Approximate Inference, Structure Learning and Feature Estimation in Markov Random Fields

Pradeep Ravikumar

August 2007
CMU-ML-07-115

Approximate Inference, Structure Learning and Feature Estimation in Markov Random Fields

Pradeep Ravikumar

August 2007
CMU-ML-07-115

Machine Learning Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Keywords: Markov Random Fields, Graphical Models, Approximate Inference, Structure Learning, Feature Estimation, Non-parametric Estimation, Sparsity, ℓ_1 Regularization, Additive Models

Abstract

Markov random fields (MRFs), or undirected graphical models, are graphical representations of probability distributions. Each graph represents a family of distributions – the nodes of the graph represent random variables, the edges encode independence assumptions, and weights over the edges and cliques specify a particular member of the family.

There are three main classes of tasks within this framework: the first is to perform inference, given the graph structure and parameters and (clique) feature functions; the second is to estimate the graph structure and parameters from data, given the feature functions; the third is to estimate the feature functions themselves from data.

Key inference subtasks include estimating the normalization constant (also called the partition function), event probability estimation, computing rigorous upper and lower bounds (interval guarantees), inference given only moment constraints, and computing the most probable configuration.

The thesis addresses all of the above tasks and subtasks.

Acknowledgements

I will start with a couple of historical notes. For the first, we go to the post-Renaissance period, to the German polities. And for the second, we go to 2000 B.C. in India. These are the roots of the research university system, and the roots of the Indian monasteries respectively. What is common to both, is an environment of learned wise people, who devote their ascetic lives to understanding the world and existence. I apologize for the bombast, but I'm not done with the history and culture lesson yet. India has many good things; among them is a multitude of holy men who devote their life to introspection and religion. A non-Indian might perhaps not fully grasp this, but my thesis advisor John Lafferty has a lot of an intangible something common to the best of such holy men; an inner peace, wisdom. In the finest of Indian traditions dating back to 2000 B.C., I have learnt under a holy wise man. My parents must be very proud. In his other professorial role, John has taught me, as much as I could learn at the least, a lot about how to think about research; I've benefited a lot from his clarity of thought and creative intellect.

I have much to thank other monks at Carnegie Mellon as well. William Cohen showed me how to zoom in when doing research; Steve Fienberg showed me how to zoom out for the big picture. I've had a lot of fun working with Larry Wasserman, who is as insightful with acronyms as he is with All of Statistics. I've benefitted not only from his insights, but also, at a fundamental level, from his style of doing research; which has greatly influenced my research style as well.

I must also thank my thesis committee. The creative energies of Carlos Guestrin and Eric Xing have always fascinated me, and I was lucky to be the TA of a course they jointly offered, “Probabilistic Graphical Models”; their insights were quite helpful. Martin Wainwright's research on graphical models, and his beautifully written papers, were principal motivators for this thesis and my research.

I'm thankful to Diane Stidle who has been an ever-present source of help and advice. I'm also grateful for the kindness and help of Sharon Cavlovich and Monica Hopes.

High-fives to my squash team; Vineet Goyal, Mohit Kumar, Matt Mason; for all the good times playing and winning the league.

Finally, I thank my parents Usha and Pattabhiraman Ravikumar, for their love and their encouragements.

Contents

Chapter 1

Introduction

The task of prediction; estimating an output or response given an input; predates humans. Our large brains, evolutionary biologists insist, evolved in part to solve this onerous task. These brains, though large, are very “stochastic” and fragile and seem to have limited computational powers. This, and perhaps mere curiosity, led to the conceit of Artificial Intelligence; to solve this prediction task using not the evolved large brains but using mathematics. Unfortunately, due to the continuing influences of the Enlightenment period, for a while AI scientists equated reasoning with “rational” deduction; and tried to come up with if-then-else rules for various prediction tasks. More modern influences such as quantum physics made many appreciate the use of probabilistic machinery, not just for modeling uncertainty but for making efficient inductive prediction possible at all.

Thus we arrive at the task of probabilistic inference: to predict using a statistical model; a model which describes the probabilistic relationship between the input and the response. An elementary way to represent such a relationship is a random field, or distribution, over the input and the response. A general stochastic system has many variables of interest, not just a particular input and response; such a system then can be modeled by a random field over the variables characterizing the system.

In [18], Pedro Domingos defines an “interface layer” for any field of research as an intermediate layer that provides an easy language for applications above the layer and which would be “implemented” by infrastructure below the layer. This would make any innovation in the infrastructure immediately available to applications above the interface layer. In programming systems for instance, high-level languages act as an interface between the infrastructure of compilers and code optimizers below, and the application programs above. In AI and machine learning then, a framework for representing a random field over the stochastic variables of a system, and which would allow efficient inference, can act as an interface layer. Probabilistic graphical models, to a certain extent, serve such a purpose.

Probabilistic graphical models, as the name suggests, borrow from both probability theory and graph theory. In this framework, the conditional independences among the random variables of the system are represented by the edges of a graph; in particular, a distribution is specified by functions over the cliques (fully connected components) of the graph. When this graph is undirected, these are called undirected graphical models. This modular and graphical nature of the representation offers not only a visually intuitive view of the stochastic interactions in a system, but also the ingredients enabling a good "infrastructure" layer: graph-theoretic and related combinatorial techniques are naturally available for inference and prediction. As befits an interface layer, applications abound, and include medical diagnosis, error correcting codes, control and tracking problems, image and speech processing, bio-informatics, statistical mechanics, social networks and contingency table analysis: evolution would be most proud of this computational brain. The next section gives an overview of this framework.

1.1 Representation theory

Another term for undirected graphical models is Markov random fields and the reason why forms the essence of this representation theory section. Let us first parse the "Markov random fields" phrase. In physics, a field is an assignment of a physical quantity to points in space-time. For instance, a gravitational field is an assignment of a gravitational vector to points in space-time. Consider now a p-dimensional space, spanned by values of p random variables instead of just the four of space and time. A random field is an assignment of a probability measure to points in the $p-$dimensional space. Just as a gravitational field describes a gravitational system, a random field describes a stochastic system. Thus a random field with a compact representation, and accessible inference procedures can be used as an interface layer for stochastic system applications.

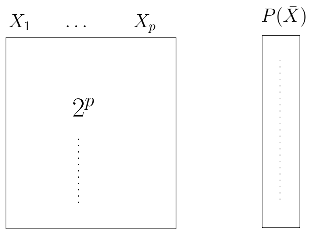


Figure 1.1: Random field over binary variables

The key aspect is compact representation. Figure (1.1) shows a brute-force "table" representation for a random field over p binary valued random variables $\{X_1, \ldots, X_p\}$. The number of configurations is 2^p, and storing an assignment of a probability measure to each of these would lead to a table of size 2^p. This is quite large for large p, and this when variables are just binary valued, instead of taking values from a larger set, or even being continuous valued. Even simple inference tasks such as computing $\Pr(X_1 = 1)$ would require accessing $O(2^p)$ entries of the table.

Markov random fields use Markov assumptions to give compact representations for random fields. A common example of a Markov property is the first order Markov property of a Markov chain (Figure (1.2)). This asserts that the future variables are conditionally independent of the past variables given the present variable. This can be generalized to graphs other than a chain. Let $G = (V, E)$ denote an undirected graph, with V the set of nodes and E the set of undirected edges. Let X_i denote the variable associated with node i, for $i \in V$; giving a collated random vector $X = \{X_1, \ldots, X_p\}$. The local Markov property for variable X_i states that X_i given its set of neighbors $X_{N(i)}$ is conditionally independent of the rest of the variables. Figure (1.3) shows an example. This in turn is generalized by the global Markov property which is defined as follows. A *separating set* of nodes in a graph G is a set of nodes which when removed disconnect the graph. Let A and B be the components a separating set $\mathcal{S}$ disconnects. The global Markov property states that variables X_A in A are conditionally independent of variables X_B in B given the separating variables $X_\mathcal{S}$ in $\mathcal{S}$.

$$X_A \perp_p X_B | X_\mathcal{S} \tag{1.1}$$

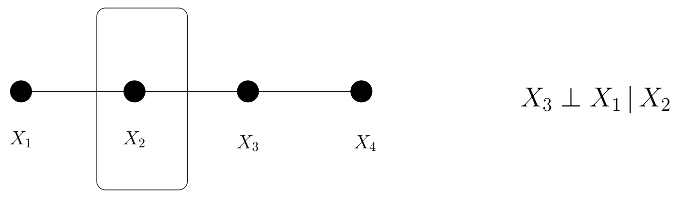
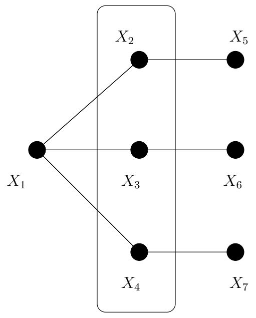


Figure 1.2: First order Markov property

This brings us to the definition of a Markov random field over a graph G: it is the set of distributions which satisfy the set of all global Markov properties for the graph G.

As stated, it might be unclear how this conditional independence formalism, while intuitive, leads to a compact representation; in fact, it might seem unwieldly for the purposes of inference.



$$X_1 \perp X_5 \,|\, (X_2, X_3, X_4)$$

Figure 1.3: Local Markov property

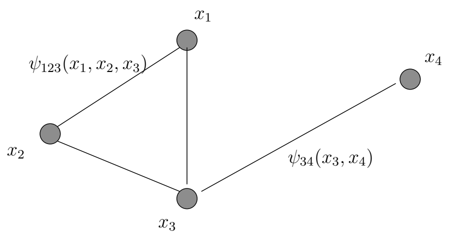
The following theorem by Hammersley and Clifford [35] however specifies an equivalent algebraic condition that any distribution from a graphical model family must satisfy.

Theorem 1. *(Hammersley and Clifford) A positive probability distribution P over $X = \{X_i,\, i \in V\}$ satisfies the global Markov properties for a graph $G = (V, E)$ if and only if it factorizes according to the set of cliques (fully connected components) $\mathcal{C}$ in G,*

$$P(X) \propto \prod_{C \in \mathcal{C}} \psi_C(X_C) \tag{1.2}$$

where ψ_C is a function that depends only on the variables $\{X_i,\, i \in C\}$.

The Hammersley Clifford theorem translates the knowledge of conditional independencies in a stochastic system into a compact representation for a random field. To see this, consider a distribution over $|V| = p$ random variables – this is a $p-$ variate function, but the graphical model represents this distribution as a collection of clique functions, each of which depends on a smaller subset of variables. Figure (1.4) gives an example.



$P(X) \propto \psi_{123}(x_1, x_2, x_3)\psi_{34}(x_3, x_4)$

Four variate function $\rightarrow$ Three, Two variate function

Figure 1.4: Hammersley Clifford theorem: compact representation

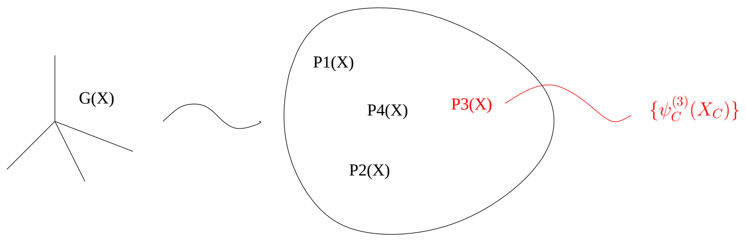


Figure 1.5: Undirected graphical models

The representation theory of graphical models is summarized in Figure (1.5). A graph denotes a family of distributions, each of which satisfies the set of all Markov properties of that graph. An instance of this family is specified by a set of clique functions over the cliques of the graph.

1.1.1 Exponential Family Representation

A product of positive functions can also be written as the exponential of a sum of functions,

$$\begin{aligned} P(X) &\propto \prod_{C \in \mathcal{C}} \psi_C(x_C) \\ &\propto \exp\left(\sum_{C \in \mathcal{C}} \log \psi_C(X_C \right) \end{aligned} \tag{1.3}$$

This motivates the exponential family representation. Let $\phi = \{\phi_\alpha, \alpha \in \mathcal{C}\}$ denote a set of feature functions or *potential* functions, for an index set $\mathcal{C}$. Associated with ϕ is a vector of parameters $\theta = \{\theta_\alpha, \alpha \in \mathcal{C}\}$. With this notation, the exponential family of distributions of X, associated with ϕ, is given by

$$P(x; \theta) \;=\; \exp\left(\sum_\alpha \theta_\alpha \phi_\alpha - \Phi(\theta) \right). \tag{1.4}$$

where $Z = \exp \Phi(\theta)$ is the normalization constant; also called the partition function. We will typically focus on the logarithm of this normalization constant, the log-partition function $\Phi(\theta)$.

Equations (1.3),(1.4) show that a graphical model distribution with clique factors $\{\psi_C(X_C)\}$ can be represented by an exponential family distribution, with feature functions $\{\log \psi_C\}$, and unit parameters. When the variables are discrete-valued, we can represent any graphical model family – not just a distribution within that family – with an exponential family with indicator function features, as follows.

