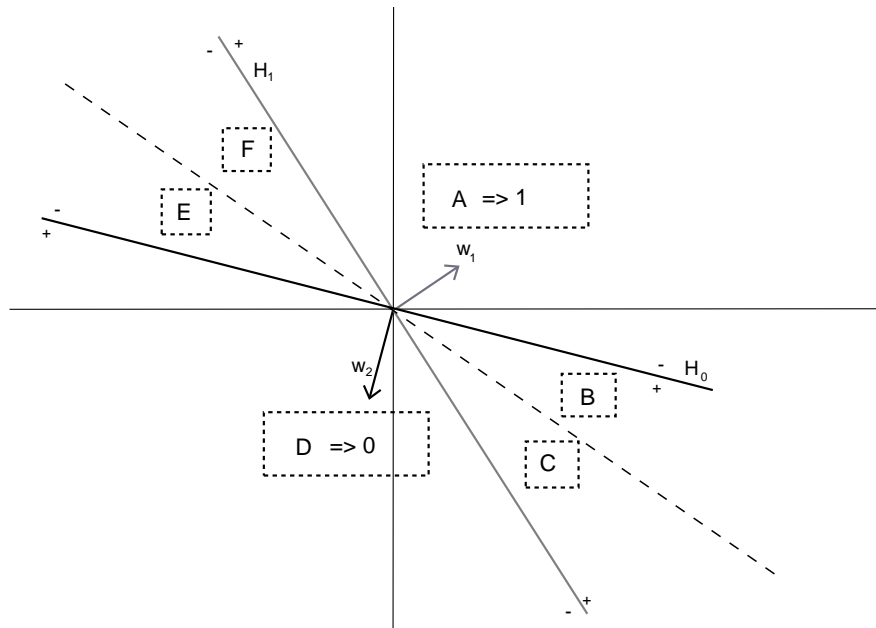


Figure 4.4: Binary Classification in 2 dimensions

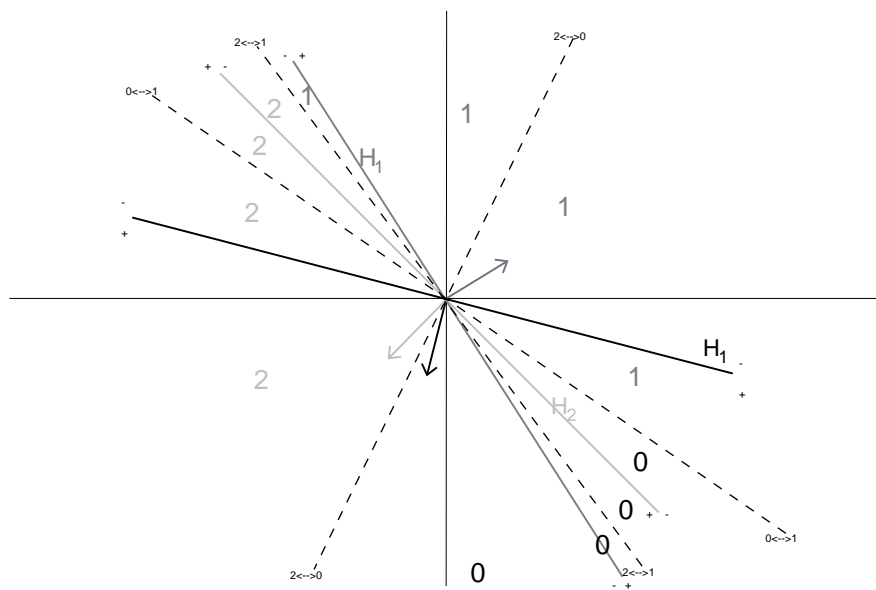


Figure 4.5: Multiclass Classification in 2 dimensions

independent and identically distributed, the likelihood term is given by:

$$\prod_{g \in D} \mathbf{P}(g.M \mid g.\mathbf{R}) \quad (4.7)$$

where  $g.M$  and  $g.\mathbf{R}$  denote the module and the regulation variables assignment respectively, of gene  $g$ . We then substitute in the probability distribution given in 4.1, which we repeat here:

$$P(M = m' \mid \mathbf{R} = \mathbf{r}') = \frac{\exp\{\sum_{i=1}^L u_{m'i}r'_i\}}{\sum_{m=1}^M \exp\{\sum_{i=1}^L u_{mi}r'_i\}}$$

We now wish to maximise 4.7 or equivalently minimise the computationally friendlier negative log likelihood:

$$\mathcal{L} = \sum_g [\sum_{i=1}^L u_{g.mi}g.r_i - \log(\sum_{m'=1}^K \exp\{\sum_{i=1}^L u_{m'i} \cdot g.r_i\})] \quad (4.8)$$

There is no analytical expression for the optimal assignment of the parameters with respect to 4.8. We therefore use gradient descent methods to learn the optimum. In this case gradient descent is guaranteed to locate the global optimum, provided one exists. This is because the expression is convex in the parameters, which can be taken together with the following statement (proof omitted) to conclude that the likelihood surface has no local optima.

**Statement 3** *If  $f(x)$  is convex and there exists a point at which:*

$$\frac{\partial f}{\partial x_k} = 0 \text{ for all } k, \quad (4.9)$$

*then  $f(x)$  has a minimum value at that point.*

Moreover, since 4.8 is everywhere differentiable, the only way for the function to fail to have a minimum is if the optimum is obtained at the limit to infinity (or negative infinity) of one or more of the parameters. We discuss such divergent behaviour in the next chapter.

To learn the optimal solution to 4.8, we employ *conjugate gradient* as our default descent method and substitute it with *quasi-newton* for cases where convergence is too slow. The gradient is given by:

$$\frac{\partial \mathcal{L}}{\partial u_{pq}} = \sum_{g \in D} [g.r_q [\mathbb{I}(g.m = p) - \mathbf{P}(M = p \mid \mathbf{R} = g.\mathbf{r})]] \quad (4.10)$$

where  $\mathbb{I}(g.m = p)$  is equal to 1 if  $g.m = p$  and 0 otherwise.

## 4.4 Directions of Invariance

It is straightforward to observe that if we add the same constant vector  $\mathbf{c}$  to each  $\mathbf{w}^{(m)}$ , the decision boundary remains unaffected, since each pairwise decision boundary depends solely on pairwise *differences* and pairwise *norm ratios*, both operations being invariant to translation. Algebraically:

Let  $\mathbf{u}'$  such that:  $u'_{ij} = u_{ij} + c_j$  Then:

$$\begin{aligned} \mathbf{P}'_u(M = m \mid \mathbf{R} = \mathbf{r}) &= \frac{\exp\{\sum_i u'_{mi} r_i\}}{\sum_{m'} \exp\{\sum_i u'_{m'i} r_i\}} \\ &= \frac{\exp\{\sum_i u_{mi} r_i\} \exp\{\sum_i c_i r_i\}}{\sum_{m'} (\exp\{\sum_i u_{m'i} r_i\} \exp\{\sum_i c_i r_i\})} \\ &= \mathbf{P}_u(M = m \mid \mathbf{R} = \mathbf{r}) \end{aligned} \quad (4.11)$$

For the invariance to vanish, one of the rows of  $\mathbf{u}$  has to be *fixed* independently of anything that affects the decision boundary (ie the data). The geometric interpretation of this is clear enough (Figure 4.3): the decision boundary depends on the *relative* position of the hyperplanes. Therefore, so as to construct any given decision boundary, the choice of the first hyperplane is free. In order for the decision boundary to uniquely identify the parameters, we then have to *fix* the first hyperplane.

This information is crucial with respect to the interpretation of the parameters, since it implies that the numerical value of a parameter independently of the rest is *uninterpretable*. The function of these vectors are to parameterise our model and hence we are interested in their values only insofar as they serve to define the decision boundary.

This invariance is known in the literature as the 'unidentifiability' of the multilogit classifier and the act of constraining the first row of the weight matrix is known as 'identifiability constraint'. However, it can be argued that *numerically* it makes no sense to impose an identifiability constraint. Recall that we learn the optimal hyperplane using gradient descent. However, by definition, the gradient will always be ver-

tical to any directions of invariance of the likelihood and will therefore never *move* along the directions of invariance. As a result, if for any interpretability reasons it is required that an identifiability constraint be made, imposing the constraint before and throughout training is equivalent to training freely and appropriately translating along the invariant directions after training. Moreover, recall that a global optimum does not imply a *unique* global optimum. In fact, precisely because of the invariance properties, entire subspaces of the hyperspace are now optimal. Therefore, training freely will in general be *more* efficient, since the algorithm is free to find the optimum that is nearest to the initialisation in the entire space, as opposed to the optimum that is nearest to it in the space delineated by the identifiability constraint, which of course can only be further away since the latter space is properly contained in the former. Therefore, we impose no identifiability constraint in our testing.

## 4.5 Interpreting inactivity

A key concern of the Segal group in [1] is the interpretation of the parameters as denoting the level of activity of each motif in each module:

As we expect a motif to be active in regulating only a small set of modules in a given setting, we limit the number of weights  $u_{i1}, \dots, u_{Ki}$  that are non-zero to some  $h \ll K$ . This restriction results in a sparse weight matrix for  $P(M | \mathbf{R})$ , and ensures that each regulator affects at most  $h$  modules. In addition, for interpretability considerations, we require all weights to be non-negative. Intuitively, this means that a gene's assignment to specific transcriptional modules can only depend on features that correspond to the presence of certain motifs and not on the absence of motifs.

In view of the invariance properties, this interpretation seems odd, since we can translate the constrained weight matrix along the directions of invariance to obtain a matrix that is neither sparse nor nonnegative and yet gives rise to exactly the same probability distribution.

The interpretation can indeed be shown to fail, but to justify this in terms of unidentifiability is misleading. In reality, the interpretation is so to speak *radically false* in that it attempts to answer ill-posed questions: the property assigned to non-negativity is altogether inconsistent with the multilogit model, whereas the property accorded to zero

cannot be possibly held by the same number for each datapoint. In the two subsections below we proceed to establish these results and then offer what seems to be the best possible alternative. We revisit this issue in the next chapter and offer a nonformal intuitive explanation of the derivation below in terms of the graphical representation of the model.

### 4.5.1 The insignificance of zero

We will now establish that the properties the authors attribute to zero and nonnegativity are indeed ill-defined themselves. First, we wish to find the value of  $u_{mi}$  for which it can be said of the model that:

**Statement 4** *Motif 1 is inactive in module  $m$ .*

This probabilistically translates precisely to the statement:

**Statement 5** *The probability that any given gene belongs to module  $m$  is independent of whether motif  $i$  is present or not in that gene*

which can of course be written as:

$$P(M = m | r_1 = 1, r_2, \dots, r_L) = P(M = m | r_1 = 0, r_2, \dots, r_L) \quad \forall r_2, \dots, r_L \quad (4.12)$$

Let us now take some time to prove the interpretation fails beyond doubt. We can rewrite 4.12 as:

$$\sum_{m'} \exp\left\{\sum_{i=2}^L u_{m'i} r_i\right\} (\exp\{u_{m1}\} - \exp\{u_{m'1}\}) = 0 \quad (4.13)$$

According to the passage the value of  $u_{mi}$  that makes 4.12 hold true is zero. Then 4.13 becomes:

$$\sum_{m'} \exp\left\{\sum_{i=2}^L u_{m'i} r_i\right\} (1 - \exp\{u_{m'1}\}) = 0 \quad (4.14)$$

where all the summands are nonpositive since  $\exp\{u_{m'1}\} > 1$  for all  $m'$  due to the non-negativity constraint. This directly implies that unless *all* the terms are zero (and hence  $\exp\{u_{m1}\} = \exp\{u_{m'1}\}$ ), motif  $i$  cannot be inactive in module  $m$ , even if its respective weight is 0.

This demonstration does not immediately exclude the possibility that some value other than zero or some constraint other than nonnegativity might possess the desired properties explained in the passage. But a closer look at 4.13 reveals that the dependence of the truth of 4.12 on the remaining parameters and the assignment of the rest of the regulates variables is ineliminable in general and hence the value of  $u_{mi}$  that makes 4.12 true is in fact a function of the remaining parameters and *also* varies with each datapoint. In particular, it is given by:

$$u_{mi} = \log\left(\frac{\sum_{m'=1}^K \exp\{\sum_{i=2}^L u_{m'i}\} \exp\{u_{m'1}\}}{\sum_{m'=1}^K \exp\{\sum_{i=2}^L u_{m'i}\}}\right) \quad (4.15)$$

which attains no simpler form in general.

This fact also implies that no alternative constraint can replace the nonnegativity constraint to the intended effect. The value representing inactivity can be seen to be obtained by an averaging operation over the rest of the components of that column. Therefore, for *any* given assignment of the parameters, the value representing inactivity will necessarily lie somewhere in the midrange of values and it therefore is impossible to force all parameter values to be *greater than or equal* to the "inactivity" value, which is precisely the desired property expressed by the authors. We briefly establish this more formally now.

The objective is to pick a constraint on  $\mathbf{u}$  such that it can be said of the model that:

**Statement 6** *The classification of a gene to a module can only depend on the presence of certain motifs and not on the absence of motifs.*

which probabilistically translates to the statement<sup>5</sup> :

$$P(M = m | r_1 = 1, r_2, r_3, \dots, r_L) \geq P(M = m | r_1 = 0, r_2, r_3, \dots, r_L) \quad \text{for all } m \quad (4.16)$$

To better see the correspondence, one can consider the negation of 4.16: assume there is a gene that is more likely to belong to a module  $m$  if it *does not* have motif 1 in its

---

<sup>5</sup>A variant understanding of 6 is this:

$$P(M = m | r_1 \text{ absent}, r_2, \dots, r_L, \mathbf{u}) = P(M = m | r_1 \text{ absent}, r_2, \dots, r_L, \mathbf{u}')$$

where  $\mathbf{u}'$  differs from  $\mathbf{u}$  only in parameters related to  $r_1$  (ie its first column). Our model indeed has this property by virtue of the choice of  $\{0, 1\}$  as labels, without any further constraint necessary.

promoter sequence than if it *does*. This precisely and directly contradicts statement 6. According to [1] one such constraint is nonnegativity. However, it is straightforward to show that 4.16 leads to a contradiction and hence no type of constraint can make it true. First we rewrite it as:

$$\sum_{m'} \exp\left\{\sum_{i=2}^L u_{m'i}r_i\right\}(\exp\{u_{m1}\} - \exp\{u_{m'1}\}) \geq 0 \quad (4.17)$$

Now consider the  $m$  for which  $u_{m1}$  is smallest (ie the least element of that column). Then:

$$\exp\{u_{m1}\} - \exp\{u_{m'1}\} \leq 0 \quad \forall m' \quad (4.18)$$

Equation 4.18 implies that all the summands in 4.17 are nonpositive. Therefore, it is necessary that  $u_{m1} = u_{m2} = \dots = u_{mL}$ , otherwise at least one inequality in 4.18 is strict and the LHS in 4.17 turns negative. But since we want 4.16 to hold *for all*  $m$ , this would imply that all parameters have to be equal. Therefore 4.16 is not tenable.

Finally, as a concluding remark, it ought to be mentioned that although the interpretation of the sparsity constraint has been shown to be radically false, its successful performance is not argued against here and is indeed accounted for on different grounds in the next chapter.

## 4.5.2 Approximating inactivity

We proceed to propose an alternative way of enforcing "sparsity", in the sense of *enforcing a certain number of parameters to be set to the value representing inactivity*. As mentioned already, the averaging operation 4.15 which yields this value for each datapoint ineliminably depends on the rest of the parameters and the  $r_i$ 's. The dependence on the parameters implies that enforcing sparsity becomes a hard combinatorial problem, since the order in which we "inactivate" weights becomes important. The dependence on the rest of the  $r_i$ 's poses even bigger a problem, since it means that a matrix can only be said to be "sparse" with respect to a *given* datapoint, never *for all* possible datapoints.

One way to deal with this is to approximate 4.15 using a function independent of the

$r_i$ 's, symmetrically <sup>6</sup>dependent on the rest of the parameters and in accordance with the directions of invariance.

A good guess for a first approximation to 4.15 is the biased estimator of the *mean* of the parameters:

$$u_{mi}^{(inactive)} \leftarrow \mu_m = \frac{1}{K} \sum_{m=1}^K u_{mi} \quad (4.19)$$

Equation 4.19 is appealing in terms of complexity since it is both linear and symmetric in the parameters<sup>7</sup> Moreover, it has the desirable property of being *conserved* during gradient descent. This is straightforward to establish. Let the parameter space be  $\mathbb{R}^{(M \times L)}$ , which corresponds to collapsing the parameter matrix to a row vector by placing the rows of  $u$  side by side. In this space, there are  $L$  directions of invariance with respect to the likelihood, where the  $i$ 'th direction is given by:  $(\delta_{ij})_j$ , where  $\delta_{ij} = 1$  if  $i = j$  and 0 otherwise. By definition, the gradient has to be perpendicular to each of these directions at all times. For instance, in a model with 3 motifs and 2 modules, the first direction of invariance is given by  $(1, 0, 0, 1, 0, 0)$ , and this implies that the following holds of the gradient:

$$(g_1, g_2, g_3, g_4, g_5, g_6) \cdot (1, 0, 0, 1, 0, 0) = 0 \quad (4.21)$$

$$g_1 + g_4 = 0 \quad (4.22)$$

Overall, then, the updates obtained by gradient descent methods always preserve  $\sum_m u_{mi}$  for all  $i$ , since they always move in directions of zero contribution in these directions. The sum of each motif over the modules is then conserved and hence so is the mean. Is it then a good approximation to 4.15? Unfortunately, there is no simple analytical way to test its quality for the total of  $2^L$  possible datapoints. Instead, we perform a series of numerical tests. Given the above, the initialisation is of course immaterial, so

---

<sup>6</sup>Symmetry reduces the complexity of the combinatorial problem, since the values of two parameters are equal or approximately equal the effect of changing the value of either of them on the value of "inactivity" for the rest of the parameters is identical.

<sup>7</sup>A nonlinear alternative could be the biased estimator of the mean of the logarithms of the parameters:

$$u_{mi}^{(inactive)} \leftarrow -\log(K) + \log\left(\sum_{m=1}^K \exp\{u_{mi}\}\right) \quad (4.20)$$

Datapoints considered	actual value	biased estimator	unbiased estimator
Module 1 only:	0.2316	0.0784	0.0989
All Modules:	0.0816	0.0090	0.0088

Table 4.1: Approximating inactivity using the mean over the modules

we start at  $\mathbf{u} = \mathbf{0}$ . After training, we evaluate the following quantities. We first set:

$$\mathbf{u}' \text{ such that: } u'_{11} = \frac{1}{K} \sum_{m=1}^M u_{m1} \quad (4.23)$$

$$\mathbf{u}'' \text{ such that: } u''_{11} = \frac{1}{K} \sum_{m=1}^M u_{m1} \quad (4.24)$$

and then evaluate:

$$d_{\mathbf{r}} = |\mathbf{P}_{\mathbf{u}}(M = 1 \mid r_1 = 1, r_2, \dots, r_L) - \mathbf{P}_{\mathbf{u}}(M = 1 \mid r_1 = 1, r_2, \dots, r_L)| \quad (4.25)$$

$$d'_{\mathbf{r}} = |\mathbf{P}'_{\mathbf{u}}(M = 1 \mid r_1 = 1, r_2, \dots, r_L) - \mathbf{P}'_{\mathbf{u}}(M = 1 \mid r_1 = 1, r_2, \dots, r_L)| \quad (4.26)$$

$$d''_{\mathbf{r}} = |\mathbf{P}''_{\mathbf{u}}(M = 1 \mid r_1 = 1, r_2, \dots, r_L) - \mathbf{P}''_{\mathbf{u}}(M = 1 \mid r_1 = 1, r_2, \dots, r_L)| \quad (4.27)$$

for all  $\mathbf{r}$  in DATASET A (totalling 720 datapoints) and consider the average values. We present the results in Table 4.5.2. A perfect approximator would show a mean absolute value of 0. Our approximator seems to be asymptotically correct, since the value increases as the sample becomes more representative. Table 4.5.2 indicates that our estimator is probably accurate enough and that not much difference exists between the biased and the unbiased estimator. We can now perform more extensive testing. We only consider the biased estimator this time. We numerically search for the value of  $u_{11}$  that minimises the mean absolute difference between  $\mathbf{P}(M = 1 \mid r_1 = 1, r_2, \dots, r_L)$  and  $\mathbf{P}(M = 0 \mid r_1 = 1, r_2, \dots, r_L)$ , where the averaging operation is over datapoints in each module and finally over the entire dataset. These results are presented in table 4.5.2. The solid conclusion we can draw is that the mean is an asymptotically correct estimator and a good estimator for most modules but performs poorly for certain modules. Since we know the dependence on the rest of the  $r_i$ 's is ineliminable, we cannot hope to do much better, without introducing an explicit dependence of the estimator on  $\mathbf{r}$ .

<b>actual min over all modules:</b>	<b>-0.06</b>
actual min over module 1:	0.32
actual min over module 2:	-0.1
actual min over module 3:	0.69
actual min over module 4:	0.49
actual min over module 5:	-0.32
actual min over module 6:	1.12
actual min over module 7:	-2.57
actual min over module 8:	0.25

According to these results it is natural to interpretate a "sparse" weight matrix as one where, for each column, the values of a large number of parameters equal the mean over the parameters for that column. As a consequence, the degree of sparsity can only be coherently controlled *per column*. This is in agreement with the biological intuition offered by the authors of [1], as seen in the passage quoted previously.

At the end of the main chapter we offer an alternative approach to this problem.

## Chapter 5

# Regulation Model: Regularisation and Pruning

In the section concerned with the maximum likelihood estimation of the parameters we mentioned the possibility of the multilogit model admitting a degenerative form where there is no finite global optimum in the parameter space and indefinitely increasing the magnitude of certain parameters leads to ever decreasing negative log likelihoods. Such degenerative behavior means that the classification confidence increases indefinitely and the model effectively turns deterministic, losing its expressive power. This loss leads to a significant fall in generalisation performance. It is therefore customary to impose certain "dampening" constraints on the parameters to avoid overconfidence. This approach was first studied in the context of neural networks and is known as *regularisation*. Most of the results for neural networks carry over almost directly to feed-forward Bayesian Networks.

A related issue in neural networks is the one of *pruning*, whereby a link between a unit and the next one is severed so as to reduce the degrees of freedom of the model and force it to be less confident in its classifications. This roughly corresponds to *freezing* a parameter in the weight matrix in our case, which is of course closely related to the sparsity constraint discussed previously, although the latter's motivation was different. The unifying concern then is that of the *optimal complexity* of the model. The complexity of a model can be understood in several ways. A particularly fitting expression

is the term ‘discriminative capacity’. When the complexity of the model is too low, the model is unable to capture the relevant patterns in the data and therefore is bound to misclassify a large proportion of the data. However, when the complexity of the model rises, the model is increasingly likely to locate a set of patterns that can act as a perfect classification criterion in the training dataset and then becomes overconfident in their significance as classification criteria. Since many of these patterns are bound to be highly specific to the training dataset as opposed to the generating process, the model then fails to generalise well. Both regularisation and pruning deal with precisely this issue: the former by explicitly constraining overconfidence despite the high complexity of the model and the latter by forcibly lowering the complexity itself.

In this chapter we investigate the issues of regularisation and pruning with respect to our model. As a result of this theoretical discussion we achieve drastic improvements in the generalisation performance of the model, as can be seen in the test results appended at the end of this chapter.

## 5.1 Training vs Test Datasets

Overconfidence poses a problem mainly in supervised learning tasks, where the parameters of the model are learnt from a set where both the predicted and the conditional variable are observed, but the model itself is then applied to datasets where the predicted variable is unobserved. In clustering tasks like our own, the model is usually meant cluster a certain dataset in the most likely way and new data is usually taken in joint consideration with old data to retrain the model and propose an alternative clustering. At any given application of the model, then, it will not be necessary to learn the parameters on a distinct dataset to the one it is meant to perform inference over. Does it make sense then to ban overconfidence?