Let $\mathcal{X}_i$ denote the domain of variable X_i. An *indicator function* of an event is one if that event occurs and zero otherwise. Node value indicator functions are thus given by

$$\mathbb{I}_k(X_i) = \begin{cases} 1 & \text{if} \quad X_i = k \\ 0 & \quad\text{o.w.} \end{cases} \tag{1.5}$$

Any potential function $\phi(X_C)$ can thus be represented as a linear combination of indicator functions,

$$\phi_C(X_C) = \sum_{x_C} \phi_C(x_C)\, \mathbb{I}_{x_C}(X_C) \tag{1.6}$$

This allows us to represent any graphical model family, for a given graph, by the exponential family with the clique indicator functions $\{\mathbb{I}_{x_C}(X_C)\}$ as features,

$$\begin{aligned} p(X;\theta) &\propto \exp\left(\sum_C \theta_C\, \phi_C(X_C)\right) \\ &\propto \exp\left(\sum_C \sum_{x_C} \theta_C\, \phi_C(x_C) \quad \mathbb{I}_{x_C}(X_C)\right) \\ &\propto \exp\left(\sum_C \sum_{x_C} \theta'_{C,x_C} \quad \mathbb{I}_{x_C}(X_C)\right) \end{aligned}$$

1.2 Pairwise MRFs

As discussed in [68], at the expense in increasing the state space one can assume without loss of generality that the graphical model is a pairwise Markov random field, *i.e.*, the set of cliques I is the set of edges $\{(s,t) \in E\}$. In most of what follows, we shall thus assume a pairwise random field. Note that this would allow us to express the potential function and parameter vectors in more compact form as matrices:

$$\Theta := \begin{pmatrix} \theta_{11} & \dots & \theta_{1n} \\ \vdots & \vdots & \vdots \\ \theta_{n1} & \dots & \theta_{nn} \end{pmatrix} \Phi(x) := \begin{pmatrix} \phi_{11}(x_1,x_1) & \dots & \phi_{1n}(x_1,x_n) \\ \vdots & \vdots & \vdots \\ \phi_{n1}(x_n,x_1) & \dots & \phi_{nn}(x_n,x_n) \end{pmatrix} \tag{1.7}$$

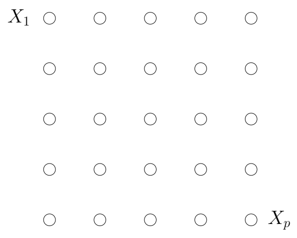
We will denote the trace of the product of two matrices A and B by the inner product $\langle\langle A, B\rangle\rangle$. The normalization constant would thus be given by $\Phi(\Theta) = \sum_{x\in\mathcal{X}} \exp\langle\langle\Theta, \Phi(x)\rangle\rangle$.

Till now we have referred only to undirected graphs and undirected graphical models; graphical models can also have directed graph representations, these are more commonly known as Bayesian networks. In this thesis we focus however on undirected graphical models, and so in what follows, we continue to refer to undirected graphical models even when we omit the phrase "undirected".

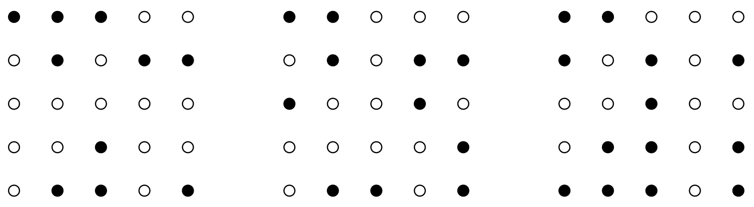
1.3 Tasks in a graphical model

To use this graphical model framework for prediction, the main tasks are to first build a graphical model from observed data, and then to perform the prediction tasks using the built model. These tasks, which we now describe, are typically intractable, and the contribution of this thesis is a set of techniques to perform them tractably, if approximately.

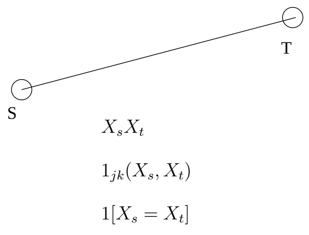
A domain expert first lists the random variables of the given stochastic system.



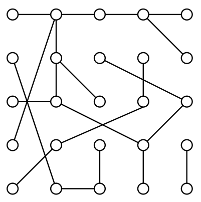
Data, consisting of multiple (perhaps partially observed) i.i.d. samples of the random variables, is observed.



Then comes the specification of the feature functions, which are the functions over potential cliques of the graph. These are typically specified by the domain expert; who either hand-designs them or uses standard functions such as Ising, Potts or indicator functions for discrete-valued models; they can also be estimated from data – this is the feature estimation task.



Given the features, the next task is to specify the graph structure. Here too, the graph structure can either be specified by the domain expert (grid, chain, etc.), or it can be learned from data – this is the structure learning task.



Given the features and the structure, the next task is to learn the parameters, which are the weights over the clique feature functions. These are typically learn from data – this is the parameter learning task. These three tasks specify the graphical model distribution; which we can now use for *inference*.

Inference is the task of querying the graphical model. Not any query at large, but queries about the distribution represented by the graphical model. The basic inference tasks are as follows.

Computing the log partition function: The partition function is the normalization constant of the graphical model distribution. While this serves an aesthetic purpose – a distribution without its normalization constant seems amiss – it is also required to compute the probabilities of assignments.

Event probability estimation: This is the most natural query to a random field; to compute the probability of an event involving the random variables of the graphical model. A common example is the probability of a marginal, which is the event of setting a subset of nodes to a particular value, e.g. $Pr(X_i = 1)$.

Computing upper and lower bounds: Applications might require some guarantees for the approximate estimates of the event probabilities. Computing rigorous upper and lower bounds for the event probabilities gives an interval in which the true event probability lies – and provides just such a guarantee.

Inference given moments: Here, we are not given the distribution parameters; just the expected values (moments) of a given set of functions. The task however is no less: that of computing event probabilities.

Estimating the *MAP* configuration: Given an assignment of values to a subset of the random variables, the maximum a posteriori or MAP configuration is the most probable assignment of values to the rest of the variables.

1.4 What this thesis is about

In this thesis, we address all the tasks listed above; the three inference tasks, the structure learning task as well as the feature estimation task. The last two tasks greatly lighten the load of the domain expert, who is now required to merely list the random variables of the system; given data, the procedures detailed in this thesis, as well as allied procedures in the literature, can then be used to construct a graphical model, and perform efficient, albeit approximate, inference on those estimated models.

In inference, as noted above, there are three basic subtasks. To approximate the log partition function, we propose preconditioner approximations (Chapter 3). To compute the MAP configuration, we propose a quadratic programming relaxation (Chapter 4). To estimate general event probabilities, we propose (a) variational Chernoff bounds and (b) variational Chebyshev-Chernoff bounds (Chapters 6 and 7). As the names suggest, we propose rigorous upper and lower bounds for the general event probabilities. Even approximation in graphical models is NP-hard; if one requires a constant-factor approximation. Upper and lower bounds provide an interval approximation instead; and specify an interval in which the true event probability lies. It is hoped that such guarantees enhance the appeal of graphical models as an interface layer. The Chernoff bounds require the distribution parameters, whereas the Chebyshev bounds require just the expected values or moments of a given set of functions.

For structure learning, we investigate procedures based on edge-appearance parameterizations (Chapter 8) and ℓ_1 regularized regression (Chapter 9). For feature estimation, we propose additive conditional random fields (aCRFs); a class of models which allow efficient estimation of feature functions from data given the structure (Chapter 11), and sparse additive models (SpAM); a class of models which allow simultaneous predictor selection and feature estimation from data (Chapter 12).

All is joint work with John Lafferty. The ℓ_1 regularized regression work is also joint with Martin Wainwright; aCRF with Douglas Vail; and SpAM with Han Liu and Larry Wasserman.

Part I

Approximate Inference

Chapter 2

Log Partition Function

In this chapter, we will briefly review the task of estimating the log-partition function, or normalization constant, of an undirected graphical model. Revisiting our notation; let $G = (V, E)$ denote an undirected graph, with V the set of nodes, and E the set of edges, and let $X = \{X_s,\ s \in V\}$ denote the random variable associated with the graphical model. Letting $\mathcal{C}$ denote the set of cliques of the graph, the graphical model distribution of X is given by,

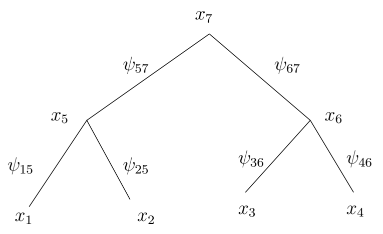
$$p(X) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(X_C)$$

We focus on pairwise graphical models, so that $\mathcal{C} = E$, the set of edges;

$$p(X) = \frac{1}{Z} \prod_{(s,t) \in E} \psi_{st}(X_s, X_t) \tag{2.1}$$

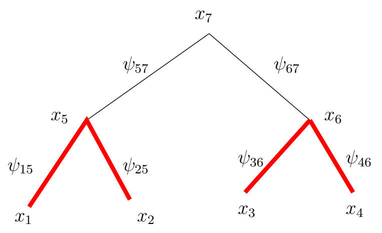
The task is to estimate the partition function, $Z = \sum_x \prod_{(s,t)\in E} \psi_{st}(x_s, x_t)$. As it stands this is a sum over exponentially many assignments; with $p = |V|$ nodes, the number of assignments is 2^p if the variables are binary valued.

At the very heart of the representation theory of graphical models lies a graph. This suggests graph-theoretic techniques as the first line of attack. Variable elimination [70] (and its extensions) is just such an algorithm. Consider the tree-structured graphical model in the figure below with nodes $X_1, \ldots, X_4$ as the leaves, X_5, X_6 as the first level nodes, and X_7 as the root. The graphical model is pairwise, with potential functions on the edges.



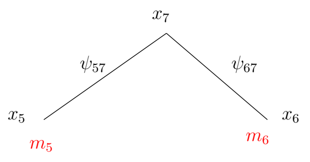
$$\sum_{x_1,\ldots,x_7} \psi_{15}(x_1,x_5)\psi_{25}(x_2,x_5)\psi_{36}(x_3,x_6)$$
$$\psi_{46}(x_4,x_6)\psi_{57}(x_5,x_7)\psi_{67}(x_6,x_7)$$

In variable elimination, instead of a blind-force sum over exponentially many configurations, we *eliminate* variables one at a time: we sum over the values of the variable being eliminated to leave a new factor which only depends on the rest of the variables. An appropriately constructed elimination order can greatly reduce the computations required. In the given example, the graph suggests the following elimination order. First we eliminate the leaves,



$$\sum_{x_7} \quad \sum_{x_5} \psi_{57} \left(\sum_{x_1} \psi_{15} \sum_{x_2} \psi_{25} \right)$$
$$\sum_{x_6} \psi_{67} \left(\sum_{3} \psi_{36} \sum_{x_4} \psi_{46} \right)$$

Eliminating the leaf nodes leaves us with factors over the first level nodes, which are now the new leaves.



$$\sum_{x_7} \sum_{x_5} \psi_{57}\, m_5$$
$$\sum_{x_6} \psi_{67}\; m_6$$

where

$$m_5(x_5) = \left(\sum_{x_1} \psi_{15}(x_1,x_5) \right) \left(\sum_{x_2} \psi_{25}(x_2,x_5) \right)$$
$$m_6(x_6) = \left(\sum_{x_3} \psi_{36}(x_3,x_6) \right) \left(\sum_{x_4} \psi_{46}(x_4,x_6) \right)$$

Eliminating these in turn leaves us with a factor m_7 over the root, where

$$m_7(x_7) = \left(\sum_{x_5} m_5(x_5)\psi_{57}(x_5, x_7)\right)\left(\sum_{x_6} m_6(x_6)\psi_{67}(x_6, x_7)\right)$$

Eliminating the root in the end, yields the partition function,

$$Z = \sum_{x_7} m_7(x_7)$$

In the computations above, we computed sums over assignments of single variables at a time, the time required is thus linear instead of exponential in the number of nodes; we could do this because the graph was a tree. Gathering connected nodes into clusters and forming a tree of such node clusters, over which one then performs variable elimination, forms the main idea behind the junction tree algorithm [35]. However its complexity is exponential in the size of the largest node cluster formed, a quantity also called the tree-width. Thus, performing exact inference using graph-theoretic techniques is tractable only for sparse graphs.