It does, for two distinct reasons. Firstly, it is always good to be able to distinguish between highly likely cluster assignments and more uncertain cluster assignments, for the purposes of drawing safe biological conclusions. Secondly, since in our task a *hidden* variable indicates the cluster assignment, in effect generalisation performance is

very significant. This is because the initial assignment of the hidden variables is highly uncertain and the model is meant to capture the uncertainty in this initial assignment and hence suggest more likely assignments. An unregularised model will instead tend to overfit and hence view the initial assignment as highly likely, hence holding back or at least failing to contribute to the iterative refinements that are the basis of most training schemes for hidden variables models.

## 5.2 The Data Augmentation Approach to Regularisation

In the multilogit model, overfitting is not observed for all datasets. We now attempt to describe the kind of datasets that give rise to overfitting.

### 5.2.1 Divergence

The possibility of divergence is immediately visible algebraically. Optimality is attained when all the components of the gradient are equal to zero, that is when for all  $p$  and  $q$  we have:

$$\frac{\partial \mathcal{L}}{\partial u_{pq}} = 0 \quad (5.1a)$$

$$\sum_{g \in D} g \cdot r_q \mathbb{I}(g \cdot m = p) = \sum_{g \in D} g \cdot r_q \mathbf{P}(M = p \mid \mathbf{R} = g \cdot \mathbf{r}) \quad (5.1b)$$

$$N_D(p, q) = \sum_{g \in D \text{ where motif } q \text{ is present}} \mathbf{P}(M = p \mid \mathbf{R} = g \cdot \mathbf{r}) \quad (5.1c)$$

where  $N_D(p, q)$  is simply the number of genes in module  $p$  that have motif  $q$ . If then we assume that in our dataset motif  $q$  is absent in all genes under module  $p$ , then  $N_D(p, q) = 0$  and the equilibrium is attained when:  $P(M = p \mid \mathbf{r}) = 0$  for all genes that have motif  $q$ . This in fact corresponds to a divergence of at least one of the parameters, as can be seen by referring back to the formula for the probability distribution 4.1:

$$P(M = p \mid \mathbf{R} = \mathbf{r}) = 0 \quad \forall \mathbf{r} : r_q = 1$$

implies that

$$\frac{\exp\{\sum_{i \neq q} u_{pi} r_i'\} \exp\{u_{pq}\}}{\sum_{m=1}^M \exp\{\sum_{i=1}^L u_{mi} r_i'\}} = 0$$

Since the parameters are exponentiated these equations are only satisfied under divergence. In particular they imply that  $u_{pq} \rightarrow -\infty$ . What this implies in practice is that when a certain motif remains unobserved in module  $m$  in the training set, the presence of that motif in a gene is taken to imply with absolute certainty that this gene does not belong to module  $m$ .

This suggests *data augmentation* as a regularisation method: by appropriately augmenting the dataset with a few pseudopoints we can try to bring it into a form that will not give rise to divergence problems. We therefore have to describe a sufficient and hopefully also necessary (ie minimal) augmentation that will disallow divergence in any dataset.

However, the absence of a certain motif from a certain module is not *necessary* for divergence. In fact, relatively complicated non-sparse datasets where all motifs are present at least once and absent at least once in each module may still demonstrate divergent behavior.

This is expected. Let us take a moment to clarify why. If the parameter  $u_{mi}$  could diverge to negative infinity<sup>1</sup> only if motif  $i$  was entirely absent in module  $m$  it would then resemble the ratio:

$$\log\left\{\frac{\text{Number of occurrences of } i \text{ in module } m}{\text{Number of occurrences of } i \text{ in all modules}}\right\} \quad (5.3)$$

Indeed, we can construct a model analytically similar to the multilogit in which the parameters are equal to this ratio and which diverges only when a motif is entirely absent from a certain module. The construction of this model, which we call the "generative multilogit model" is included in the appendix.

The reason why the multilogit model diverges more often is that it implicitly models dependencies between motifs. Therefore, in a dataset where all motifs are present at least once and at present at least once in each module, it can still locate a "pattern"

---

<sup>1</sup>Relaxing this constraint somewhat, we could expect the parameter  $u_{mi}$  never to diverge to negative infinity, but diverge to positive infinity only if motif  $i$  was entirely absent from all modules *except*  $m$ . The parameter would then resemble the ratio:

$$\log\left\{\frac{\text{Number of occurrences of } i \text{ in module } m}{\text{Number of occurrences of } i \text{ in all modules except } m}\right\} \quad (5.2)$$

which is specific to one module only and become overconfident about it. This seems to suggest that each module ought to be augmented with *each possible pattern* to ensure convergence<sup>2</sup>, which would evidently be sufficient but is computationally unrealistic. The algebraic description of the model does not provide an easy way of handling this issue, whose nature is combinatorial, rather than analytical. The question that remains to be answered is this: can we ensure convergence without such an exhaustive scheme? We resort to the geometrical interpretation to answer this.

## 5.2.2 Linear Separability for Binary Classification

In binary classification tasks, it is often possible to find a hyperplane such that *all* datapoints of class 0 lie on one side of it and *all* datapoints of class 1 on the other. Datasets for which such a thing is possible are called **linearly separable**. Not all datasets are linearly separable. An example used often is the XOR dataset in two dimensions (fig 5.1). Low dimensional populous datasets are most likely to be linearly inseparable whereas high dimensional sparse datasets are usually linearly separable.

In supervised probabilistic modelling, the classification confidence is a function of the number of erroneous classifications. This means that linearly separable datasets are perilous in that the confidence of the classification reaches certainty, reflecting the errorless classification. If the linear separability was specific to the sample and not representative of the true data generating process, overconfidence will lead to a dramatic increase in the number of classification errors the model is expected to make when encountering a novel dataset from the same generating process. This account essentially provides a geometrical interpretation of the previous discussion of regularisation and complexity.

## 5.2.3 Linear Separability for the Multilogit model

Can we utilise the concept of linear separability to characterise divergent datasets in the multilogit model? Recall that the overall decision boundary is made up of the

---

<sup>2</sup>The relative size of the set of pseudopoints might then greatly exceed the size of the original dataset. This can be resolved by introducing scaling factors, as we do in the next section.

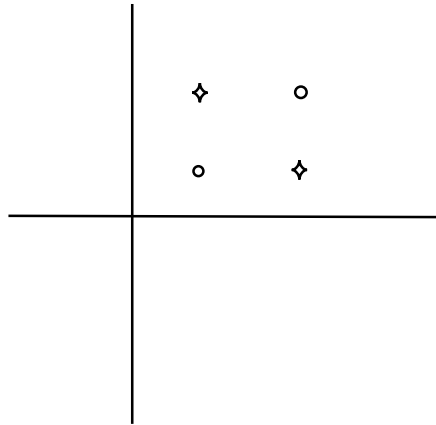


Figure 5.1: The XOR dataset

It is impossible to separate the diamonds from the circles using a line in the above diagram. This dataset is in fact generated by the logic ‘eXclusive OR’ function:

$$1 \text{ XOR } 1 = 0$$

$$1 \text{ XOR } 0 = 1$$

$$0 \text{ XOR } 1 = 1$$

$$0 \text{ XOR } 0 = 0$$

where the values 1 and 0 are understood with their Boolean meaning as ‘true’ and ‘false’ respectively. The dataset in the image is generated by taking the arguments of the XOR function to specify the coordinates of the point and the output of the XOR function to be the label.

appropriate segments of the pairwise decision boundaries. Therefore, the multilogit model does not even need to perform errorless classification in order to diverge. All it needs is to perform errorless *pairwise* classification: for instance, a classification with several errors but without any datapoints labelled '1' falsely given the label '3' can still lead to divergence. We did not attempt to prove this result formally for lack of time since it is intuitive and was numerically confirmed in all divergent datasets we tested the multilogit model on. Within the data augmentation framework, we then need to ensure each pair of labels are linearly inseparable. Since this strictly corresponds to  $\binom{L}{2}$  constraints, we ought to ensure the augmentation is done in a minimal way.

A way to easily ensure inseparability in discrete datasets is to introduce a certain datapoint once in each module. Despite the obvious linear inseparability, the model can *still* diverge, provided it can manage to place its decision boundary in such a way so as to contain that datapoint and linearly separate the rest of the data. However, when one introduces more such distinct linearly independent datapoints<sup>3</sup> in each module than the dimensions of the space, then the decision boundary cannot possibly contain all of them. In our case, since the model has no bias term and is therefore constrained to contain the origin in its decision boundary, an additional  $L$  datapoints will be minimally sufficient to ensure inseparability for each pair of labels.

However, since our dataset is contained within the unit hypercube, it turns out that we can use the *same*  $L$  datapoints for each pair of labels. We prove this result immediately. Consider a pairwise decision boundary  $\mathbf{D}$  which linearly separates datapoints of label '1' from datapoints of label '2'. It is forced to contain the origin and therefore must lie at the positive side of at least one axis and contain the rest of the axis.

Now consider the set of datapoints given by the rows of the identity matrix:  $\mathbf{e}^{(1)} = (1, 0, \dots, 0)$ ,  $\mathbf{e}^{(2)} = (0, 1, 0, \dots, 0)$  and so on. We follow standard vector algebra terminology in calling these our *basis vectors*. Each basis vector lies on one of the axes of our hyperspace and they are hence independent. Then, as explained, these datapoints cannot all be contained in  $\mathbf{D}$ . Assume only one of them is not contained in  $\mathbf{D}$ . It directly follows that any point in the unit hypercube either lies on one side of  $\mathbf{D}$ , or is

---

<sup>3</sup>A set of  $N$  datapoints are linearly independent when they define a hyperplane of  $N$  dimensions - for instance, three colinear datapoints in two dimensions are not linearly independent.

contained in it<sup>4</sup>. Therefore,  $\mathbf{D}$  cannot possibly linearly separate *any* set of points in the hypercube, since the latter's entire volume lies at one side of it. Therefore, since we have assumed that  $\mathbf{D}$  does separate a certain given set in the unit hypercube, we are forced to assume that *at least two* basis vectors lie outside  $\mathbf{D}$ . But since  $\mathbf{D}$  goes through the unit hypercube by assumption, then the two basis vectors must also lie *at different sides* of  $\mathbf{D}$ . Therefore  $\mathbf{D}$  cannot possibly linearly separate the augmented version of the dataset.

Finally note that although divergence is avoided by the introduction of the above minimal set of pseudocounts per module, additional pseudocounts will have further regularising effects and might be desirable. In general, we can express the contribution of the pseudocounts as a penalty term that is added onto the negative log likelihood of the model and is *scaled* by a parameter (*hyperparameter* as we will see in the next chapter)  $\alpha$ :

$$-\text{Penalty}_{PI} = \sum_m \sum_i (u_{mj} \cdot \delta_{ij} - \log \{ \sum_{m'} \exp \{ u_{m'j} \delta_{ji} \} \}) \Rightarrow \quad (5.4)$$

$$\text{Penalty}_{PI} = - \sum_m \sum_i u_{mi} + M \cdot \sum_{i=1}^L \log \sum_{m'=1}^M \exp u_{mi} \Rightarrow \quad (5.5)$$

### 5.3 The Smoothing Effect of the Data Augmentation Approach

The data augmentation scheme has a very desirable property: it introduces a smoothing constraint to the behavior of the dataset which in the unregularised model can be very sensitive to small perturbations. It is easy to establish this numerically. Consider a pure dataset, where each datapoint in each module have identical motif profiles and the motif profiles have empty pairwise intersections. The model on this dataset certainly diverges. By augmenting the dataset using our scheme, the divergence is removed and the model converges fast. Then, by removing a *single* datapoint, the divergence is reintroduced. This is of course an artificial situation but since our datasets are gener-

---

<sup>4</sup>To make this evident imagine the situation in 3 dimensions. If  $\mathbf{D}$  contains the origin and a point on each of two axes, then it must contain one of the sides of the unit hypercube

These two figures represent the high sensitivity of the multilogit model to negligibly small amounts of noise for some datasets. The graph on the left represents the dataset, each row being a datapoint and each column a variable. The dataset is separated in three modules: 1-90, 91-180 and 181-270. The dark value indicates presence whereas the light one absence. The original unaugmented dataset consisting of the three main 'blocks' diverged with weights exceeding sizes of 100 when our algorithm stopped due to numerical accuracy issues. The augmented, unperturbed dataset, consisting of the original data with an additional identity matrix inserted in each of the three modules showed fast convergence (in fact, it converged in one single iteration of conjugate gradient descent) learning values for the weights that did not exceed 2.5 in absolute value. Finally, we perturbed the dataset by setting the right-topmost datapoint to 0. The effect in the training can be seen in graph 5.2: the behavior has drastically changed from one of convergence to divergence, resulting in a dramatic change in terminal values for the parameter set. Notably, in a dataset of 270 datapoints, the perturbation we have effected corresponds to  $< 1/1000$  noise contamination.

ally sparse, we expect counterparts of this situation to arise very frequently: it is highly likely that in at least one pairwise classification the model will either be able to linearly separate the data or be on the *verge* of linear separability. Therefore, switching a single value of a single datapoint, which corresponds to a negligibly small noise contamination, the behavior of the model parameters can change dramatically, from convergence to divergence. Such an example is shown in figure 5.3 for the dataset 5.2.