This motivates the “projection optimization” paradigm: approximate the given complex model by a simpler model, for which exact inference is possible. The task of approximate inference, under this paradigm, reduces to the task of obtaining a simpler graph and its parameters thereof, for a simpler graphical model. This is typically not that Faustian a bargain: even in complex graphs, averaging phenomena can decouple nodes leading to good approximate simpler graphical models, perhaps with altered parameter settings. In short, the combinatorial task of partition function estimation is replaced with the following programme: given a complex model; and a candidate set of simpler models, compute the model from the candidate set which minimizes a divergence measure with the original complex model. A commonly used measure is the KL divergence measure; most approximate inference techniques are a combination of selecting candidate sets of simpler models, and approximating the KL divergence measure itself. Adopting the notation of [68], if $\{b(x)\}$ is the simpler model distribution, and $\{p(x)\}$ is the given complex model, then the KL divergence measure is given by,

$$D(\{b\}\|\{p\}) = \sum_x b(x)\log b(x) - \sum_x b(x)\log p(x) \tag{2.2}$$

If $\{q(x)\}$ is the given unnormalized distribution, then,

$$D(\{b\}\|\{p\}) = \sum_x b(x)\log b(x) - \sum_x b(x)\log q(x) + \log\sum_x q(x) \tag{2.3}$$

Since the last term does not depend on $\{b\}$, we need optimize just the first two terms; various approximation procedures such as belief propagation and mean field thus have two components: approximation for the entropy term (which when added with the second term is called the "free energy"), and a candidate set of approximate models. We refer to [68] for further details.
The process of obtaining a simpler model from a complex model reduces the degrees of freedom. We would then want to simultaneously introduce extra "parameters" to take up the slack. This loosely characterizes the methodology of variational methods; the extra variational parameters are optimized over to obtain a candidate simpler model. The earlier divergence minimization techniques can also be cast as variational methods. In the next section, we give the convex dual characterization of the log-partition function, which lies at the heart of variational methods.

2.1 Conjugate Dual of the log-partition function

An exponential family distribution with potential function $\phi(x)$ and parameter θ is given by,

$$p(X;\theta) = \exp(\theta^\top \phi(X) - \Phi(\theta)) \tag{2.4}$$

where $\Phi(\theta)$ is the logarithm of the normalizing constant of the model; the log-partition function. It is a convex function of θ satisfying $\partial\Phi(\theta)/\partial\theta_\alpha = E_\theta\left[\phi_\alpha(X)\right]$. The convex conjugate Φ^* is defined by $\Phi^*(\mu) = \sup_{\theta\in\mathbb{R}^m} \langle \mu, \theta\rangle - \Phi(\theta)$. If $\hat{\theta} = \theta(\mu)$ is the parameter attaining the supremum, a calculation shows that $\Phi^*(\mu)$ can be expressed as a negative entropy $\Phi^*(\mu) = \sum_x p(x \mid \hat{\theta}) \log p(x \mid \hat{\theta})$ and $\mu_\alpha = E_{\hat{\theta}}\left[\phi_\alpha(X)\right]$. These relations show that the dual parameters μ are the set of vectors that can be realized as moments of ϕ. The collection of such dual parameters is the *marginal polytope*

$$\begin{aligned} \mathrm{MARG}(G,\phi) \;&= \\ &\left\{ \mu \in \mathbb{R}^m \mid \sum_x p(x \mid \theta)\, \phi(x) = \mu \text{ for some } \theta \in \mathbb{R}^m \right\} \end{aligned} \tag{2.5}$$

and plays a central role in the analysis of $\Phi(\theta)$. Since $\mathcal{X}$ is finite, the closure of $\mathrm{MARG}(G,\phi)$ is a finite intersection of halfspaces, and is thus indeed a polytope. It can be shown that

$$\Phi(\theta) \;=\; \sup_{\mu\in\mathrm{MARG}(G,\phi)} \langle\theta,\mu\rangle - \Phi^*(\mu) \tag{2.6}$$

$$\;=\; \sup_{\mu\in\mathcal{M}(\phi)} \langle\theta,\mu\rangle - \Phi^*(\mu) \tag{2.7}$$

where $\mathcal{M}(\phi) = \{\mu \in \mathbb{R}^n \mid \sum_x \phi(x)p(x) = \mu \text{ for some } p\}$. Variational approximations then have the following programme: approximate the polytope to one with a compact description, and approximate the dual entropy function. We refer to [64] for a comprehensive introduction to these constructions and their relevance to variational approximations.

Chapter 3

Preconditioner Approximations

In the previous chapter, we described the "projection optimization" paradigm; which reduces the log-partition function computation to an optimization problem: of estimating the "closest" simpler model from a candidate set of simple models. The implementation of this programme however was largely restricted to approximating the KL divergence measure; or the conjugate dual (entropy) of the log-partition function. In what follows, we leverage recent scientific computing developments to propose a low time-complexity, non-variational class of approximation techniques that implements the "projection optimization" paradigm differently. We begin with some background on preconditioners and linear systems, give a high level idea of the technique, and then move on to formal method and experiments.

3.1 Preconditioners in Linear Systems

Consider a linear system, $Ax = c$, where the variable x is n dimensional, and A is an $n \times n$ matrix with m non-zero entries. Solving for x via direct methods such as Gaussian elimination has a computational complexity $O(n^3)$, which is impractical for large values of n. Multiplying both sides of the linear system by the inverse of an invertible matrix B, we get an equivalent "preconditioned" system, $B^{-1}Ax = B^{-1}c$. If B is similar to A, $B^{-1}A$ is in turn similar to I, the identity matrix, making the preconditioned system easier to solve. Such an approximating matrix B is called a preconditioner.

The computational complexity of preconditioned conjugate gradient is given by

$$T(A) = \sqrt{\kappa(A, B)}\,(m + T(B)) \log\left(\frac{1}{\epsilon}\right)$$

where $T(A)$ is the time required for an ϵ-approximate solution; $\kappa(A, B)$ is the *condition number* of A and B which intuitively corresponds to the quality of the approximation B, and $T(B)$ is the time required to solve $By = c$.

Recent developments in the theory of preconditioners are in part based on *support graph theory*, where the linear system matrix is viewed as the Laplacian of a graph, and graph-based techniques can be used to obtain good approximations. While these methods require diagonally dominant matrices ($A_{ii} \geq \sum_{j \neq i} |A_{ij}|$), they yield "ultra-sparse" (tree plus a constant number of edges) preconditioners with a low condition number. In our experiments, we use two elementary tree-based preconditioners in this family, Vaidya's Spanning Tree preconditioner [54], and Gremban-Miller's Support Tree preconditioner [23].
One example is Vaidya's preconditioner [54], which is essentially the maximum spanning tree of the graph. Another is the support tree of [23], which introduces Steiner nodes, in this case auxiliary nodes introduced via a recursive partitioning of the graph. We present experiments with these basic preconditioners in the following section.

3.2 Graphical Model Preconditioners

We shall assume a pairwise random field, and thus can express the potential function and parameter vectors in more compact form as matrices:

$$\Theta := \begin{pmatrix} \theta_{11} & \dots & \theta_{1n} \\ \vdots & \vdots & \vdots \\ \theta_{n1} & \dots & \theta_{nn} \end{pmatrix} \Phi(x) := \begin{pmatrix} \phi_{11}(x_1, x_1) & \dots & \phi_{1n}(x_1, x_n) \\ \vdots & \vdots & \vdots \\ \phi_{n1}(x_n, x_1) & \dots & \phi_{nn}(x_n, x_n) \end{pmatrix} \tag{3.1}$$

In the following we will denote the trace of the product of two matrices A and B by the inner product $\langle\langle A, B\rangle\rangle$. Assuming that each X_i is finite-valued, the partition function $Z(\Theta)$ is then given by $Z(\Theta) = \sum_{x \in \chi} \exp \langle\langle \Theta, \Phi(x)\rangle\rangle$. The computation of $Z(\Theta)$ has a complexity exponential in the tree-width of the graph G and hence is intractable for large graphs. Our goal is to obtain rigorous upper and lower bounds for this partition function, which can then be used to obtain rigorous upper and lower bounds for general event probabilities; this is discussed further in [45].

3.2.1 Main Idea

Consider the graphical model with graph G, potential-function matrix $\Phi(x)$, and parameter matrix Θ. For purposes of intuition, think of the graphical model "energy" $\langle\langle \Theta, \Phi(x)\rangle\rangle$ as the matrix norm $x^\top \Theta x$. We would like to obtain a sparse approximation B for Θ. If B approximates Θ well, then the condition number κ is small:

$$\kappa(\Theta, B) \;=\; \max_x \frac{x^\top \Theta x}{x^\top B x} \Bigg/ \min_x \frac{x^\top \Theta x}{x^\top B x} \;=\; \lambda_{max}(\Theta, B) \,/ \lambda_{min}(\Theta, B) \quad (3.2)$$

This suggests the following procedure for approximate inference. First, choose a matrix B that minimizes the condition number with Θ (rather than KL divergence as in mean-field). Then, scale B appropriately, as detailed in the following sections. Finally, use the scaled matrix B as the parameter matrix for approximate inference. Note that if B corresponds to a tree, approximate inference has linear time complexity.

3.2.2 Generalized Eigenvalue Bounds

Given a graphical model with graph G, potential-function matrix $\Phi(x)$, and parameter matrix Θ, our goal is to obtain parameter matrices Θ_U and Θ_L, corresponding to sparse graph approximations of G, such that

$$Z(\Theta_L) \;\leq\; Z(\Theta) \;\leq\; Z(\Theta_U). \quad (3.3)$$

That is, the partition functions of the sparse graph parameter matrices Θ_U and Θ_L are upper and lower bounds, respectively, of the partition function of the original graph. However, we will instead focus on a seemingly much *stronger* condition; in particular, we will look for Θ_L and Θ_U that satisfy

$$\langle\langle \Theta_L, \Phi(x) \rangle\rangle \;\leq \langle\langle \Theta, \Phi(x) \rangle\rangle \;\leq\; \langle\langle \Theta_U, \Phi(x) \rangle\rangle \quad (3.4)$$

for all x. By monotonicity of $\exp$, this stronger condition implies condition (3.3) on the partition function, by summing over the values of X. However, this stronger condition will give us greater flexibility, and rigorous bounds for general event probabilities since then

$$\frac{\exp \langle\langle \Theta_L, \Phi(x) \rangle\rangle}{Z(\Theta_U)} \;\leq\; p(x; \Theta) \;\leq\; \frac{\exp \langle\langle \Theta_U, \Phi(x) \rangle\rangle}{Z(\Theta_L)}. \quad (3.5)$$

In contrast, while variational methods give bounds on the log partition function, the derived bounds on general event probabilities via the variational parameters are only heuristic.

Let $\mathcal{S}$ be a set of sparse graphs; for example, $\mathcal{S}$ may be the set of all trees. Focusing on the upper bound, we for now would like to obtain a graph $G' \in \mathcal{S}$ with parameter matrix B, which approximates G, and whose partition function upper bounds the partition function of the original graph. Following (3.4), we require,

$$\langle\langle \Theta, \Phi(x) \rangle\rangle \;\leq\; \langle\langle B, \Phi(x) \rangle\rangle \,, \text{ such that } G(B) \in \mathcal{S} \tag{3.6}$$

where $G(B)$ denotes the graph corresponding to the parameter matrix B. Now, we would like the distribution corresponding to B to be as close as possible to the distribution corresponding to Θ; that is, $\langle\langle B, \Phi(x) \rangle\rangle$ should not only upper bound $\langle\langle \Theta, \Phi(x) \rangle\rangle$ but should be close to it. The distance measure we use for this is the minimax distance. In other words, while the upper bound requires that

$$\frac{\langle\langle \Theta, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle} \;\leq\; 1, \tag{3.7}$$

we would like

$$\min_x \frac{\langle\langle \Theta, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle} \tag{3.8}$$

to be as high as possible. Expressing these desiderata in the form of an optimization problem, we have

$$B^\star \;=\; \operatorname*{argmax}_{B:\, G(B)\in\mathcal{S}} \; \min_x \; \tfrac{\langle\langle \Theta, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle}, \;\; \text{such that} \;\; \tfrac{\langle\langle \Theta, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle} \leq 1.$$

Before solving this problem, we first make some definitions, which are generalized versions of standard concepts in linear systems theory.

Definition 1. *For a pairwise Markov random field with potential function matrix $\Phi(x)$; the generalized eigenvalues of a pair of parameter matrices (A, B) are defined as*

$$\lambda^{\Phi}_{max}(A,B) \;=\; \max_{x:\, \langle\langle B, \Phi(x) \rangle\rangle \neq 0} \frac{\langle\langle A, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle} \tag{3.9}$$

$$\lambda^{\Phi}_{min}(A,B) \;=\; \min_{x:\, \langle\langle B, \Phi(x) \rangle\rangle \neq 0} \frac{\langle\langle A, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle}. \tag{3.10}$$

Note that

$$\lambda^{\Phi}_{\max}(A,\alpha B) \;=\; \max_{x:\, \langle\langle \alpha B, \Phi(x) \rangle\rangle \neq 0} \frac{\langle\langle A, \Phi(x) \rangle\rangle}{\langle\langle \alpha B, \Phi(x) \rangle\rangle} \tag{3.11}$$

$$= \frac{1}{\alpha} \max_{x:\, \langle\langle B, \Phi(x) \rangle\rangle \neq 0} \frac{\langle\langle A, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle} \;=\; \alpha^{-1} \lambda^{\Phi}_{\max}(A,B). \tag{3.12}$$

We state the basic properties of the generalized eigenvalues in the following lemma.

Lemma 1. *The generalized eigenvalues satisfy*

$$\lambda^{\Phi}_{min}(A,B) \le \frac{\langle\langle A,\Phi(x)\rangle\rangle}{\langle\langle B,\Phi(x)\rangle\rangle} \le \lambda^{\Phi}_{max}(A,B) \tag{3.13}$$

$$\lambda^{\Phi}_{max}(A,\alpha B) = \alpha^{-1}\lambda^{\Phi}_{max}(A,B) \tag{3.14}$$

$$\lambda^{\Phi}_{min}(A,\alpha B) = \alpha^{-1}\lambda^{\Phi}_{min}(A,B) \tag{3.15}$$

$$\lambda^{\Phi}_{min}(A,B) = \frac{1}{\lambda^{\Phi}_{max}(B,A)}\,. \tag{3.16}$$

In the following, we will use A to generically denote the parameter matrix Θ of the model. We can now rewrite the optimization problem for the upper bound in equation (3.9) as

$$(\text{Problem } \Lambda_1) \qquad \max_{B:\ G(B)\in\mathcal{S}} \lambda^{\Phi}_{\text{min}}(A,B), \ \text{ such that } \ \lambda^{\Phi}_{\text{max}}(A,B) \le 1 \tag{3.17}$$

We shall express the optimal solution of Problem Λ_1 in terms of the optimal solution of a companion problem. Towards that end, consider the optimization problem

$$(\text{Problem } \Lambda_2) \qquad \min_{C:\ G(C)\in\mathcal{S}} \frac{\lambda^{\Phi}_{\text{max}}(A,C)}{\lambda^{\Phi}_{\text{min}}(A,C)}. \tag{3.18}$$

The following proposition shows the sense in which these problems are equivalent.

Proposition 1. *If $\widehat{C}$ attains the optimum in Problem Λ_2, then $\widetilde{C} = \lambda^{\Phi}_{max}(A,\widehat{C})\,\widehat{C}$ attains the optimum of Problem Λ_1.*

Proof. For any feasible solution B of Problem Λ_1, we have

$$\begin{aligned}
\lambda^{\Phi}_{\text{min}}(A,B) &\le \frac{\lambda^{\Phi}_{\text{min}}(A,B)}{\lambda^{\Phi}_{\text{max}}(A,B)} \quad (\text{since } \lambda^{\Phi}_{\text{max}}(A,B) \le 1) && (3.19)\\
&\le \frac{\lambda^{\Phi}_{\text{min}}(A,\widehat{C})}{\lambda^{\Phi}_{\text{max}}(A,\widehat{C})} \quad (\text{since } \widehat{C} \text{ is the optimum of Problem } \Lambda_2) && (3.20)\\
&= \lambda^{\Phi}_{\text{min}}\left(A, \lambda^{\Phi}_{\text{max}}(A,\widehat{C})\widehat{C}\right) \quad (\text{from Lemma 1}) && (3.21)\\
&= \lambda^{\Phi}_{\text{min}}(A,\widetilde{C}). && (3.22)
\end{aligned}$$

Thus, $\widetilde{C}$ upper bounds all feasible solutions in Problem Λ_1. However, it itself is a feasible solution, since

$$\lambda^{\Phi}_{\text{max}}(A,\widetilde{C}) = \lambda^{\Phi}_{\text{max}}\left(A, \lambda^{\Phi}_{\text{max}}(A,\widehat{C})\widehat{C}\right) = \frac{1}{\lambda^{\Phi}_{\text{max}}(A,\widehat{C})}\lambda^{\Phi}_{\text{max}}(A,\widehat{C}) = 1 \tag{3.23}$$

from Lemma 1. Thus, $\widetilde{C}$ attains the maximum in the upper bound Problem Λ_1. □

The analysis for obtaining an upper bound parameter matrix B for a given parameter matrix A carries over for the lower bound; we need to replace a maximin problem with a minimax problem. For the lower bound, we want a matrix B such that

$$B_\star \;=\; \min_{B\colon G(B)\in\mathcal{S}} \;\max_{\{x\colon \langle\langle B,\Phi(x)\rangle\rangle\neq 0\}} \frac{\langle\langle A,\Phi(x)\rangle\rangle}{\langle\langle B,\Phi(x)\rangle\rangle}, \quad \text{such that} \;\; \frac{\langle\langle A,\Phi(x)\rangle\rangle}{\langle\langle B,\Phi(x)\rangle\rangle} \geq 1 \tag{3.24}$$

This leads to the following lower bound optimization problem.

$$\text{(Problem } \Lambda_3\text{)} \qquad \min_{B\colon G(B)\in\mathcal{S}} \lambda^{\Phi}_{\max}(A,B), \quad \text{such that} \;\; \lambda^{\Phi}_{\min}(A,B) \geq 1. \tag{3.25}$$

The proof of the following statement closely parallels the proof of Proposition 1.

Proposition 2. *If $\hat{C}$ attains the optimum in Problem Λ_2, then $\underline{C} = \lambda^{\Phi}_{\mathit{min}}(A,\hat{C})\hat{C}$ attains the optimum of the lower bound Problem Λ_3.*

Finally, we state the following basic lemma, whose proof is easily verified.

Lemma 2. *For any pair of parameter-matrices (A,B), we have*

$$\left\langle\left\langle \lambda^{\Phi}_{\mathit{min}}(A,B)B,\Phi(x)\right\rangle\right\rangle \;\leq\; \langle\langle A,\Phi(x)\rangle\rangle \;\leq\; \left\langle\left\langle \lambda^{\Phi}_{\mathit{max}}(A,B)B,\Phi(x)\right\rangle\right\rangle. \tag{3.26}$$

3.2.3 Main Procedure

We now have in place the machinery necessary to describe the procedure for solving the main problem in equation (3.4), to obtain upper and lower bound matrices for a graphical model. Lemma 2 shows how to obtain upper and lower bound parameter matrices with respect to any matrix B, given a parameter matrix A, by solving a generalized eigenvalue problem. Propositions 1 and 2 tell us, in principle, how to obtain the optimal such upper and lower bound matrices. We thus have the following procedure. First, Obtain a parameter matrix C such that $G(C)\in\mathcal{S}$, which minimizes $\lambda^{\Phi}_{\max}(\Theta,C)/\lambda^{\Phi}_{\min}(\Theta,C)$. Then $\lambda^{\Phi}_{\max}(\Theta,C)\,C$ gives the optimal upper bound parameter matrix and $\lambda^{\Phi}_{\min}(\Theta,C)\,C$ gives the optimal lower bound parameter matrix. However, as things stand, this recipe appears to be even more challenging to work with than the generalized mean field procedures. The difficulty lies in obtaining the matrix C. In the following section we offer a series of relaxations that help to simplify this task.

3.3 Generalized Support Theory for Graphical Models

In what follows, we begin by assuming that the potential function matrix is positive semi-definite, $\Phi(x) \succeq 0$, and later extend our results to general Φ.

Definition 2. *For a pairwise MRF with potential function matrix $\Phi(x) \succeq 0$, the generalized support number of a pair of parameter matrices (A, B), where $B \succeq 0$, is*

$$\sigma^{\Phi}(A,B) = \min \{ \tau \in \mathbf{R} \,|\, \langle\langle \tau B, \Phi(x) \rangle\rangle \geq \langle\langle A, \Phi(x) \rangle\rangle \text{ for all } x \} \tag{3.27}$$

The generalized support number can be thought of as the "number of copies" τ of B required to "support" A so that $\langle\langle \tau B - A, \Phi(x) \rangle\rangle \geq 0$. The usefulness of this definition is demonstrated by the following result.

Proposition 3. *If $B \succeq 0$ then $\lambda^{\Phi}_{max}(A,B) \leq \sigma^{\Phi}(A,B)$.*

Proof. From the definition of the generalized support number for a graphical model, we have that $\left\langle\left\langle \sigma^{\Phi}(A,B)B - A, \Phi(x) \right\rangle\right\rangle \geq 0$. Now, since we assume that $\Phi(x) \succeq 0$, if also $B \succeq 0$ then $\langle\langle B, \Phi(x) \rangle\rangle \geq 0$. Therefore, it follows that $\frac{\langle\langle A, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle} \leq \sigma^{\Phi}(A,B)$, and thus

$$\lambda^{\Phi}_{\text{max}}(A,B) = \max_x \frac{\langle\langle A, \Phi(x) \rangle\rangle}{\langle\langle B, \Phi(x) \rangle\rangle} \leq \sigma^{\Phi}(A,B) \tag{3.28}$$

giving the statement of the proposition. □

This leads to our first relaxation of the generalized eigenvalue bound for a model. From Lemma 1 and Proposition 3 we see that

$$\frac{\lambda^{\Phi}_{\text{max}}(A,B)}{\lambda^{\Phi}_{\text{min}}(A,B)} \;=\; \lambda^{\Phi}_{\text{max}}(A,B)\lambda^{\Phi}_{\text{max}}(B,A) \;\leq\; \sigma^{\Phi}(A,B)\sigma^{\Phi}(B,A) \tag{3.29}$$

Thus, this result suggests that to approximate the graphical model (Θ, Φ) we can search for a parameter matrix $B^{\star}$, with corresponding simple graph $G(B^{\star}) \in \mathcal{S}$, such that

$$B^{\star} = \operatorname*{argmin}_B \; \sigma^{\Phi}(\Theta, B)\sigma^{\Phi}(B, \Theta) \tag{3.30}$$

While this relaxation may lead to effective bounds, we will now go further, to derive an additional relaxation that relates our generalized graphical model support number to the "classical" support number.

Proposition 4. *For a potential function matrix $\Phi(x) \succeq 0$, $\sigma^{\Phi}(A,B) \leq \sigma(A,B)$, where $\sigma(A,B) = \min\{\tau \,|\, (\tau B - A) \succeq 0\}$.*

Proof. Since $\sigma(A,B)B - A \succeq 0$ by definition and $\Phi(x) \succeq 0$ by assumption, we have that $\langle\langle \sigma(A,B)B - A, \Phi(x)\rangle\rangle \geq 0$. Therefore, $\sigma^{\Phi}(A,B) \leq \sigma(A,B)$ from the definition of generalized support number. □

The above result reduces the problem of approximating a graphical model to the problem of minimizing classical support numbers, the latter problem being well-studied in the scientific computing literature [6, 3], where the expression $\sigma(A,C)\sigma(C,A)$ is called the *condition number*, and a matrix that minimizes it within a simple family of graphs is called a *preconditioner*. We can thus plug in any algorithm for finding a sparse preconditioner for Θ, carrying out the optimization

$$B^{\star} = \underset{B}{\operatorname{argmin}}\, \sigma(\Theta, B)\, \sigma(B, \Theta) \tag{3.31}$$

and then use that matrix $B^{\star}$ in our basic procedure.

Before turning to the experiments, we comment that our generalized support number analysis assumed that the potential function matrix $\Phi(x)$ was positive semi-definite. The case when it is not can be handled as follows. We first add a large positive diagonal matrix D so that $\Phi'(x) = \Phi(x) + D \succeq 0$. Then, for a given parameter matrix Θ, we use the above machinery to get an upper bound parameter matrix B such that

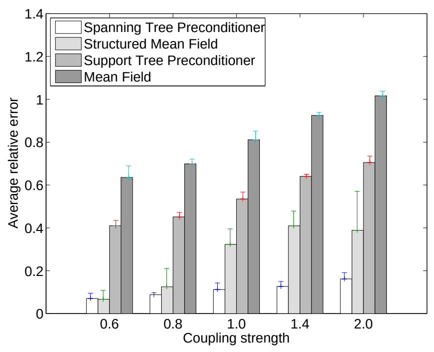
$$\langle\langle A, \Phi(x) + D\rangle\rangle \leq \langle\langle B, \Phi(x) + D\rangle\rangle \;\Rightarrow\; \langle\langle A, \Phi(x)\rangle\rangle \leq \langle\langle B, \Phi(x)\rangle\rangle + \langle\langle B - A, D\rangle\rangle\,. \tag{3.32}$$

Exponentiating and summing both sides over x, we then get the required upper bound for the parameter matrix A; the same can be done for the lower bound.

3.4 Experiments

As the previous sections detailed, the preconditioner based bounds are in principle quite easy to compute—we compute a sparse preconditioner for the parameter matrix (typically $O(n)$ to $O(n^3)$) and use the preconditioner as the parameter matrix for the bound computation (which is linear if the preconditioner matrix corresponds to a tree). This yields a simple, non-iterative deterministic procedure as compared to the more complex propagation-based or iterative update procedures. In this section we evaluate these bounds on small graphical models for which exact answers can be readily computed, and compare the bounds to variational approximations.