The converse of this, however, is that a divergent dataset can be efficiently brought to a state of convergence by the introduction of noise. Indeed, when the noise level is sufficient, the model will no longer be sensitive to single changes and the noise will have a smoothening effect on the model parameters. Training by adding noise is therefore *suggested* as a possible regularising solution by the same rationale as our own data augmentation technique. The smoothening effect of training with noise has been treated in more depth under a different perspective in M.Bishop (1995). For lack of time, we did not have the opportunity to compare these two regularisation techniques. Our argument however indicates that the data augmentation technique manages to ensure convergence in a minimal manner, which might be an advantage over regularisation with noise.

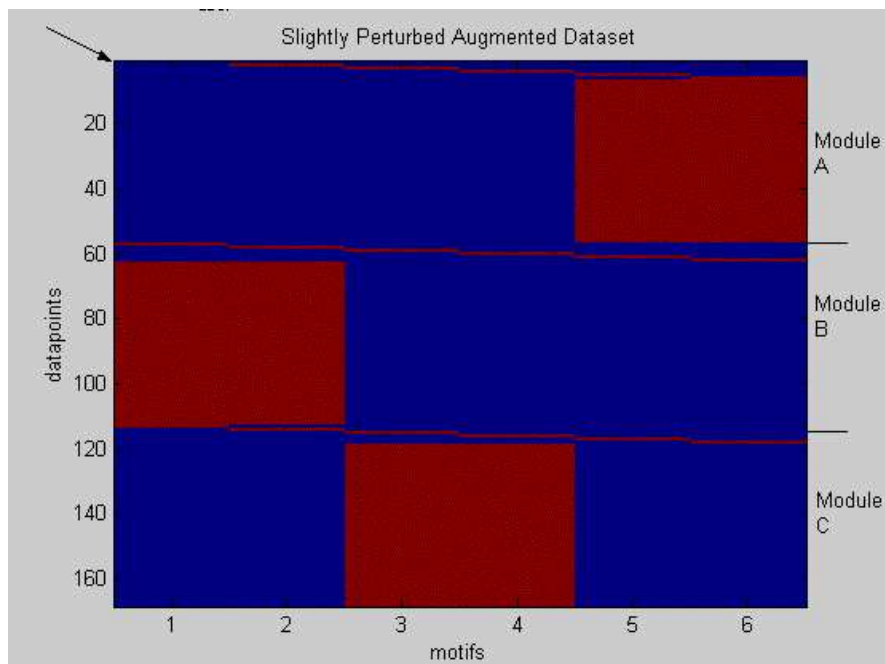


Figure 5.2: Slightly perturbed augmented dataset

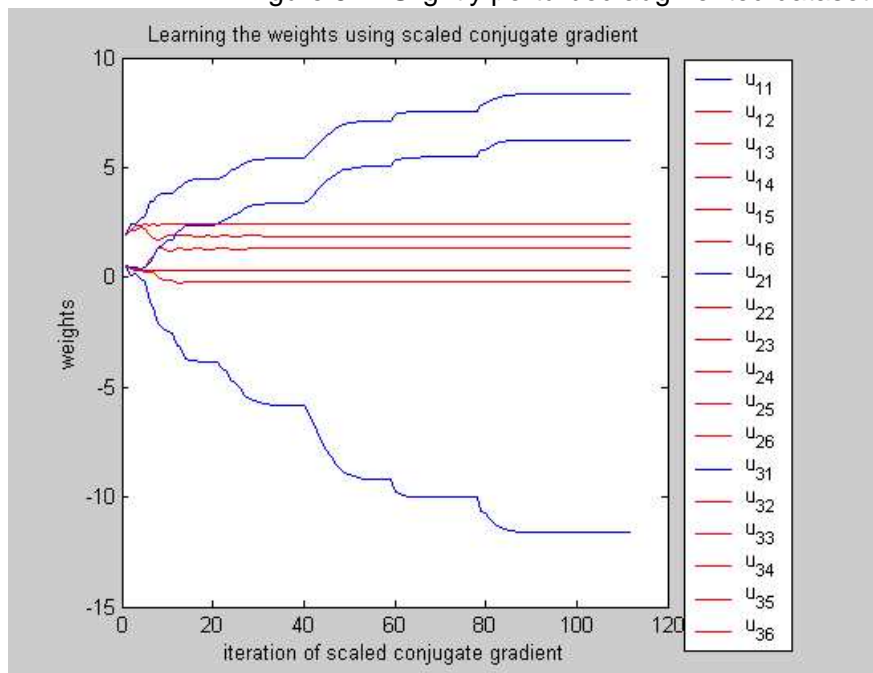


Figure 5.3: Learning of the weights under a slight perturbation

## 5.4 The Bayesian Approach to Regularisation

### 5.4.1 Prior Distributions

Under a Bayesian understanding of regularisation, any penalty we wish to impose on the parameters to prevent them from diverging is essentially the expression of a prior belief about the size of the parameters. To put it simply, we express our belief that the nature of the dataset is such that it is highly unlikely for the parameters to receive very big values. This belief then ought to be introduced formally as a *prior distribution* over the parameter values.

Formally, we denote the prior distribution over the parameter  $\mathbf{u}$  by  $p(\mathbf{u})$ . Then by Bayes' theorem we obtain an expression for the posterior distribution of the parameters given the weights:

$$P(\mathbf{u} | D) \propto P(D | \mathbf{u})P(\mathbf{u}) \quad (5.6)$$

The most likely setting of the parameters is then given by the *maximum a posteriori* (MAP) solution:

$$\mathbf{u}_{MAP} = \underset{\mathbf{u}}{\operatorname{argmax}} P(\mathbf{u} | D) \underset{\mathbf{u}}{\operatorname{argmax}} P(D | \mathbf{u})P(\mathbf{u}) \quad (5.7)$$

which is merely the assignment of  $\mathbf{u}$  that has the highest probability according to the posterior distribution over the parameters. The difference between the MLE and MAP assignment of the parameters is then simply that to obtain the latter we maximise  $P(D | \mathbf{u})P(\mathbf{u})$  whereas to obtain the former we maximise  $P(D | \mathbf{u})$ . When the prior  $P(\mathbf{u})$  expresses a regularisation constraint, then taking the negative logarithms of the posterior for computational purposes we obtain the expression for the *regularised likelihood*:

$$\mathcal{L}_R = \mathcal{L} + \log(1/P(\mathbf{u})) \quad (5.8)$$

where now  $\log(1/P(\mathbf{u}))$  can be viewed in more familiar terms as the *penalty term*.

As we have seen the prior over the parameters is meant to be independent of the data and hence of the data generating process. As a result, neither the form, nor the parameters of the prior distribution (or the "hyperparameters" as they are often called), are dictated by any frequency data. We are therefore faced with the practical problem

of making an appropriate choice of prior distribution in the absence of frequency data and the theoretical problem of justifying it. We will only be concerned with offering a solution to the former *practical* problem here and we will motivate our solution with mainly *intuitive* arguments, exploiting the geometrical interpretation to bypass formal arguments.

### 5.4.2 Motivating the Exponential Priors

The degree of confidence in a certain pairwise classification is geometrically expressed by the relative moduli of the respective rows of  $\mathbf{u}$ . The undesirability of overconfident classification therefore translates to a belief that the variation in the moduli of the rows of  $\mathbf{u}$  ought not be very large. Almost equivalently<sup>5</sup>, we could assume that we expect the moduli of the rows of  $\mathbf{u}$  to be dispersed around a certain mean value and given the invariance properties, we can take this value to be 0. The expected value of 0 is also suggested by the fact that 0 is the only value for the modulus of a vector that indicates all the components of the vector are equal (namely, 0). This is a desirable property, since when all components of  $\mathbf{u}$  are equal, then *all* points in the hyperspace have equal probability of being assigned to any module. Since in the absence of a training set we expect our parameters to receive their expected value, setting that value to 0 is appropriate since it forces all the parameters to be identical and hence reflects total ignorance.

Now, apart from this we have no further prior beliefs about  $\mathbf{u}$ . We are therefore looking for a probability distribution that forces the square of the modulus of each row of  $\mathbf{u}$  to receive values close to 0 and apart from that is maximally uncertain about what values the modulus actually takes. We take the square of the modulus to eliminate the square root that features in the modulus and hence make computations easier. In other words, we are looking for the *maximum entropy* distribution over a positive variable  $x$  with constrained mean. This is given by the Laplacian density:

$$P(d) \propto e^{-\alpha d} \tag{5.9}$$

---

<sup>5</sup>We discharge this at the Appendix

where  $\alpha$  is a positive hyperparameter and  $d$  is the modulus of a row. We expect all rows are identically distributed and therefore we now have our prior distribution over  $\mathbf{u}$ :

$$\begin{aligned}
 P(\mathbf{u} \mid \alpha) &= \prod_{m=1}^M P(|\mathbf{w}^{(m)}| \mid \alpha) \\
 &= \prod_{m=1}^M \frac{e^{-\alpha|\mathbf{w}^{(m)}|}}{Z_m} \\
 &= \prod_{m=1}^M e^{-\alpha \sum_{i=1}^L u_{mi}^2} \quad \text{since } \mathbf{w}^{(m)} \equiv (u_{mi})_i
 \end{aligned} \tag{5.10}$$

where  $Z_m$  is the normalising constant for each  $m$ . This distribution is in fact a Gaussian density over each row of the matrix  $\mathbf{u}$  with 0 mean and variance equal to  $2\alpha$ . Hence, we call this prior *Module-Specific Gaussian*.

In motivating the Gaussian prior, we chose to consider the modulus of each row vector as a measure of overconfidence. However, all the properties of the modulus that we have actually used are also possessed by the quantity  $|w_1^{(m)}| + |w_2^{(m)}| + \dots + |w_L^{(m)}|$ , known as *norm-1*<sup>6</sup>. Using norm-1 as opposed to the modulus of the row, we obtain the following prior:

$$P(\mathbf{u}) = \prod_{m=1}^M e^{-\alpha \sum_{i=1}^L |u_{mi}|} \tag{5.11}$$

This corresponds to a module-specific *Laplacian Density*. The choice between the Laplacian and the Gaussian priors is not univoqually dictated by the argument presented so far<sup>7</sup>. However, the methods are fundamentally different since, as we will see shortly, the Laplacian density allows us to perform *pruning* as well as regularisation in a systematic way. We therefore implement both and compare their performance in the results section.

---

<sup>6</sup>The number '1' denotes the power to which we raise each component of the vector whose norm we are considering. Therefore, the modulus (or Euclidean norm) is also known as *norm-2*.

<sup>7</sup>An attempt to investigate whether there are any formal grounds for choosing between the two is included in the theoretical discussion of this section in the Appendix

### 5.4.3 Eliminating the Hyperparameters

Observe that the prior distributions are in turn dependent on a hyperparameter  $\alpha$  which is not known. To get some insight into the role of  $\alpha$ , we substitute our priors in 5.8 and obtain the expression for the regularised negative log likelihoods in terms of the unregularised negative log likelihood  $\mathcal{L}$  and a penalty term:

$$\begin{aligned}\mathcal{L}_R &= \mathcal{L} + \log(1/P(\mathbf{u})) \\ &= \mathcal{L} + \alpha \sum_m |\mathbf{w}^{(m)}| \end{aligned} \quad (5.12)$$

for the Gaussian prior and similarly:

$$\mathcal{L}_R = \mathcal{L} - \frac{\alpha}{2} \sum_m \|\mathbf{w}^{(m)}\|_1 \quad (5.13)$$

where  $\|\cdot\|_1$  is used to denote the *norm-1* we discussed before. Therefore, the role of  $\alpha$  is significant since it *scales* the penalty term.