We show simulation results averaged over a randomly generated set of graphical models. The graphs used were 2D grid graphs, and the edge potentials were



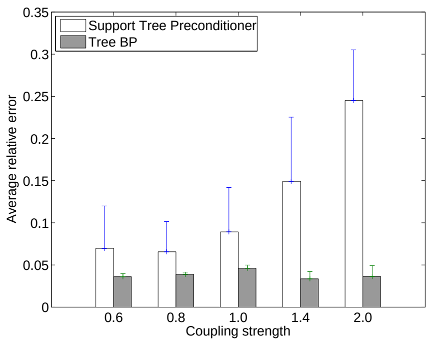


Figure 3.1: Comparison of lower bounds (left), and upper bounds (right) for small grid graphs

selected according to a uniform distribution Uniform$(-2d_{coup}, 0)$ for various coupling strengths d_{coup}. We report the relative error,

$$\text{rel. error} = (\text{bound} - \text{log-partition-function})/\text{log-partition-function}$$

As a baseline, we use the mean field and structured mean field methods for the lower bound, and the [59] tree-reweighted belief propagation approximation for the upper bound. For the preconditioner based bounds, we use two very simple preconditioners, (a) Vaidya's maximum spanning tree preconditioner [54], which assumes the input parameter matrix to be a Laplacian, and (b) [23]'s support tree preconditioner, which also gives a sparse parameter matrix corresponding to a tree, with Steiner (auxiliary) nodes. To compute bounds over these larger graphs with Steiner nodes we average an internal node over its children; this is the technique used with such preconditioners for solving linear systems. We note that these preconditioners are quite basic, and the use of better preconditioners (yielding a better condition number) has the potential to achieve much better bounds, as shown in Propositions 1 and 2. We also reiterate that while our approach can be used to derive bounds on event probabilities, the variational methods yield bounds only for the partition function, and only apply heuristically to estimating simple event probabilities such as marginals.

As the plots in Figure (3.1) show, even for the simple preconditioners used, the new bounds are quite close to the actual values, outperforming the mean field method and giving comparable results to the tree-reweighted belief propagation method. The spanning tree preconditioner provides a good lower bound, while the support tree preconditioner provides a good upper bound, however not as tight as the bound obtained using tree-reweighted belief propagation. Although we cannot compute the exact solution for large graphs, we can compare bounds. Figure (3.2)

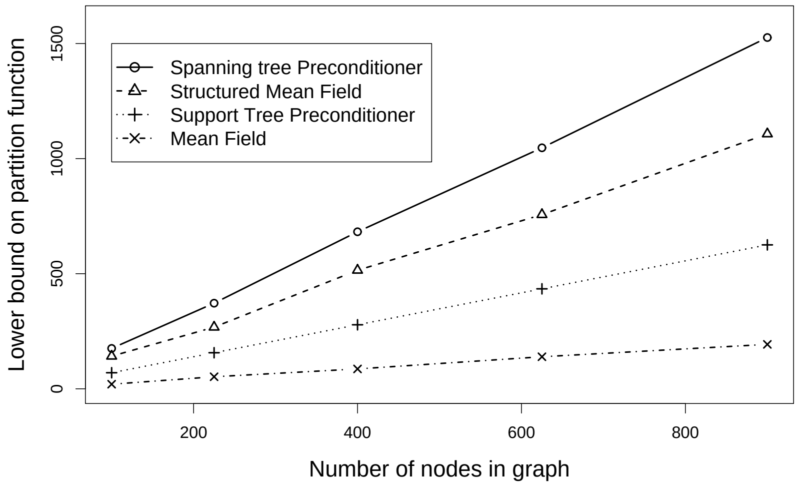


Figure 3.2: Lower bounds for grid graphs of increasing size

compares lower bounds for graphs with up to 900 nodes; a larger bound is necessarily tighter, and the preconditioner bounds are seen to outperform mean field and structured mean field.

Chapter 4

Quadratic Programming Relaxations for MAP

4.1 MAP Estimation

In this chapter, we consider the inference problem of computing the maximum a posteriori (MAP) configuration – the most probable assignment of values to the nodes – for undirected graphical models.

For tree-structured distributions, the MAP estimate for random fields can be computed efficiently by dynamic programming. It can also be computed in polynomial time using graph cuts [22] when the parameter settings yield a submodular energy function. In the general setting, a widely used approximation technique is max-product belief propagation [42]. The algorithm is convergent on trees, and its fixed point configuration upon convergence can be shown to be locally optimal with respect to a large set of moves [66]. A similar message passing algorithm, tree-reweighted max product [57], has stronger correctness and convergence guarantees. [8] have proposed graph-cut based algorithms that efficiently find a local energy minimum with respect to two types of large moves. A different direction has been taken in recent work on linear program relaxations for the MAP problem in the specific setting of metric labeling. In the metric labeling formulation, the goal is to find a minimum cost labeling of a set of objects, where the energy or cost of different labelings is the sum of node and edge costs specified by by a weighted graph and a metric over the labels. Casting this as an integer linear program, [28] proposed linear relaxations for specific metrics. [10] recently extended these techniques using the natural linear relaxation of the metric labeling task, and obtained stronger approximation guarantees.

We propose a quadratic programming (QP) relaxation to the MAP or metric labeling problem. While the linear relaxations have $O(|E|K^2)$ variables, where $|E|$ is the number of edges in the graph and K is the number of labels; in our QP formulation there are nK variables, and yet we show that the quadratic objective function more accurately represents the energy in the graphical model. In particular, we show that the QP formulation computes the MAP solution exactly. Under certain conditions however, the relaxation results in a non-convex problem, which requires an intractable search over local minima. This motivates an additional convex approximation to the relaxation, which we show satisfies an additive approximation guarantee. We also extend the relaxation to general variational "inner polytope" relaxations which we also show to compute the MAP exactly. Experiments indicate that our quadratic relaxation with the convex approximation outperforms or is comparable to existing methods under most settings.

We were made aware recently of an unpublished manuscript by T. Wierschin and D. Fuchs, where they investigate a quadratic approach to the labeling problem, and where they show that the quadratic program relaxation of the quadratic integer program formulation of the labeling problem, is tight; just as we do in Section 4.4 for the general MRF MAP problem. Also, [51] and [31] investigate semidefinite programming (SDP) and second order cone programming (SOCP) approaches to the MAP problem, which fall between the QP and the LP in restricting the constraint set. While we shall formulate the LP and QP in detail in the next section, we now describe the differences between these approaches at a high level. The LP is of the form,

$$\min_{x,X} \quad a^\top x + \mathrm{Tr}(BX) \tag{4.1}$$

$$\text{s.t.} \quad \sum_i x_i = 1 \tag{4.2}$$

$$0 \le x_i \le 1 \tag{4.3}$$

where x is $nK \times 1$ and X is $nK \times nK$. The QP dispenses away with the huge X parameter matrix as follows,

$$\min_{x,X} \quad a^\top x + x^\top Bx \tag{4.4}$$

$$\text{s.t.} \quad \sum_i x_i = 1 \tag{4.5}$$

$$0 \le x_i \le 1 \tag{4.6}$$

As to why this makes sense will be clear from the next section, but note that this can be rewritten as a quadratic constraint,

$$\begin{aligned} \min_{x,X} \quad & a^\top x + \mathrm{Tr}(BX) && (4.7)\\ \text{s.t.} \quad & \sum_i x_i = 1 && (4.8)\\ & 0 \le x_i \le 1 && (4.9)\\ & X = xx^\top && (4.10) \end{aligned}$$

Note that the LP throws away the quadratic constraint; merely relaxing it gives the SDP below,

$$\begin{aligned} \min_{x,X} \quad & a^\top x + \mathrm{Tr}(BX) && (4.11)\\ \text{s.t.} \quad & \sum_i x_i = 1 && (4.12)\\ & 0 \le x_i \le 1 && (4.13)\\ & X \succeq xx^\top && (4.14) \end{aligned}$$

The SOCP lies between even the SDP and the LP in restricting the constraint set. The key observation is that $M \succeq 0$ is equivalent to a large set of constraints: Tr$(SM) \ge 0$ for all $S \succeq 0$. The SOCP explicitly lists a few such linear constraints with matrices $S \in \mathcal{S}$ instead of the semi-definite constraint,

$$\begin{aligned} \min_{x,X} \quad & a^\top x + \mathrm{Tr}(BX) && (4.15)\\ \text{s.t.} \quad & \sum_i x_i = 1 && (4.16)\\ & 0 \le x_i \le 1 && (4.17)\\ & \mathrm{Tr}(SX) \ge 0,\ S \in \mathcal{S} && (4.18) \end{aligned}$$

4.2 Problem Formulation

Consider the pairwise graphical model with potentials $\phi(x)$ and parameters θ,

$$p(x;\theta) \propto \exp\left(\sum_s \theta_s \phi_s(x_s) + \sum_{(s,t)\in E} \theta_{st}\phi_{st}(x_s,x_t)\right).$$

If each X_s takes values in a discrete set $\mathcal{X}_s$, we can represent any potential function as a linear combination of indicator functions, $\phi_s(x_s) = \sum_j \phi_s(j)\,\mathcal{I}_j(x_s)$ and

$\phi_{st}(x_s, x_t) = \sum_{jk} \phi_{st}(j,k)\, \mathcal{I}_{j,k}(x_s, x_t)$ where

$$\mathcal{I}_j(x_s) \quad = \quad \begin{cases} 1 & x_s = j \\ 0 & \text{otherwise} \,. \end{cases}$$

and

$$\mathcal{I}_{j,k}(x_s, x_t) \quad = \quad \begin{cases} 1 & x_s = j \text{ and } x_t = k \\ 0 & \text{otherwise} \end{cases}$$

We can thus, without loss of generality, consider pairwise MRFs with indicator potential functions as

$$p(x|\theta) \propto \exp\left(\sum_{s,j} \theta_{s;j} \mathcal{I}_j(x_s) + \sum_{s,t;j,k} \theta_{s,j;t,k} \mathcal{I}_{j,k}(x_s, x_t) \right) .$$

The MAP problem is then given by

$$x^* = \operatorname*{argmax}_x \sum_{s,j} \theta_{s;j} \mathcal{I}_j(x_s) + \sum_{s,t;j,k} \theta_{s,j;t,k} \mathcal{I}_{j,k}(x_s, x_t). \qquad (4.19)$$

4.3 Linear Relaxations

MAP estimation in the discrete case is essentially a combinatorial optimization problem, and it can be cast as an integer program. Recent work has studied approximate MAP estimation using linear relaxations [4]. Letting variables $\mu(s;j)$ and $\mu(s,j;t,k)$ correspond to the indicator variables $\mathcal{I}_j(x_s)$ and $\mathcal{I}_{j,k}(x_s, x_t)$, we obtain the following integer linear program (ILP),

$$\begin{aligned} \max \quad & \sum_{s;j} \theta_{s;j}\, \mu_1(s;j) + \sum_{s,t;j,k} \theta_{s,j;t,k}\, \mu_2(s,j;t,k) \\ \text{such that} \quad & \sum_k \mu_2(s,j;t,k) = \mu_1(s;j) \\ & \sum_j \mu_1(s;j) = 1 \\ & \mu_1(s;j) \in \{0,1\} \\ & \mu_2(s,j;t,k) \in \{0,1\}. \end{aligned}$$

This ILP can then be relaxed to the following linear program (LP),

$$\begin{aligned}
\max \quad & \sum_{s;j} \theta_{s;j}\, \mu_1(s;j) + \sum_{s,t;j,k} \theta_{s,j;t,k}\, \mu_2(s,j;t,k) \\
\text{such that} \quad & \sum_{k} \mu_2(s,j;t,k) = \mu_1(s;j) \\
& \sum_{j} \mu_1(s;j) = 1 \\
& 0 \le \mu_1(s;j) \le 1 \\
& 0 \le \mu_2(s,j;t,k) \le 1.
\end{aligned} \tag{4.20}$$

[10] propose the above LP relaxation as an approximation algorithm for the metric labeling task, which is the MAP problem with spatially homogeneous MRF parameters; thus, $\theta_{s,j;t,k} = w_{st}\, d(j,k)$, where w_{st} is a non-negative edge weight and d is a metric that is the same for all the edges. [28] proposed related linear relaxations for specific metrics. The above LP relaxation was also proposed for the general pairwise graphical model setting by [61]. Letting θ and $\phi(x)$ denote the vectors of parameters and potential functions, respectively, and letting $\langle \theta, \phi(x) \rangle$ denote the inner product

$$\langle \theta, \phi(x) \rangle = \sum_{s;j} \theta_{s;j} \mathcal{I}_j(x_s) + \sum_{(s,t)\in E;\, j,k} \theta_{s,j;t,k} \mathcal{I}_{j,k}(x_s, x_t)$$

the MAP problem is then given by

$$x^* = \operatorname*{argmax}_{x} \langle \theta, \phi(x) \rangle = \sup_{\mu \in \mathcal{M}} \langle \theta, \mu \rangle$$

where $\mathcal{M}$ is the set of moment parameters

$$\mathcal{M} = \left\{ \mu \,:\, \sum_{x} p(x)\phi(x) = \mu \text{ for some distribution } p \right\}.$$

The polytope $\mathcal{M}$ can be seen to be upper-bounded by the set LOCAL(G) of all single and pairwise vectors μ_1 and μ_2 that satisfy the local consistency constraints

$$\begin{aligned}
\textstyle\sum_k \mu_2(s,j;t,k) &= \mu_1(s;j) \\
\textstyle\sum_j \mu_1(s,j) &= 1 \\
0 \le \mu_1(s;j) &\le 1 \\
0 \le \mu_2(s,j;t,k) &\le 1.
\end{aligned}$$

[61] thus proposed the upper-bounding relaxation of using $LOCAL(G)$ as an outer bound for the polytope $\mathcal{M}$,

$$\mu^* = \sup_{\mu \in LOCAL(G)} \langle \theta, \mu \rangle \,, \tag{4.21}$$

which is the same LP formulation as in equation (4.20). Furthermore, [57] show that under certain conditions, the tree-reweighted belief propagation updates solve the dual of the LP in equation (4.21); since strong duality holds, the tree updates also give the optimal primal value for the LP.

4.4 Quadratic Relaxation

In the linear relaxation of equation (4.20), the variables $\mu_2(s,j;t,k)$ are relaxations of the indicator variables $\mathcal{I}_{j,k}(x_s,x_t)$, with a value of one indicating that for edge $(s,t) \in E$, variable X_s is labeled j and variable X_t is labeled k. These pairwise variables are constrained by demanding that they be consistent with the corresponding "marginal" variables $\mu_1(s,j)$. Note, however, that the binary indicator variables satisfy the additional "independence" constraint

$$\mathcal{I}_{j,k}(x_s,x_t) = \mathcal{I}_j(x_s)\,\mathcal{I}_k(x_t). \tag{4.22}$$

This then suggests that constraining the relaxation variables in a similar manner, $\mu_2(s,j;t,k) = \mu_1(s;j)\,\mu(t;k)$, might yield a tighter relaxation. This leads to the following quadratic program

$$\begin{aligned} \text{max} \quad & \sum_{s;j} \theta_{s;j}\mu(s;j) + \sum_{s,t;j,k} \theta_{s,j;t,k}\,\mu(s;j)\,\mu(t;k) \\ \text{subject to} \quad & \sum_{j} \mu(s;j) = 1 \\ & 0 \le \mu(s;j) \le 1 \end{aligned} \tag{4.23}$$

The following result shows that the relaxation is in fact tight; our proof uses the probabilistic method.

Theorem 2. *The optimal value of problem* (4.23) *is equal to the optimal value of the MAP problem* (4.19).

Note that the theorem just states the existence of a discrete solution with the same energy as that of the optimal real relaxation. The problem of efficiently obtaining such a discrete solution from the relaxed solution is considered next.

Theorem 3. *Any solution of the MAP problem* (4.19) *efficiently yields a solution of the relaxation* (4.23) *and vice versa. Thus the relaxation* (4.23) *is equivalent to the MAP problem* (4.19).

Proof. From theorem 2, the optimal values of problems (4.19) and (4.23) are equal; let e^* denote this maximum energy. Let $\hat{x}$ be an optimal solution of the MAP problem (4.19). As problem (4.23) is a relaxation of the MAP problem, $\mu(s;j) = \mathcal{I}(\hat{x};j)$ is also a feasible and optimal solution for (4.23).

For the converse, let μ^* be an optimal solution of problem (4.23). Its energy is given by

$$e^* = \sum_{s;j} \theta_{s;j}\mu^*(s;j) + \sum_{(s,t)\in E;j,k} \theta_{s,j;t,k}\mu^*(s;j)\mu^*(t;k) \tag{4.24}$$

If each $\mu^*(s;j)$ is integer valued, that is, in $\{0,1\}$, then we can use μ^* itself as the feasible optimal solution for the MAP problem (4.19). Otherwise, consider μ^* to be real valued; we (efficiently) construct a labeling y with the maximum energy e^*.

Consider an unlabeled node s. Assign it label

$$y_s = \operatorname*{argmax}_j \theta_{s;j} + \sum_{t:(s,t)\in E;k} \theta_{s,j;t,k}\mu^*(t;k)$$

Now, set $\mu^*(s;y_s) = 1$ and $\mu^*(s;k) = 0\ ;\ k \neq y_s$. Continue with this labeling process until all nodes are labeled. It can be shown that the energy of this assignment y is equal to the energy e^* of the optimal MAP assignment. In particular, each time we take up an unlabeled node t, we select a labeling that does not decrease the expected energy of the unlabeled nodes given the labelings of the labeled nodes. Given that the initial expected energy of all unlabeled nodes was e, the energy at the end of the process, that is, of the assignment y, is thus at least e^*. ☐

4.5 Convex Approximation

The previous section showed that the relaxation in equation (4.23), while a simple extension of the LP in equation (4.20), is actually equivalent to the MAP problem. This yields the interesting result that the MAP problem is solvable in polynomial time if the edge parameter matrix $\Theta = [\theta_{s,j;t,k}]$ is negative definite, since in this case the QP (4.23) is a convex program. Note also that the quadratic program has a simple set of constraints (only linear and box constraints), which are also small in number, and is thus a simple problem instance of general convex optimization. It should also be stressed that for an n node graph, the QP has only kn variables while the LP has $O(k^2|E|)$ variables.

The case where the edge parameter matrix Θ is not negative definite yields a non-convex program; and while we could do an iterative search procedure upto a local maximum as max-product does, we now describe a convex approximation which provides a polynomial time solution with additive bound guarantees.

Consider the quadratic integer program (QIP) corresponding to the QP, given by

$$\begin{aligned} \max_{\mu} \quad & \sum_{s;j} \theta_{s;j}\,\mu(s;j) + \sum_{s,t;j,k} \theta_{s,t;j,k}\,\mu(s;j)\,\mu(t;k) \\ \text{subject to} \quad & \sum_{j} \mu(s;j) = 1 \\ & \mu(s;j) \in \{0,1\} \\ & \mu(s,j;t,k) \in \{0,1\}. \end{aligned} \tag{4.25}$$

This is clearly equivalent to the MAP problem in equation (4.19). Let $\Theta = [\theta_{s,j;t,k}]$ be a parameter matrix that is not negative semi-definite. Let $d(s,i)$ be the (positive) diagonal terms that need to be subtracted from the matrix to make it negative semi-definite. An upper bound for d is $d(s,i) \leq \sum_{(t,k)} |\theta_{s,j;t,k}|$ (since the negative of a diagonally dominant matrix is negative semi-definite). Let $\Theta' = \Theta - \text{diag}\{d(s;i)\}$ be the negative semi-definite matrix obtained by subtracting off diagonal elements $d(s;i)$. Also, let

$$\theta'_{s;j} = \theta_{s;j} + d(s;j). \tag{4.26}$$

Now, for binary $\mu(s;i) \in \{0,1\}$, we have that $\mu(s;i)^2 = \mu(s;i)$; in particular, $d(s;i)\mu(s;j) - d(s;i)\mu(s;j)^2 = 0$. We thus get that the following QIP is equivalent to the QIP (4.25),

$$\begin{aligned} \max_{\mu} \quad & \sum_{s;j} \theta'_{s;j}\,\mu(s;j) + \sum_{s,t;j,k} \theta'_{s,j;t,k}\,\mu(s;j)\,\mu(t;k) \\ \text{such that} \quad & \sum_{j} \mu(s;j) = 1 \\ & \mu(s;j) \in \{0,1\} \end{aligned}$$

Relaxing this QIP as before, we obtain the following convex program.

$$\begin{aligned} \max_{\mu} \quad & \sum_{s;j} \theta'_{s;j}\,\mu(s;j) + \sum_{s,t;j,k} \theta'_{s,j;t,k}\mu(s;j)\mu(t;k) \\ \text{such that} \quad & \sum_{j} \mu(s;j) = 1 \\ & \mu(s;j) \in [0,1] \end{aligned}$$

This is a convex program solvable in polynomial time. The optimality results of the previous section do not follow, however, and the relaxation (4.27) is not always tight. But as shown next, we can get an additive approximation bound for the discrete solution obtained using the rounding procedure described in the previous section.

Theorem 4. *Let μ^* be the optimal solution for the convex QP* (4.27)*; and e^* be the optimal MAP energy. Then there is a discrete configuration y (from μ^*) with energy $E(y)$ satisfying*

$$\begin{aligned} E(y) &\geq e^* - \sum_{s,i} d(s;i)\mu^*(s;i)(1-\mu^*(s;i)) \\ &\geq e^* - \frac{1}{4}\sum_{s,i} d(s;i). \end{aligned}$$

This result shows that if either Θ is close to negative definite, so that $\sum_{s,i} d(s;i)$ is small, or if the solution is close to integral, so that $\mu^*(s;i)$ is close to zero or one, then the convex relaxation achieves a solution that is close to the optimal MAP solution.

4.6 Iterative Update Procedure

Just as tree-reweighted max product gives a set of iterative updates for solving the LP in equation (4.21), we might ask if there is an iterative update counterpart for the QP. Max-product is a co-ordinate ascent algorithm in the dual (Lagrangian) space for the LP; however, since the dual space of the QP (4.23) is more complicated, we look at a set of fixed point co-ordinate ascent updates in its primal space.

The QP is given by

$$\begin{aligned} \mu^* = \max_{\mu} & \sum_{s;j} \theta_{s;j}\mu(s;j) \\ & + \sum_{s,t;j,k} \theta_{s,j;t,k}\,\mu(s;j)\,\mu(t;k) \end{aligned} \tag{4.27}$$

subject to

$$\begin{aligned} \sum_j \mu(s;j) &= 1 \\ \mu(s;j) &= [0,1]. \end{aligned}$$

Consider node s, and suppose that values for $\mu(t;.)$ are fixed for other nodes $t \neq s$. Then the optimal parameter values $\mu(s;.)$ for node s are given by

$$\begin{aligned}\mu(s;.) &= \max_{\mu(s;.)} \sum_j \theta_{s;j}\mu(s;j) \\ &\quad + \sum_{t;j,k} \theta_{s,j;t,k}\,\mu(s;j)\,\mu(t;k)\end{aligned}$$

subject to $\sum_j \mu(s;j) = 1$. This is easily seen to be solved by taking

$$j^*(s) = \operatorname*{argmax}_j \theta_{s;j} + \sum_{t;j,k} \theta_{s,j;t,k}\mu(t;k)$$

and setting $\mu(s,j) = \mathcal{I}_{j^*(s)}(j)$. This is essentially the iterative conditional modes algorithm [5], which iteratively updates each node with a labeling that most increases the energy, holding fixed the labels of the other nodes.

A better iterative procedure, with stronger and faster convergence properties, albeit for convex programs, is projected conjugate gradient ascent [2]. Thus, another advantage of our convex approximation is that we can use conjugate gradient ascent as a simple iterative procedure that is guaranteed to converge (unlike coordinate ascent for max product). This makes our convex approximation to the QP applicable to large scale problems.

4.7 Inner Polytope Relaxations

In the previous section, we obtained a quadratic relaxation by imposing an "independence" constraint on the parameters $\mu(s,j;t,k)$ in equation (4.22). We also showed that this relaxation is actually tight, and is equivalent to the MAP problem. In this section, we show how one can think of this relaxation as the counterpart of mean-field for MAP, and how any of the corresponding relaxation counterparts of structured mean-field are also tight.

Consider [61]'s polytope formulation of MAP in equation (4.21), given by

$$\mu^* = \max_x \langle\theta,\phi\rangle = \sup_{\mu\in\mathcal{M}} \langle\theta,\mu\rangle$$

where $\mathcal{M}$ is the convex hull of all configuration potentials $\phi(x)$. The second equality follows from the fact that in a linear program, the optimum occurs at an extremal point $\phi(x^*)$. Thus, if $M_I \subset \mathcal{M}$ is any subset of the marginal polytope that includes all of the vertices, then the equations

$$\mu^* = \max_x \langle\theta,\phi\rangle = \sum_{\mu\in M_I} \langle\theta,\mu\rangle$$

still hold. In other words, any relaxation of the indicator variables to $\mu(s,j;t,k) \in \mathcal{M}_I$ would lead to a tight relaxation, as long as $\mathcal{M}_I$ contains all vertices. In contrast, tree-reweighted max product is not tight, since the domain set for its relaxed parameters is $LOCAL(G) \supseteq \mathcal{M}$; see Section 4.3.