There are two approaches in the literature as to how to deal with  $\alpha$ . Firstly, one can leave the optimise over  $\alpha$  by first marginalising over the parameters and then maximising this expression, which will still be a function of the hyperparameter  $\alpha$ :

$$\begin{aligned}\alpha_{optimal} &= \underset{\alpha}{\operatorname{argmax}} P(D, \alpha) \\ &= \int_{\alpha} P(D | \mathbf{u}) P(\mathbf{u} | \alpha) d\mathbf{u} \end{aligned} \quad (5.14)$$

This method is very natural since it is the marginal likelihood of the data that we are ultimately interested to maximise. However, the integral featured in 5.14 is usually analytically intractable and therefore has to be approximated using quadratic or variational approximations. This process is involved, computationally expensive and often suffers from convergence problems. An alternative simpler route that is sometimes available is to *integrate over the hyperparameters*. We follow Williams (1995) in pursuing the latter route<sup>8</sup>.

In this approach, one finds an analytical expression for the prior that is independent of the hyperparameters by solving analytically the following integral:

$$P(\mathbf{w}^{(m)}) = \int P(\mathbf{u} | \alpha) P(\alpha) d\alpha \quad \text{where } \mathbf{w}^{(m)} \text{ is the } m\text{'th row of matrix } \mathbf{u} \quad (5.15)$$

---

<sup>8</sup>A comparison of the two schemes is unfortunately beyond the scope of this dissertation but is available in Williams (1995).

where  $P(\alpha)$  is a *hyperprior* over the positive hyperparameter  $\alpha$ . As we explained before, the hyperparameter is in this case a scaling factor and the most common choice of hyperprior in such circumstances is the so-called *ignorance prior* given by:  $P(\alpha) = \frac{1}{\alpha}$ . The intuitive explanation for this is that since  $\alpha$  is a scale variable, it should not depend on the scale of the ruler that is used and should be invariant to scale changes.  $P(\alpha) = \frac{1}{\alpha}$  is then appropriate since it is invariant with scale changes to  $\alpha$ . Using this ignorance prior, the integral can then be solved to yield:

$$P(\mathbf{w}^{(m)}) = \frac{L}{2} \log\left(\frac{1}{2} \|\mathbf{w}^{(m)}\|\right) \quad (5.16)$$

for the Gaussian and the similar expression:

$$P(\mathbf{w}^{(m)}) = L \log(\|\mathbf{w}^{(m)}\|) \quad (5.17)$$

for the Laplacian, where  $L$  is the number of motifs. The actual integration is not difficult and can be found in the Appendix.

## 5.5 A Bayesian approach to the Sparsity Constraint

This section can be outlined as follows:

- Sparsity can be understood as pruning
- Pruning can improve generalisation performance, especially in sparse datasets
- Pruning ought not be seen as a combinatorial problem, but can be dealt with in a systematical Bayesian manner with the use of appropriate prior distributions

This reconceptualisation of the sparsity constraint is arguably the main import of the thesis.

### 5.5.1 Pruning and Sparsity

The sparsity constraint is strongly related to the practice of *pruning* in neural networks. There, too, weight matrices representing the association strength of each unit to units in the next layer are sometimes forced to be sparse, indicating that the two units whose

link is severed by setting the corresponding weight to 0, are now dissociated. However, in the case of pruning, zero *has* a special significance since it represents zero association. As shown in the previous chapter neither zero nor any other fixed number has this significance. Bypassing the mathematics of the previous chapter, this can be readily seen in the graphical representation of the model, where severing a link coming out from a motif effectively prunes the node, removing its influence on all modules. Note that the graphical representation is *meant* to incorporate such information: pruning precisely consists in introducing a conditional independence assumption, which, in turn, is precisely what the representation of graphical models depicts. However, a feature of the sparsity constraint that remains significant is the fact that the parameters that are set to zero remain fixed at that value once pruned. In this sense sparsity remains similar to pruning. As mentioned in the beginning of this chapter, pruning is used to decrease the complexity of the model and avoid overfitting. This ought to be ideally done by dissociating certain nodes, but the main intended effect is to decrease the degrees of freedom of the model, therefore lowering its complexity. Freezing the parameters to a certain value has that effect as well.<sup>9</sup> The dimensionality of the model is not, strictly speaking, lowered as in the case of node deletion, but the model still becomes less flexible and hence less able to linearly separate the data. Therefore, the sparsity constraint can be thought of as a regularisation scheme. Finally, to the extent that the fixed value approximates the value representing dissociation for this particular motif, freezing the parameters also to a large extent dissociates the motif with the module in question.

Finally, a word of caution is in order, since the similarity between pruning and freezing ought not be overstated. Freezing in general makes the optimisation task considerably harder than pruning. Pruning usually<sup>10</sup> preserves the form of the distribution and hence simply recasts the search space to a lower dimensional counterpart of the original, preserving convexity/concavity properties. In contrast, freezing will in general distort

---

<sup>9</sup>In terms of our geometrical interpretation, the effect of freezing one parameter in a given module is essentially to force the model to consider, for that module, normal vectors whose point where the head of the arrow belongs to a given fixed hyperplane. The more parameters that are frozen, the lower the dimension of the constraining hyperplane and, hence, the largest the constraint.

<sup>10</sup>In all cases where pruning is traditionally applied the probability distribution/transfer function is functionally symmetric in the inputs, ie it differs only at the assignment of the parameters.

and/or fragment the error surface.

Having clarified the relationship between pruning and freezing, for simplicity we will use the two terms *interchangeably* for the remainder of this thesis, with the understanding that, strictly speaking, we are not performing *pruning* but rather *freezing*.

### 5.5.2 Pruning via the Laplacian Prior Without Mathematics

Choosing what and when to prune presents a challenging optimisation problem of a combinatorial nature. Both the set of parameters to be pruned and the order in which they are pruned in general affects the shape of the error surface, whereas even after these two choices have been made, a global optimum is not guaranteed.

Heuristic methods of pruning usually involve a greedy search over sparse matrices, with an optimisation step for each matrix considered. The method of Segal et al. (2003) falls in this category. We present a radically different method. Having placed the problem of enforcing a sparsity constraint in the context of regularisation, we can now exploit a property of the Laplacian prior to perform both pruning and regularisation naturally and ‘on the fly’ during normal training.

The key property of the Laplacian distribution is its *discontinuity at the origin*. The effect of the discontinuity is easy to describe intuitively. Let us take the standard example of the Gaussian prior. The Gaussian prior introduces a penalty term which is an increasing function<sup>11</sup> of the modulus of the parameter. Under a Gaussian penalty, those parameters that contribute very little at the likelihood of the model will be drawn towards the origin. However, since the Gaussian is smooth and the penalty at the origin is evidently zero, as the parameters approach the origin the penalty will tend to vanish. However, we do not expect any parameters, no matter how insignificant, to have negligibly small contributions to the overall likelihood. Therefore, as a general rule, insignificant parameters will tend to receive terminal values in the *extended* neighborhood of zero, as opposed to its *immediate* neighborhood.

In the Laplacian case, the penalty is not a function of the modulus of the parameter and therefore it does not vanish as the parameter approaches the mean of the prior. At the mean itself the penalty is of course zero, but even at negligibly small values near the

---

<sup>11</sup>A quadratic function, in specific

origin, the penalty is fixed and as large as anywhere else in the parameter space. Now consider one of the insignificant parameters referred to previously. If the contribution of the parameter to the likelihood is sufficiently small, the penalty term will dominate and therefore the parameter will be trapped within a small neighborhood of the origin. In fact, the parameter will be forced to switch from one side of zero to the other, since every time the penalty will exceed the contribution of the likelihood in one of the two directions. The key observation then is the following: *unless the region of the parameter space drastically changes* as a result of the behavior of the rest of the parameters, this specific parameter will remain trapped in a state of Brownian motion around the origin. The suggestion then is to identify such parameters and set the value of both the gradient and the parameter itself to exact zeros and remove them from consideration from that point onwards in the algorithm.

To summarise:

- Choose a prior with a simple discontinuity at the origin
- Every time a parameter crosses the origin, establish whether it is receiving enough pressure from the likelihood term to escape the penalty term and leave the origin, or whether it is trapped in Brownian motion around the origin.
- If trapped, prune it.

This process can be accelerated by amending the gradient descent algorithm to actively look for nearby zeros. No bias in the direction of descent is introduced, but merely the *size of the step* is selected to bring one parameter at a time as near as possible to the origin. The mathematical and interpretational details of this algorithm are nontrivial and can be found in sufficient detail in Williams (1995). For completeness, a very brief outline of the pruning algorithm is included in Table 5.1.

There are several practical reasons to prefer the Laplacian pruning scheme to a greedy search approach. Firstly, it combines a dampening scheme with a pruning scheme, providing a more integrated regularising solution. Secondly, its formulation as an exponential prior makes it easy to deal with analytically, making it potentially possible to incorporate it in more sophisticated training methods like variational methods of estimating the true posterior or Markov Chain Monte Carlo sampling schemes. Thirdly, it

<p>Let <math>w</math> denote the set of parameters <math>(w_j)_{j \in \{1, \dots, W\}}</math>. Then:</p> <p>Retrieve direction of descent <math>s</math> and recommended step size <math>\xi^*</math> from a gradient descent algorithm</p> <p>Define, for all <math>j</math> such that <math>s_j \neq 0</math>, <math>\xi_j = -\frac{w_j}{s_j}</math></p> <p>Find <math>k</math> such that <math>\xi_k</math> is the closest to <math>\xi^*</math> of the <math>\xi_j</math>'s.</p> <p>If <math> 1 - \frac{\xi_k}{\xi^*}  &lt; \delta</math> for <math>\delta = 1</math> replace <math>\xi^*</math> by <math>\xi_k</math>.</p> <p>Ensure <math>w_k</math> subsequently is set equal to an <i>exact</i> zero by explicitly setting it to that value.</p> <p>Find all parameters that satisfy <math>w_j = 0</math> and <math>\frac{\partial \mathcal{L}}{\partial w_j} = 0</math></p> <p>Prune them by appropriately contracting the parameter vector.</p> <p><i>Reevaluate</i> the gradient and likelihood for the new parameter vector and proceed.</p>
--

Table 5.1: A summary of the algorithm for pruning using a Laplace prior

arguably poses less of a strain on the optimisation task than a greedy search approach. This point is hard to argue conclusively, but several features of the Laplacian scheme seem to hint in this direction. In particular, we emphasize again that the scheme only prunes those parameters that contribute the least to the unregularised likelihood and that are guaranteed to remain at an incapacitant state within the local neighborhood of the overall parameter space.

## 5.6 Performance of Regularisers

We now present the results of the testing we performed to compare the performance of various regularisers. The synthetic data was generated as follows: for the first two datasets, we constructed a sparse binary  $19 \times 8$  matrix. This we took to be the *true structure* of the data. We then generated 720 datapoints, 90 per module by adding random noise to the respective rows of the structure matrix. These datapoints form Dataset A. Dataset B is obtained by adding even more noise to Dataset A. The true structure of the data can be seen in 5.9, where black indicates presence and white indi-

cates absence. Finally, dataset C was obtained with exactly the same process as above, but the true structure matrix was much larger and sparser. We used the matrix relating motifs to modules learnt by the model used in [1] applied on the Yeast stress dataset, to ensure the difficulty of the task as far as sparsity and size are concerned was realistic. We then trained the following 10 schemes:

1. Unregularised Default: the multilogit model without any regularisation scheme
2. Unregularised Generative: a multinomial model recast in a softmax basis without regularisation
3. Unregularised Symmetric: the multilogit model without regularisation where the input is binary symmetric
4. Gaussian Regulariser: we impose a single Gaussian on the entire parameter matrix
5. Motif Specific Gaussian: we impose a Gaussian distribution on each column of the matrix
6. Module Specific Gaussian: we impose a Gaussian distribution on each row of the matrix
7. Laplacian Regulariser: we impose a single Laplacian on the entire parameter matrix
8. Motif Specific Laplacian: we impose a Laplacian on each column of the matrix
9. Module Specific Laplacian: we impose a Laplacian on each row of the matrix
10. Pseudopoints Insertion Method: this is the data augmentation method

The initialisation of the parameters was identical for all schemes and was set to small random values with a mean of 0 for each row of the weight matrix. In the Laplacian regulariser, this is hoped to ensure that after training the mean of the parameters is

equal to the value that the Laplacian regulariser sets pruned weights to, therefore automatically ensuring that pruned weights are approximately inactive.

In tables 5.6, 5.6 and 5.6 we present certain measures of the performance of each scheme for each dataset. We performed non-overlapping 3-fold validation for datasets A and B and 2-fold for C, separating each dataset in 3 folds with each module equally represented in each fold and then training the scheme for each fold, testing its performance on the other two. We then present the generalisation error for each fold (GEF1, GEF2, GEF3) as well as the average generalisation error (AvGE). We also compute the value of the *unregularised* likelihood of *each fold* under the assignment of the parameters that was found optimal *for each fold* for each scheme. This is a good indicator of overfitting, since when the model overfits, we expect that the unregularised likelihood will be relatively lower for the fold the particular assignment of the weights was learnt on than the other two folds. It is important *not* to take into consideration the regularised likelihood for such comparisons, since the size of the penalty terms are not comparable between different schemes. These three figures for each fold are presented under 'LL' (for negative Log Likelihood), are separated by commas and the test fold that is identical to the training fold is always put in bold fonts for readability. Finally, the likelihoods of the symmetric and the generative models are put in brackets since they are not comparable to the likelihoods of the multilogit model.

These figures are easily interpretable and provide strong support for almost all the claims we have made in the last two chapters. We present our observations:

1. The unregularised multilogit model always has a poorer performance than its regularised counterparts.
2. The generative unregularised model indeed proves inferior to the multilogit model.
3. The symmetric unregularised model has a better generalisation performance, proving that the choice of labelling affects the performance of the model significantly.
4. The data augmentation significantly improves the generalisation performance, performing nearly as well as the best performance in the two smaller datasets.

5. The Gaussian and Laplacian regularisers offer very significant improvements in generalisation performance.
6. The Module specific regulariser performs better than the single regulariser both in the case of the Gaussian and the Laplacian
7. The Module specific Laplacian regulariser performs very well in all datasets and performs best by a significant margin in the case of the larger dataset.

There were also two observations that were less in tune with our predictions. Firstly, the motif specific regularisers perform nearly as well or even marginally better than the module specific ones in the smaller datasets. However, in the larger dataset the module specific schemes significantly outperform them.

Secondly, the mean of the learnt weight matrix which was conserved in the unregularised scheme and the data augmentation scheme was shifted by 0.1-0.4 upwards after training the Bayesian regularisers. This compromises the interpretability of pruning somewhat, since the value of pruned weights is no longer equal to the mean value that represents inactivity. This seems to indicate that the multilogit model scales weights differently when they are above the mean value than when they are below it. This is also indicated by the asymmetry between the maximum and minimum values encountered in the weight matrices for the Bayesian schemes: a maximum of 4 versus a minimum of just -1.5. The failure to conserve the mean could hence potentially be corrected by carefully selecting an appropriately *skewed* prior distribution with a simple discontinuity at the origin.

To assess the contribution of the Laplacian density to the interpretability considerations that have been central to our discussion in this chapter, we include a more graphical representation of the results. We present the learnt parameter matrices for the first fold under schemes 1,6,9 and 10 as a matrix coloured with a greyscale colormap, where black colour is reserved for the highest value encountered for any parameter in this matrix and white for the lowest. Alongside them we present the matrices representing the true structure of the dataset, where black indicates presence and white absence. Firstly, a careful observation quickly establishes that the black regions of the

regularised methods largely coincide with the true structure behind the datasets, especially in the case of the Bayesian regularisers. However, the Laplacian regulariser also manages to "inactivate" most entries that correspond to absence in the true structure. Note that, in view of the interpretation of inactivity given in the previous chapter, the white areas in the matrices representing the true structure of the data are supposed to be mapped to *grey* regions in the weight matrices, as indeed they are in the case of the Laplacian regularisers.

As a general conclusion, we can fairly safely conclude that the Laplacian regulariser offers significant improvements both in performance and interpretability, remaining at the same time computationally tractable and theoretically justified.

	GEF1	LL	GEF2	LL	GEF3	LL	AvGE
Unregularised Default	67	<b>161,547,555</b>	69	355, <b>166,350</b>	52	265,247, <b>195</b>	62.67
Unregularised Generative	72	[ <b>235,484,498</b> ]	69	[477, <b>253.5,462</b> ]	58	[368.8,342.2, <b>289.8</b> ]	66.333
Unregularised Symmetric	69	[ <b>246,309,322.2</b> ]	63	[379, <b>250.6,374.2</b> ]	49	[310.2,292, <b>272.7</b> ]	60.333
Gaussian Non Specific Regulariser	56	<b>238,262,277</b>	53	278, <b>240,282</b>	45	273,261, <b>256</b>	51.333
Gaussian Motif Specific Regulariser	49	<b>211,239.8,255.3</b>	50	248, <b>213,254</b>	46	244,236, <b>229</b>	48.333
Gaussian Module Specific Regulariser	49	<b>211,239.7,255.17</b>	51	248.4, <b>214,255</b>	46	244,236.4, <b>229</b>	48.667
Laplacian Non Specific Regulariser	57	<b>213.4,254,271.4</b>	51	260.4, <b>212,265.9</b>	47	253.8,244, <b>234.5</b>	51.667
Laplacian Motif Specific Regulariser	60	<b>190,269,283</b>	56	266.7, <b>184.3,264.6</b>	48	249,233, <b>211</b>	54.667
Laplacian Module Specific Regulariser	58	<b>229,257,274.7</b>	52	266.7, <b>226,273</b>	46	261,250, <b>244</b>	52
Pseudocounts Method ( $\alpha = 1.7$ )	54	<b>259,277,290</b>	49	291.5, <b>252.4,293.4</b>	43	286.6,274.2, <b>268.7</b>	48.667

	GEF1	LL	GEF2	LL	GEF3	LL	AvGE
Unregularised Default	106	<b>464.2,530.6,555.9</b>	112	547, <b>446.6,555.5</b>	100	524,505, <b>489.4</b>	106
Unregularised Generative	121	[ <b>563.7,631.8,662.7</b> ]	117	[627.2, <b>515.7,630.2</b> ]	112	[584.7,567.6, <b>549.2</b> ]	116.667
Unregularised Symmetric	107	[ <b>564.4,584.8,606.7</b> ]	107	[601.6, <b>545,600.4</b> ]	105	[606,587, <b>586</b> ]	106.333
Gaussian Non Specific Regulariser	104	<b>505,532,551</b>	106	543.6, <b>490,547.7</b>	98	549.5,533.9, <b>530.6</b>	102.667
Gaussian Motif Specific Regulariser	103	<b>476,512,533.4</b>	103	520.8, <b>459.5,530</b>	99	519.5,504.7, <b>500</b>	101.667
Gaussian Module Specific Regulariser	102	<b>475.7,512,533.4</b>	103	521, <b>460,530.2</b>	99	519.5,504.6, <b>500</b>	101.33
Laplacian Non Specific Regulariser	103	491.4, <b>520.8,544</b>	103	538, <b>481,545.4</b>	102	532.5,516, <b>513.9</b>	102.667
Laplacian Motif Specific Regulariser	103	<b>467,521,546</b>	108	535, <b>449,544.7</b>	102	520,504, <b>492</b>	105.33
Laplacian Module Specific Regulariser	106	<b>483,520.7,540</b>	104	531, <b>467,540</b>	96	527,510, <b>506</b>	102
Pseudocounts Method ( $\alpha = 2.4$ )	109	<b>500.8,533,551</b>	106	543, <b>487,546</b>	102	542,527, <b>520</b>	105.667

Table 5.2: Performance of various regularisers for Datasets A and B

	GEF1	LL	GEF2	LL	AvGE
Unregularised Default	164	<b>0,5363</b>	190	6665,0	177
Unregularised Generative	262	[ <b>624,1282</b> ]	236	[1258, <b>606</b> ]	249
Unregularised Symmetric	165	[ <b>1049,4233</b> ]	162	[6429, <b>3269</b> ]	163.5
Gaussian Non Specific Regulariser	109	<b>675.2,812.3</b>	109	805.3, <b>684.4</b>	109
Gaussian Motif Specific Regulariser	98	<b>227.7,368.3</b>	98	373.4, <b>244.8</b>	98
Gaussian Module Specific Regulariser	99	<b>225.8,365.9</b>	98	371.2, <b>243.8</b>	98.5
Laplacian Non Specific Regulariser	84	<b>347,386.4</b>	102	411.6, <b>346.6</b>	93
Laplacian Motif Specific Regulariser	134	<b>41.26,479</b>	127	422, <b>55.8</b>	130.5
Laplacian Module Specific Regulariser	91	<b>223,341</b>	87	350.4, <b>234.4</b>	89
Pseudocounts Method ( $\alpha = 5.1$ )	121	<b>521.4,723.8</b>	112	708.8, <b>537.2</b>	116.5