As described in [61], one can think of structured mean field methods as inner polytope approximations. For the given graph G and a subgraph H, let $\mathcal{E}(H) = \{\theta \,|\, \theta_{st} = 0, \ \forall (s,t) \notin H\}$, where θ_{st} is the vector of natural parameters associated with edge (s,t). Now for the subgraph H, we can define the following set of moment parameters:

$$\mathcal{M}(G;H) = \{\mu \,|\, \mu = E_\theta[\phi(x)] \text{ for some } \theta \in \mathcal{E}(H)\}\,.$$

In essence, the moment parameters in $\mathcal{M}(G;H)$ must be realizable by a distribution that respects the structure of H. For any $H \subseteq G$, the relation $\mathcal{M}(G;H) \subseteq \mathcal{M}(G)$ thus always holds, and $\mathcal{M}(G;H)$ is an inner polytope approximation to $\mathcal{M}$. In particular, taking H to be the completely disconnected graph ($i.e.$ no edges) H_0, we have,

$$\begin{aligned}\mathcal{M}(G;H_0) \quad = \quad & \{\mu(s;j), \mu(s,j;t,k) \,| \\ & \quad 0 \le \mu(s;j) \le 1 \\ & \quad \mu(s,j;t,k) = \mu(s;j)\mu(t;k)\}\end{aligned}$$

which can be seen to be equal to the feasible set of the QP relaxation (4.23). For this subgraph $H = H_0$, the mean field relaxation thus becomes

$$\begin{aligned}& \sup_{\mu \in \mathcal{M}(G;H_0)} \langle \theta, \mu \rangle \\ & = \sup_{\mu \in \mathcal{M}(G;H_0)} \sum_{s;j} \theta_{s;j}\mu(s;j) + \sum_{st;jk} \theta_{s,j;t,k}\mu(s,j;t,k) \\ & = \sup_{\mu \in \mathcal{M}(G;H_0)} \sum_{s;j} \theta_{s;j}\mu(s;j) + \sum_{st;jk} \theta_{s,j;t,k}\mu(s;j)\mu(t;k)\end{aligned}$$

which is equivalent to the quadratic relaxation in equation (4.23). Thus, we can, in principle, use any "structured mean-field" relaxation of the form $\sup_{\mu \in M(G;H)} \langle \theta, \mu \rangle$ to solve the MAP *exactly*. The caveat is that this problem, like structured mean field, is a non-convex problem. However, while structured mean field only solves for an approximate value of the log-partition function, the results from Section 4.4 show that its counterpart for the MAP problem is exact, if the global optimum can be found.

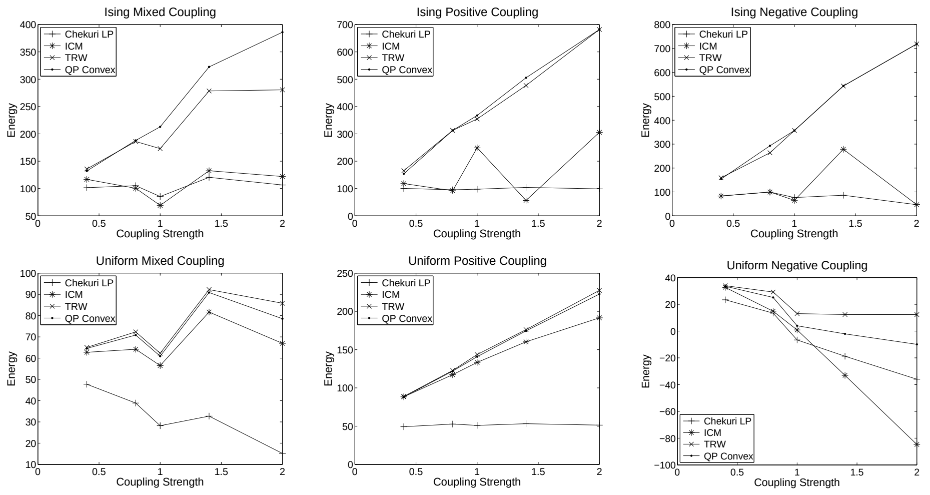


Figure 4.1: Comparison of linear relaxation (LP), iterative conditional modes (ICM), tree-reweighted max product (TRW), and quadratic programming relaxation (QP) on 10×10 grid graphs using Ising potentials (top row) and uniform potentials (bottom) with mixed (left), positive (center) and negative (right) couplings. A better MAP estimate has a higher value.

4.8 Experiments

We evaluated our quadratic relaxation with the convex approximation by comparing it against three competing methods: the linear programming relaxation [10], the tree-reweighted max product algorithm [57], and iterative conditional modes (ICM) [5]. For tree-reweighted max product, we use the sequential update variant detailed in [29], which has better convergence properties than the originally proposed algorithm.

The approximate MAP algorithms were compared on different potential functions and coupling types for 2D nearest neighbor grid graphs with 100 nodes and a label set of size four. The node potentials were generated uniformly $\mathcal{U}(-1, 1)$, while the edge potentials were generated as a product of an edge weight and a distance function on labels. For different settings of an edge coupling-strength parameter, d_{coup}, the edge weight was selected from $\mathcal{U}(-d_{coup}, d_{coup})$ for the mixed coupling, from $\mathcal{U}(0, 2\, d_{coup})$ for the positive coupling, and from $\mathcal{U}(-2\, d_{coup}, 0)$ for the negative coupling. The following four commonly used distances were used for the distance function: Ising, $\phi(l, m) = lm$; uniform, or Potts, $\phi(l, m) = \mathbb{I}(l = m)$; quadratic, $\phi(l, m) = (l - m)^2$; linear $\phi(l, m) = |l - m|$.

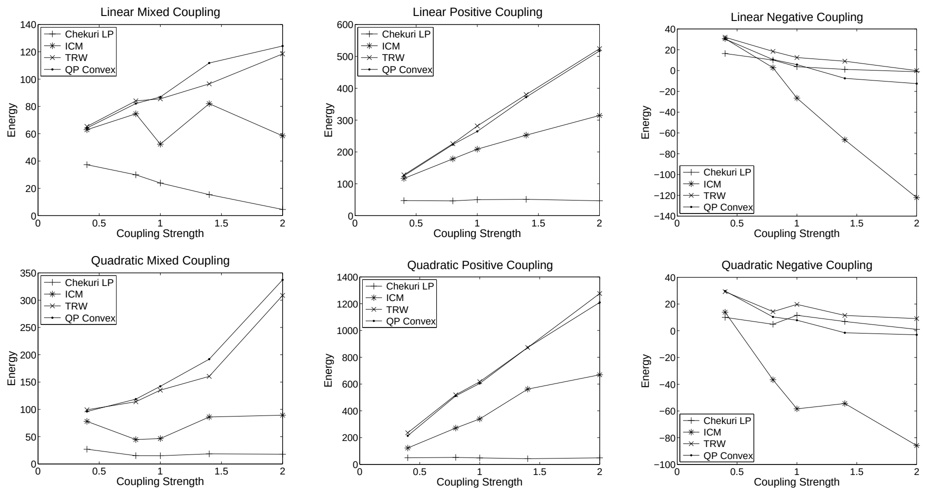


Figure 4.2: Comparison of linear relaxation (LP), iterative conditional modes (ICM), tree-reweighted max product (TRW), and quadratic programming relaxation (QP) on 10×10 grid graphs using linear potentials (top row) and quadratic potentials (bottom) with mixed (left), positive (center) and negative (right) couplings. A better MAP estimate has a higher value.

Figures (4.1) and (4.2) show plots of the value (energy) of the MAP estimates using the different algorithms for a range of model types. For any given setting of parameters and potential functions, a higher value is closer to the MAP estimate and is thus better. As the plots show, the quadratic relaxation clearly beats the tree-reweighted max product for mixed and positive couplings, and is comparable for the negative coupling. The quadratic approximation also typically beats both ICM and the linear relaxation.

In Figure (4.3) we compare the MAP estimates from different algorithms on larger graphs, using the Ising potential function with mixed coupling. The quadratic relaxation is seen to outperform ICM and tree-reweighted max product, even as the number of nodes increases.

We note that since the convex approximation to the QP is a convex program, it can be solved (in polynomial time) using standard QP solvers for small problems, and for larger-scale problems one can use iterative projected conjugate gradient, which provides a fast iterative method that is guaranteed to converge. In our experiments, the computation time for the QP method was comparable to that required by tree-reweighted max product, which in turn required much less time to solve than the linear programming relaxation. This is due primarily to the fact that the

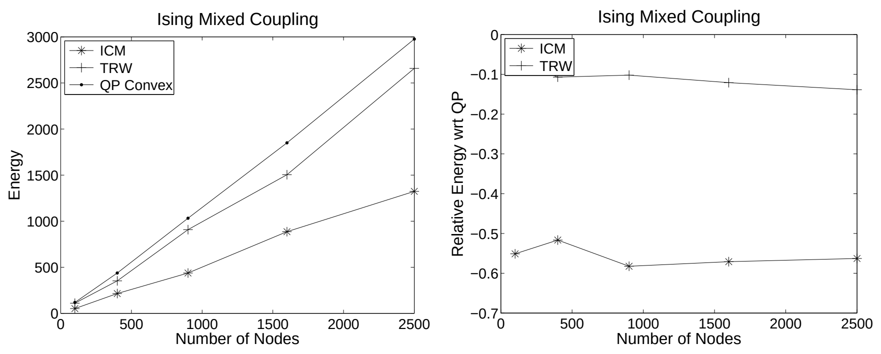


Figure 4.3: Comparison of ICM and TRW on larger graphs, using Ising potentials with mixed coupling. The right plot shows $(e_{\text{ICM}} - e_{\text{QP}})/e_{\text{ICM}}$ and $(e_{\text{TRW}} - e_{\text{QP}})/e_{\text{TRW}}$.

linear program has $|E|k^2$ variables, while the convex quadratic relaxation has only nk variables, where n is the number of nodes in the graph, $|E|$ is the number of edges, and k is the number of labels.

Chapter 5

General Event Probabilities, Bounds

In medical diagnosis, the task is to diagnose a disease given general features of the patient, such as sex and age, and measurements – such as body temperature – from physical examinations and medical equipments. In addition to the observed measurement and feature variables, there are unobserved "cause" variables; the given graphical model encodes the probabilistic relationship between all these variables. The task of medical diagnosis is then the inference task of estimating the probability of disease variables being true or false given the observed values for the measurement and feature variables.
With many intertwined latent causes, the network ends up having a large treewidth, which naturally motivates approximate inference procedures. There is however an additional demand: that of guarantees for the approximation; it is not enough to report just the "approximate" estimate for probability of the disease. There are two classes of guarantees we might provide for event probabilities: (a) constant factor guarantees: that the true values are within a specified constant factor of the approximate value. (b) additive or interval guarantees: that the true values are within a specified interval around the reported approximate value. [16] and others however dash hopes of the first class of guarantees; they show that constant factor approximations of event probabilities is NP-hard. In the next few chapters, we investigate the second kind of guarantees: providing rigorous upper and lower bounds for event probabilities.

In the quest for bounds, variational methods as well as the preconditioner approximation introduced in Chapter 3 are not without use: they provide bounds for the log partition function. Additionally, variational bounds often have associated dual parameters, and these parameters can be used as heuristic estimates of marginal probabilities. Unfortunately, there is a gap in understanding how such dual parameters can be quantitatively related to the actual marginal probabilities. For events more complicated than single or pairwise marginal probabilities, the variational machinery is not well-suited; in particular it might be difficult to obtain approximate estimates at all. In the next couple of chapters, we propose two classes of bounds: (a) variational Chernoff bounds, and (b) variational Chebyshev-Chernoff bounds.

Chapter 6

Variational Chernoff Bounds

In this chapter, we develop a class of bounds on event probabilities; variational Chernoff bounds; which combines the machineries of variational methods, and generalized Chernoff bounds. Consider a p-dimensional discrete random variable $X = (X_1, X_2, \ldots, X_p)$ whose distribution is governed by a parameter θ. Let $\mathcal{X}_s = \{1, \ldots, m_s\}$ be the domain of variable X_s, and denote $\mathcal{X} = \oplus_{s=1}^{p} \mathcal{X}_s$. A single node marginal probability is the probability of an event such as $C_i = \{\mathbf{X} : X_i = 1\}$; a pairwise marginal is the probability of an event such as $C_{ij} = \{\mathbf{X} : X_i = 1 \,\&\&\, X_j = 1\}$. In general, the events could involve all variables, $C_{sum} = \{\mathbf{X} : \sum_i X_i \leq \delta(\sum_i p_i)\}$, where $\{p_i\}$ are the single node marginals. Let $C \in \mathcal{X}$ denote a general event. The mandate of this chapter is to estimate the probability of this event, $P_\theta(X \in C)$.
Let us attempt to variationally estimate this event probability for a graphical model; represented by an exponential family distribution with sufficient statistics $\phi(X) \in \mathbb{R}^d$: $P_\theta(X) = \exp(\langle \theta, \phi(X) \rangle - \Phi(\theta))$. The event probability to be estimated is

$$P_\theta(X) = \sum_{X \in C} \exp(\langle \theta, \phi(X) \rangle - \Phi(\theta)) \tag{6.1}$$

where $\Phi(\theta)$ is the log partition function. We will denote by $\Phi(f, \theta)$ the log partition function for the (generally non-graphical) model with probabilities proportional to $\exp\left(\langle \theta, \phi(x) \rangle + f(x)\right)$; thus,

$$\Phi(f, \theta) = \sum_x \exp\left(\langle \theta, \phi(x) \rangle + f(x)\right) \tag{6.2}$$

The *indicator function* δ_C of the set $\mathcal{C}$ is defined as

$$\delta_C(x) = \begin{cases} 0 & \text{if } X \in C \\ \infty & \text{otherwise} \end{cases} \tag{6.3}$$

This allows us to write the event probability as,

$$\log p(x \in C \,|\, \theta) \;=\; \Phi(-\delta_C, \theta) - \Phi(\theta) \tag{6.4}$$

Denoting $\Phi(-\delta_C, \theta) = \Phi_C(\theta)$, the task of estimating the event probability reduces to estimating $\Phi(-\delta_C, \theta)$.