Table 5.3: Performance of various regularisers for dataset C

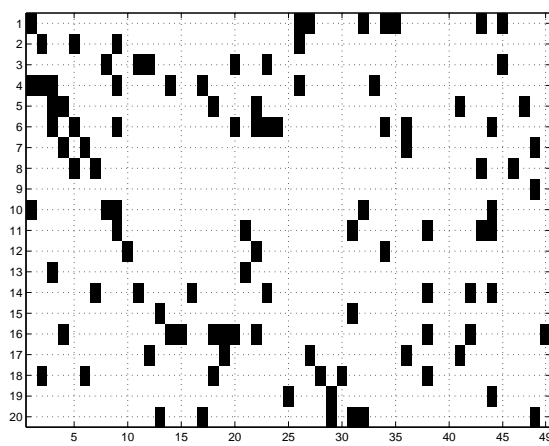


Figure 5.4: Dataset C: The true structure

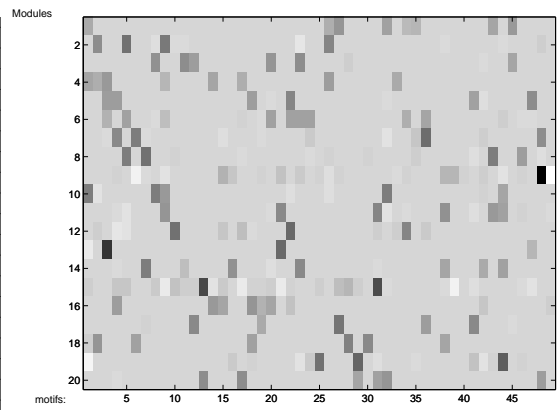


Figure 5.5: Dataset C: Laplacian Module Specific

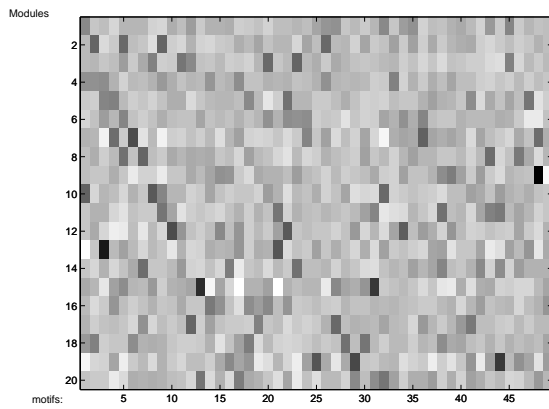


Figure 5.6: Dataset C: Unregularised Default

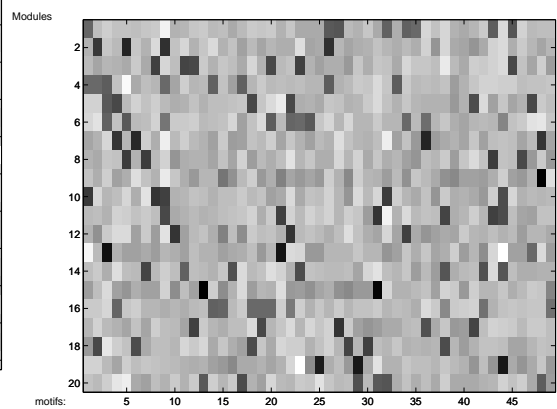


Figure 5.7: Dataset C: Data Augmentation

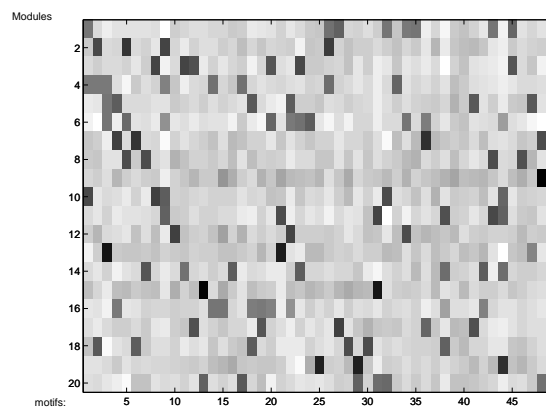


Figure 5.8: Dataset C: Gaussian Module Specific

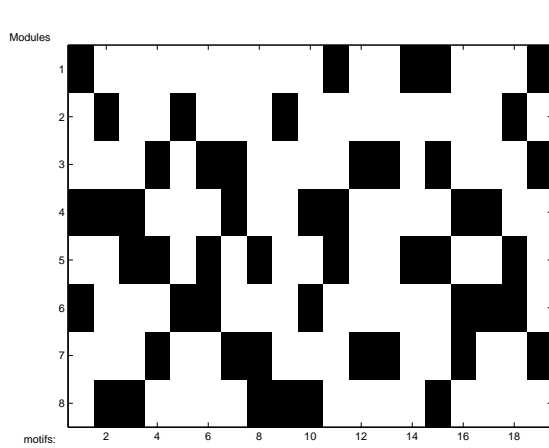


Figure 5.9: Dataset A: The true structure

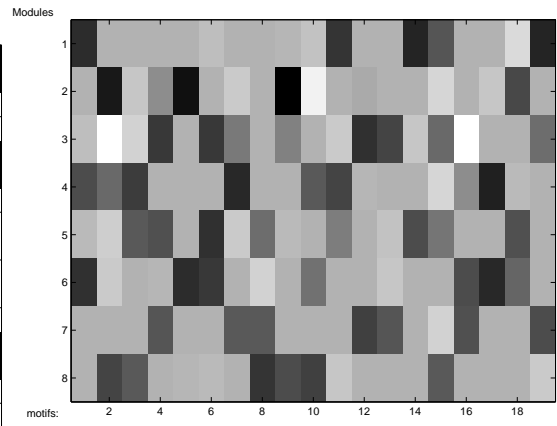


Figure 5.10: Dataset A: Laplacian Module Specific

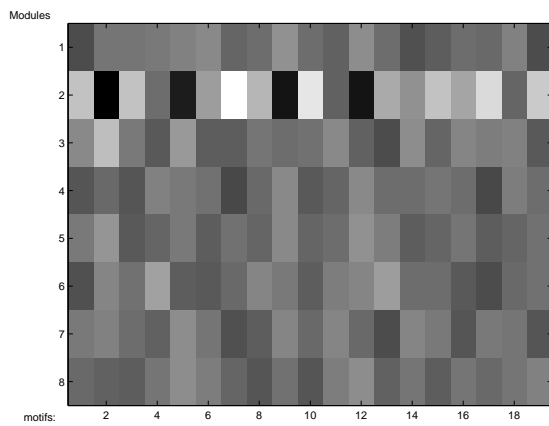


Figure 5.11: Dataset A: Unregularised Default

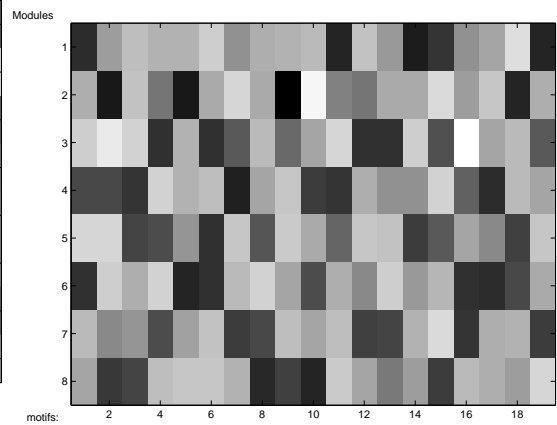


Figure 5.12: Dataset A: Data Augmentation

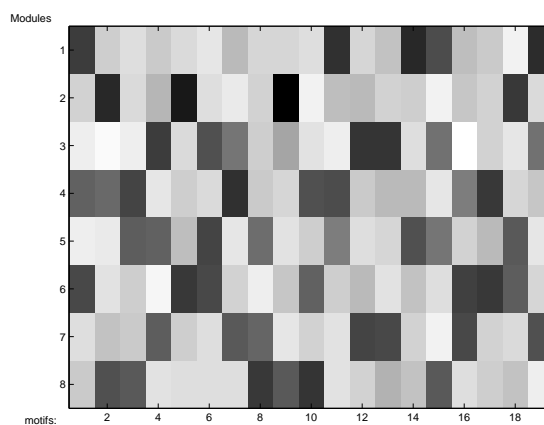


Figure 5.13: Dataset A: Gaussian Module Specific

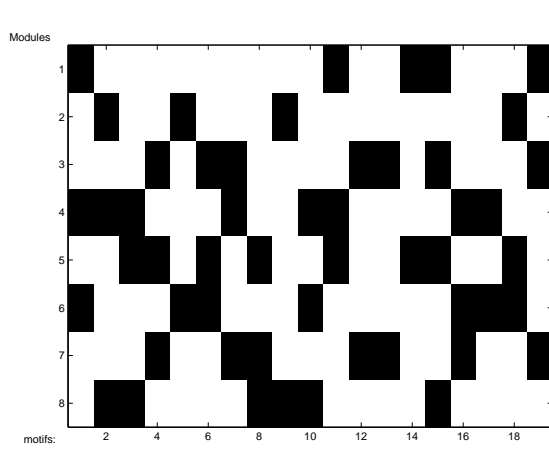


Figure 5.14: Dataset B: The true structure

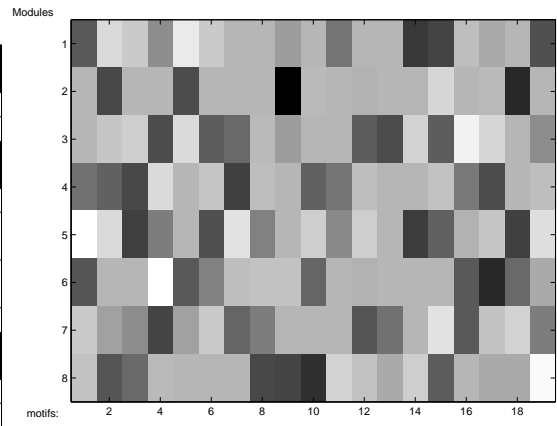


Figure 5.15: Dataset A: Laplacian Module Specific

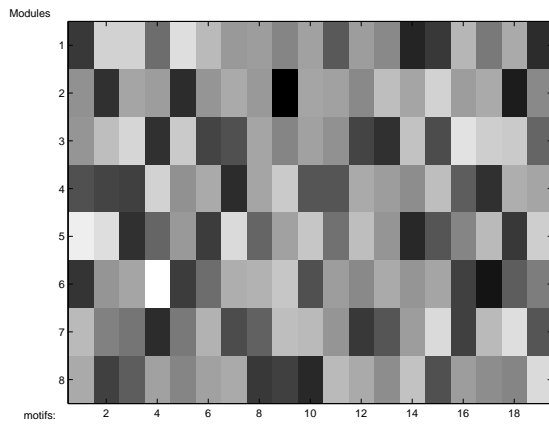


Figure 5.16: Dataset A: Unregularised Default

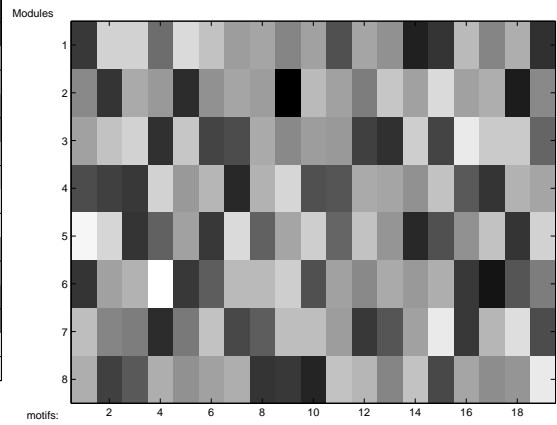


Figure 5.17: Dataset A: Data Augmentation

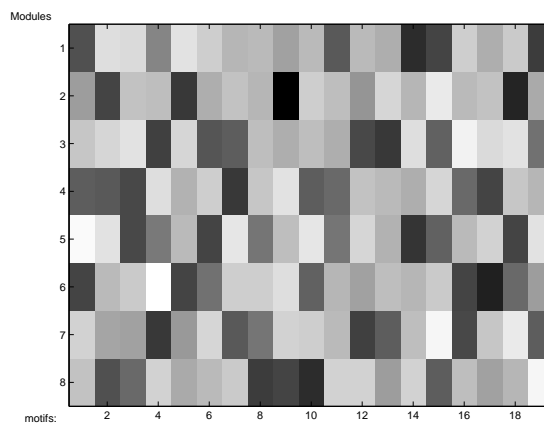


Figure 5.18: Dataset A: Gaussian Module Specific

# Chapter 6

## Biological Data

We intended to apply our model to real data from the Arabidopsis plant. Unfortunately, it has not been possible to reach final results in time. Since however the software is ready and functioning properly and all preparatory analysis had been completed, rather than omitting this section altogether, we present a brief overview of the dataset.

The Arabidopsis genome is most suitable for testing probabilistic models of gene expression and promoter sequence information since it is a widely studied, fully sequenced organism, with a relative abundance of expression data.

Segal et al. (2003) have in contrast used the *Yeast stress* dataset (Gasch et al., 2000). It would have been perhaps useful to follow them in this respect, so as to compare the effect of the improvements we claim to have made to the original model. However, the problematic character of comparing these two models is only accentuated by the transition to real datasets, where the ‘true structure’ is not known.

### 6.1 Comparing our model with the Segal model

The measure that is often used to assess performance with respect to real data is *enrichment*, which essentially represents the ability of the model to recover facts about the domain that are *known* but were not explicitly given in the data. There are two difficulties with performing comparisons using enrichment. Firstly, non-enrichment

is not a sign of poor performance. Therefore two models with identical enrichment performances are still incomparable on those grounds since the non-enriched regions can differ greatly in their depiction of the yet unknown generating process. Secondly, two models with enrichment in different parts of the data cannot be compared on these grounds, since the two models have evidently captured two different structures in the data. Which of these structures is best to capture is of course in principle impossible to agree upon. Given the above, comparisons using enrichment can only be thought of as rough guidelines. In fact, it could be argued that using enrichment as a model selection criterion can even lead to overconservative models. These problems can be sidestepped by a proper formal treatment of model selection, leading to models whose unexpected, surprising results are in principle as trustworthy as their unsurprising, ‘enriched’ ones.

## 6.2 The Dataset

We concentrate on *Arabidopsis* genes that are upregulated on viral infection, as studied by Whitnam et al. (2003). In this study, *Arabidopsis* was infected with five different RNA viruses:

- Turnip vein clearing tobamovirus (TVCV)
- Oilseed rape tobamovirus (ORMV)
- Potato Virus X potexvirus (PVX)
- Cucumber mosaic cucumovirus (CMV)
- Turnip mosaic potyvirus

Levels of *Arabidopsis* were mock inoculated with one of the five viruses. At 1, 2, 4 and 5 days after inoculation, gene expression values were measured with GeneChip microarrays (Affymetrix) containing oligonucleotide probe sets for approximately 8300 unique genes. The study found 114 unique genes that were induced in response to all five viruses at one or more points during the infection time course. This provides us

with a dataset of 20 values (fold changes: inoculation versus mock inoculation) for 114 genes.

However, certain genes were assigned multiple probes in the microarray and the results from each probe often differ. If we were clustering the expression data alone, we would have to deal with this during the preprocessing stage. Since our model is powerful enough to take the promoter sequences under consideration in its own right, we instead include all probes as distinct datapoints and let our model deal with the discrepancy. Provided the expression data for the repeated probes are not in wild disagreement, inconsistency problems, where two probes representing the same gene are assigned to different modules, are unlikely to occur: any two datapoints with identical promoter sequences are too highly correlated to be assigned to different modules. Overall, then, this leaves us with a dataset of 126 distinct datapoints (corresponding to 114 unique genes). We also discard 5 probes whose target gene is currently undetermined (denoted by ‘N/D’ in Whitnam et al. (2003)).

We retrieve the putative promoter sequences for each of these 114 genes using the Athamap interface (URL: <http://www.athamap.de>) to the TRANSFAC® public database. We follow the Athamap default value for the length of each putative promoter region as 1000 base pairs.

### 6.3 Setting up our scheme

Three different assignments were used to initialise the model. For the first run, we started off our algorithm using the clusters produced by the authors of Whitnam et al. (2003). This clustering was discovered using a hierarchical clustering algorithm applied on the expression data alone. For the second run, we exploited the classification of the genes by *function* on the basis of the GenBank protein database (Altschul et al., 1990) that is provided in Whitnam et al. (2003). For the third run, we started off with a random initialisation.

The number of modules in the two former initialisation schemes was determined by the

initialisation: 5 for the first run and 8 for the second. For the third initialisation scheme we used 5 modules to allow comparison of features with the first scheme.

For each of these initialisations, we run a seeded and a non-seeded version. For seeding, we used the MotifSampler software implementation of the Gibbs Sampling approach to the problem of motif finding, as presented in Thijs et al. (2002). We preferred this approach over the hypergeometric approach described in Segal et al. (2003), since the former includes background knowledge about the promoter sequences specific to *Arabidopsis Thaliana*.<sup>1</sup>

Finally, we had to specify the presumed length of each motif. As explained in Chapter 2, a motif length of 8 was set.

## 6.4 Results

---

<sup>1</sup>The software can be also accessed online (URL: <http://www.esat.kuleuven.ac.be/thijs/Work/MotifSampler.html>)

# Chapter 7

## Conclusion

### 7.1 Outcomes

In this thesis, we have investigated the generalisation performance and the interpretability of the model of transcriptional coregulation presented in Segal et al. (2003). We conclusively dealt with the following issues:

**Firstly** , we emphasised the difficulties in biologically interpreting the parameters of the motif model, which were not fully addressed in the paper.

**Secondly** , we revealed a fundamental flaw in the interpretation of the parameters of the regulation model by the authors of Segal et al. (2003). We then proceeded to recover the interpretability of the biologically plausible *sparsity constraint* that the authors had argued for, despite the flaw in its original motivation.

**Thirdly** , we offered a simple, but to the best of our knowledge novel geometrical interpretation of the statistical model used in the Regulation model. We then used it to motivate a novel, efficient regularising scheme to avoid overfitting, which we implemented and tested.

**Fourthly** , we offered an alternative understanding of *the sparsity constraint as a regularisation scheme*. We adapted the Bayesian pruning method of Williams (1995) to the problem of enforcing sparsity in a set of parameters with great success, in place of the heuristic method of Segal.

## 7.2 Regrets

We generally regret not having the time to perform more synthetic testing. We greatly regret that the time was not available for us to complete our project by applying our model to the *Arabidopsis* dataset of Whitnam et al. (2003). The resulting clustering would then have been compared to the one offered in Whitnam et al. (2003) and we the rich literature on *Arabidopsis* would allow us to assess the biological plausibility of either. Also, the default model could have been contrasted with the regularised and pruned version that we suggest in this thesis, in terms of their interpretability.

## 7.3 Future Work

## 7.4 Outcomes

Finally, our work suggests several hypotheses to be tested in the future. The relationship between the Tikhonov regulariser and the data augmentation scheme ought to have been explored more in depth.

# Appendix A

## Conditions for Decomposability of the weights for the Motif Model

We are looking for the conditions that need to be imposed on three positive numbers  $A, B$  and  $C$  so that there exist two distributions given by  $a, b, c$  and  $a', b', c'$  respectively such that:

$$A = \frac{a}{a'}, \quad B = \frac{b}{b'}, \quad C = \frac{c}{c'} \quad (\text{A.1})$$

Since the normalisation conditions are the only constraints we have, substituting them in A.1 quickly gives us all the constraints that are necessary and sufficient for a solution to exist: a solution to A.1 exists if and only if there exist numbers  $b'$  and  $c'$  between 1 and 0 that satisfy:

$$b' + c' < 1 \text{ and } b'(A - B) + c'(A - C) = A - 1 \quad (\text{A.2})$$

This follows by the two normalisation conditions taken in joint consideration:

$$\text{Normalisation condition} \Leftrightarrow a + b + c = 1 \Leftrightarrow \text{By A.1}$$

$$Aa' + Bb' + Cc' = 1 \Leftrightarrow \text{Normalisation condition}$$

$$A(1 - b' - c') + Bb' + Cc' = 1 \Leftrightarrow$$

$$\text{A.2}$$

If we find any two such numbers, then  $a'$  is given by normalisation constraints and  $a, b, c$  are given by A.1.

Now assume that  $A, B, C$  satisfy the following criterion:

$$\begin{aligned} &\text{Either } \{A = B = C = 1\} \\ &\text{or } \{ \text{and } \min(A, B, C) < 1 \text{ and } \max(A, B, C) > 1 \} \end{aligned} \tag{A.3}$$

We will prove that this condition is sufficient and necessary. We break down the proof in cases.

**Case 1**  $A, B, C \leq 1$  and  $\max(A, B, C) < 1$

In this case we get  $A - 1 < 0$  which then implies that either  $A < B$  or  $A < C$  but not both. So assume wlog that  $B \leq A < C \leq 1$ . Since  $A - B \leq 0$  we have:

$$\begin{aligned} c'(A - C) &= A - 1 + b'(A - B) \leq A - 1 \quad \Rightarrow \quad (\text{Since } A - C < 0) \\ c' &\geq \frac{A - 1}{A - C} \end{aligned} \tag{A.4}$$

But  $1 > C$  so  $A - 1 < A - C < 0$  which implies that  $c' = \frac{A-1}{A-C} > 1$ . We have therefore reached a contradiction and the case  $A, B, C \leq 1$  but  $\min(A, B, C) < 1$  is excluded.

**Case 2**  $A, B, C \geq 1$  and  $\min(A, B, C) > 1$

This case follows by symmetry.

We have therefore proved necessity since the only two alternatives left are the two mutually exclusive clauses of condition A.3. We now prove sufficiency. Again we consider two cases.

**Case 1**  $A = B = C = 1$

In this case any numbers  $b'$  and  $c'$  will do, since:

$$\begin{aligned} A - B &= A - C = A - 1 = 0 \\ b'(A - B) + c'(A - C) &= A - 1 = 0 \end{aligned} \tag{A.5}$$

**Case 2**  $\min(A, B, C) < 1$  and  $\max(A, B, C) > 1$

This is slightly harder to show. Recall that all we need to show is that there are

numbers  $b'$  and  $c'$  that satisfy A.2 and A.1. The way we approach this is by picking  $c'$  appropriately so that the value of  $b'$  that is derived by solving A.2 satisfies all constraints. Wlog assume that  $A = \max(A, B, C)$  and  $C = \min(A, B, C)$ . Then we only need to establish there exists a number  $c'$  that satisfies the following three statements:

- (I).  $c'(C - B) < (1 - B)$
- (II).  $0 < c' < 1$
- (III).  $1 - A < -c'(A - C) < 1 - B$

First we explain why these three conditions are all we need to prove. Assume we have such a number  $c'$ . There are two cases to consider. First, assume  $A = B$ . Then  $b'$  can be chosen as any positive number satisfying  $b' + c' < 1$ , since it is left undetermined from A.2. Then assume  $A \neq B$ . We can now define  $b' = [(A - 1) - c'(A - C)] / (A - B)$ . Since  $A - B > 0$  and  $A - 1 > c'(A - C)$  by (III) then  $b' > 0$ . Now consider the sum  $b' + c'$ :

$$\begin{aligned}
 b' + c' &= [(A - 1) - c'(A - C)] / (A - B) + c' \\
 &= [(A - 1) - c'(A - C) + c'(A - B)] / (A - B) \\
 &= \frac{A - 1 + c'(C - B)}{A - B}
 \end{aligned} \tag{A.6}$$

But  $A - 1 + c'(C - B) < A - 1 + 1 - B$  by (I). Therefore  $b' + c' = \frac{A - 1 + c'(C - B)}{A - B} < 1$  and we are done.

Finally, we now prove that there exists a number  $c'$  that satisfies (I), (II) and (III). The derivation that follows is unfortunately tedious.

Since  $C - B < 0$  (I) becomes  $c' > \frac{1 - B}{C - B}$ . Also:

$$\begin{aligned}
 1 &> C && \Leftrightarrow \\
 1 - B &> C - B && \Leftrightarrow \\
 \frac{1 - B}{C - B} &< 1
 \end{aligned} \tag{A.7}$$

So far we have:

$$0, \frac{1 - B}{A - C} < c' < 1 \tag{A.8}$$

Since  $A - C > 0$ , (III) becomes:

$$\frac{1-B}{A-C} < c' < (1-A)(A-C) \quad (\text{A.9})$$

But we have:

$$1-A < 0 < B-C \quad \Leftrightarrow$$

$$1-B < A-C \quad \Leftrightarrow \quad (\text{A.10})$$

$$\frac{1-B}{A-C} < 1$$

and  $\frac{1-A}{A-C} > 0$ , since  $A > 1$  and  $A > C$ . Finally, observe that:

$$\frac{1-A}{A-C} > \frac{1-B}{C-B} \quad \Leftrightarrow \text{since } (A > C)$$

$$1-A > \frac{(A-C)(1-B)}{C-B} \quad \Leftrightarrow \text{since } (C < B)$$

$$(1-A)(C-B) < (A-C)(1-B) \quad \Leftrightarrow$$

$$C - CA - B + BA < A - C - BA + CB \quad \Leftrightarrow$$

$$C - CA + C - CB < A - BA - BA + B \quad \Leftrightarrow$$

$$C(1-A) + C(1-B) < A(1-B) + B(1-A) \quad \Leftrightarrow$$

$$C(1-A) + C(1-B) < A(1-B) + B(1-A) \quad \text{which holds since } B, C < A \quad (\text{A.11})$$

So it is possible to pick  $c'$  such that  $0, \frac{1-B}{C-B}, \frac{1-B}{A-C} < c' < \frac{1-A}{A-C}, 1$

# Bibliography

- Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment tool. *Journal of Molecular Biology*, 215:403–410.
- Barash, Y., Elidan, G., Friedman, N., and Kaplan, T. (2003). Modeling dependencies in protein-dna binding sites.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–38.
- Gasch, A. P., Spellman, P. T., Kao, C. M., Carmel-Harel, O., Eisen, M. B., Storz, G., Botstein, D., and Brown, P. O. (2000). Genomic expression program in the response of yeast cells to environmental changes. *Molecular Biology of the Cell*, 11:4241–4257.
- Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic roc curve. *Radiology*, 143:29–36.
- Jensen, S. T., Liu, X. S., Zhou, Q., and Liu, J. S. (2004). Computational discovery of gene regulatory binding motifs: A bayesian perspective. *Statistical Science*, 19(1):188–204.
- Jordan, M. I., Ghahramani, Z., Jaakkola, T., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233.
- M.Bishop, C. (1995). Training with noise is equivalent to tikhonov regularisation. *Neural Computation*, 7(1):108–116.

- Segal, E., Yelensky, R., and Koller, D. (2003). Genome-wide discovery of transcriptional modules from dna sequence and gene expression. *Bioinformatics*, 19(Suppl 1):273–282.
- Thijs, G., Lescot, M., Marchal, K., Rombauts, S., Moor, B. D., Rouzé, P., and Moreau, Y. (2002). A gibbs sampling method to detect over-represented motifs in upstream regions of coexpressed genes. *Journal of Computational Biology (special issue Recomb'2001)*, 9(2):447–464.
- Whitnam, S. A., Quan, S., Chang, H.-S., Cooper, B., Estes, B., Zhu, T., Wang, X., and Hou, Y.-M. (2003). Diverse rna viruses elicit the expression of common sets of genes in susceptible *Arabidopsis thaliana* plants. *The Plant Journal*, 33:271–283.
- Williams, P. M. (1995). Bayesian regularization and pruning using a laplace prior. *Neural Computation*, 7:117–143.