To obtain upper and lower variational bounds; let $\Phi^{(U)}(\theta)$ and $\Phi^{(L)}(\theta)$ be upper and lower bounds on $\Phi(\theta)$, respectively. Then,

$$\log p_\theta(X \in C) \;\leq\; \Phi_C^{(U)}(\theta) - \Phi^{(L)}(\theta) \tag{6.5}$$

$$\log p_\theta(X \in C) \;\geq\; \Phi_C^{(L)}(\theta) - \Phi^{(U)}(\theta) \tag{6.6}$$

For simple events like marginals, $\{X_1 = 1\}$, $\Phi_C(\theta)$ can be written as the log-partition function of a simple exponential family distribution,

$$\Phi_C(\theta) = \sum_{X_{-1}} \exp(\langle \theta, \phi_{-1}(X_{-1}) \rangle) \tag{6.7}$$

where $\phi_{-1}(X_{-1}) = \phi([1\, X_{-1}])$. Variational methods can then be naturally applied to obtain bounds. It is not obvious however, how to apply the variational machinery to get bounds $\Phi_C^{(U)}(\theta)$ for complicated events C.
A complementary mode of attack is to “separate” the contribution of the event C from the partition function, so as to make it more amenable to variational machinery. To obtain further intuition for this, let us consider the independent graphical model, which corresponds to $X_1, \ldots, X_p$ being i.i.d random variables. Bounding event probabilities for this case is a classical problem, a classical technique for which is that of Chernoff bounds. We first review Chernoff bounds, and show how they can be adapted to yield “variational Chernoff” bounds.

6.1 Classical and Generalized Chernoff Bounds

Let X be a real-valued random variable with distribution determined by some parameter θ. The Markov inequality implies that for any $\lambda > 0$,

$$\begin{aligned} p_\theta(X \geq u) &= p_\theta\left(e^{\lambda X} \geq e^{\lambda u}\right) && (6.8)\\ &\leq E_\theta[e^{\lambda(X-u)}] && (6.9) \end{aligned}$$

From this it follows that

$$\log p_\theta(X \geq u) \leq \inf_{\lambda \geq 0} \left(-\lambda u + \log E_\theta \left[e^{\lambda X}\right]\right) \tag{6.10}$$

In the classical formulations of Chernoff bounds that are so widely used in probabilistic analysis, the cumulant function $\log E_\theta\left[e^{\lambda X}\right]$ in relation (6.10) is further manipulated so that the upper bound has an analytic form. For example, if the random variable is $X \sim$ Binomial(n, p), it can easily be shown (derived in the next section) that

$$p_\theta\left(X < np\left(1-\delta\right)\right) \leq e^{-n\delta^2/2} \tag{6.11}$$

[7] observe that the basic idea behind inequality (6.10) can be considerably generalized in a way that involves convex optimization. Let X now denote a $\mathbb{R}^m$-valued random variable, whose distribution is indicated by a parameter θ, and let $C \subset \mathbb{R}^m$. To bound the probability $p_\theta(X \in C)$, consider a parameterized family of upper bounds $f_\lambda(x)$ on the indicator function $-\delta_C$; $f_\lambda(x) \geq 0$ if $x \in C$. Then clearly

$$\begin{aligned} -\delta_C &\leq f_\lambda && (6.12)\\ p_\theta(X \in C) &\leq \inf_\lambda E_\theta\left[e^{f_\lambda}\right] && (6.13) \end{aligned}$$

Where we get the second inequality from the first, by exponentiating both sides and then taking an expectation over X.

In case $f_\lambda = \langle \lambda, x\rangle + u$ is affine, where λ and u are chosen subject to the constraint that $\langle \lambda, x\rangle + u \geq 0$ for $x \in C$, we get

$$\begin{aligned} \log p_\theta(X \in C) &\leq \inf_{\lambda,u} \log E_\theta\left[e^{\langle\lambda,x\rangle+u}\right] && (6.14)\\ &= \inf_{\lambda,u}\left(u + \log E_\theta\left[e^{\langle\lambda,x\rangle}\right]\right) && (6.15) \end{aligned}$$

Now, since $u \geq \langle -\lambda, x\rangle - \delta_C(x)$, it follows that

$$\begin{aligned} \inf u = &\sup_x \langle -\lambda, x\rangle - \delta_C(x) && (6.16)\\ = &\ \delta_C^*(-\lambda) && (6.17) \end{aligned}$$

where $\delta_C^*(-\lambda)$ is the fenchel conjugate dual of $\delta_C(x)$. Therefore,

$$\log p_\theta(X \in C) \leq \inf_\lambda \left(\delta_C^*(-\lambda) + \log E_\theta\left[e^{\langle\lambda,x\rangle}\right]\right) \tag{6.18}$$

Let us now examine this conjugate dual $\delta_C^*(\lambda)$. It can be seen that

$$\begin{aligned} \delta_C^*(\lambda) &= \sup_x \langle \lambda, x\rangle - \delta_C(x) && (6.19)\\ &= \sup_{x\in C} \langle \lambda, x\rangle && (6.20) \end{aligned}$$

From equation 6.20; if x lies in C, $\langle x, \lambda \rangle \leq \delta_C^*(\lambda)$ for every λ; the conjugate dual $\delta_C^*(\lambda)$ is thus also called the support function of the set C. If C is convex, then $(\delta_C^*)^* = \delta_{\mathrm{cl}\, C}$. This shows that a closed convex set C can be represented as the solution set of a family of linear inequalities, and thus the support function characterizes C. We will also denote the support function by $S_C(\lambda)$.

6.2 Graphical Model Chernoff Bounds

For exponential family models, the line of argument of the previous section leads to the following bounds.

Proposition 1. *Suppose that $X = (X_1, \ldots, X_m)$ is an exponential model with (non-minimal) sufficient statistic $\phi(x) \in \mathbb{R}^n$, and let $C \subset \mathbb{R}^m$. Then*

$$\begin{aligned} \log p_\theta(X \in C) &= \Phi(-\delta_C, \theta) - \Phi(\theta) && (6.21) \\ &\leq \inf_\lambda \Phi(f_\lambda, \theta) - \Phi(\theta) && (6.22) \end{aligned}$$

for any family of functions $f_\lambda \geq -\delta_C$ bounding the indicator function.

Proof. The equality in (6.21) follows from

$$\begin{aligned} \log p_\theta(X \in C) &= \log \sum_{x \in C} e^{\langle \theta, \phi(x) \rangle} - \Phi(\theta) \\ &= \log \sum_x e^{-\delta_C(x) + \langle \theta, \phi(x) \rangle} - \Phi(\theta) \\ &= \Phi_C(\theta) - \Phi(\theta) \\ &\leq \log \sum_x e^{f_\lambda + \langle \theta, \phi(x) \rangle} - \Phi(\theta) \\ &= \Phi(f_\lambda, \theta) - \Phi(\theta) \end{aligned}$$

□

Proposition 2.

$$\log p_\theta(X \in C) \leq \inf_{\lambda \in \mathbb{R}^n} S_{C,\phi}(-\lambda) + \Phi(\lambda + \theta) - \Phi(\theta) \qquad (6.23)$$

where $S_{C,\phi}(y) = \sup_{x \in C} \langle y, \phi(x) \rangle$, for $y \in \mathbb{R}^n$.

Proof. Let $f_\lambda(x) = \langle \lambda, \phi(x) \rangle + u$ be an affine upper bound on $-\delta_C$. The bound in (6.23) then follows from Proposition 1 and (6.18) by observing that

$$\log \mathbb{E}_\theta \left[e^{\langle \lambda, \phi(X) \rangle} \right] = \Phi(\lambda + \theta) - \Phi(\theta)$$

□

In contrast to the initial attempt to apply variational machinery directly to $p_\theta(X \in C)$, the bound in the above proposition "separates" the event C contribution from the log-partition function; a form now amenable to log-partition function approximations. The separation is in terms of the *graphical model support function* of a set C, $S_{C,\phi}(y) = \sup_{x \in C} \langle y, \phi(x) \rangle$.

In case the vector of sufficient statistics includes each X_i, by restricting the linear function to one of the form $f_\lambda = \langle \lambda, x \rangle + u$ rather than $f_\lambda = \langle \lambda, \phi(x) \rangle + u$, we obtain a generally weaker bound of the form

$$\log p_\theta(X \in C) \leq \inf_{\lambda \in \mathbb{R}^m} S_C(-\lambda) + \Phi(\lambda + \theta) - \Phi(\theta) \tag{6.24}$$

where now $S_C = \delta_C^*$ is the standard support function.

Before we go on to incorporating variational machinery, let us look at an example each of classical and the graphical model Chernoff bounds.

6.3 Examples of Classical and Graphical Model Chernoff Bounds

6.3.1 Example: Classical Chernoff bounds

Classical Chernoff bounds [11, 40] are widely used to obtain rough, analytically convenient bounds on tail probabilities for iid observations. Let $X_1, X_2, \ldots, X_n$ are independent Bernoulli(p) trials, the upper Chernoff bound is established by using the Markov inequality to obtain

$$\log p(X \in C_\delta) \leq \inf_\lambda \left(-\lambda np\,(1+\delta) + E\left[e^{\lambda \sum_i X} \right] \right)$$

for $C_\delta = \{X \mid \sum_i X_i \geq np\,(1+\delta)\}$; this is equivalent to using the linear approximation to the indicator function employed above. Using the convexity of $\exp$, in the form $1 - x < e^{-x}$, the moment generating function $E[e^{\lambda \sum_i X_i}]$ is bounded from above as

$$\begin{aligned} \log E[e^{\lambda \sum_i X_i}] &= \sum_i \log \left(e^\lambda p + 1 - p \right) && (6.25) \\ &\leq np\,(1 - e^\lambda) && (6.26) \end{aligned}$$

This upper bound is then minimized to obtain the optimal $\lambda = \log(1+\delta)$, and thus the Chernoff bound

$$p(X \in C_\delta) \leq \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}} \right)^{np} \tag{6.27}$$

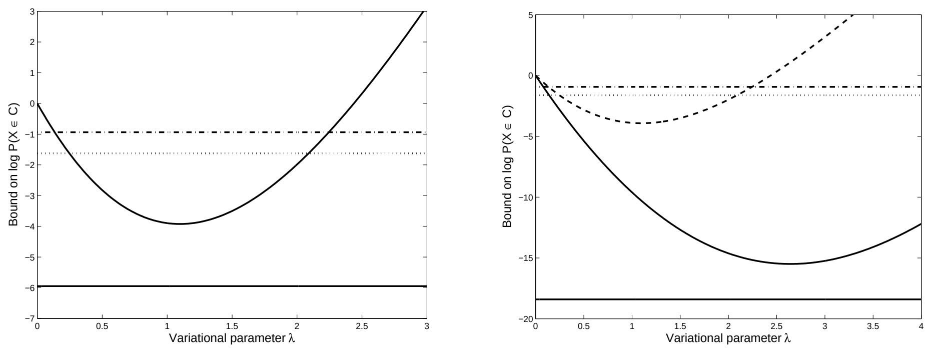


Figure 6.1: Classical and optimized Chernoff bounds for independent Bernoulli trials (left) and a Markov model (right) for $C_\delta = \{X \mid \sum X_i > np\,(1+\delta)\}$ with $p = \frac{1}{2}$ and $\delta = \frac{1}{2}$. Left: $n = 30$ Bernoulli trials—the classical Chernoff bound $\log P(X \in C_\delta) < -np\delta^2/4$ (top horizontal line), $\log P(X \in C_\delta) < np\,(\delta - (1+\delta)\log(1+\delta))$ (second horizontal line), and true probability (lower horizontal line); the curve shows the variational approximation $\log P(X \in C_\delta) < -\lambda np(1+\delta) - \Phi(\theta+\lambda) - \Phi(\theta)$. Right: bounds for a Markov model with $n = 30$, $\theta_{1,1} = -1$ and $\theta_1 = \log p/(1-p)$. The curved line is the variational approximation, where the log partition functions are computed exactly using dynamic programming; the bottom horizontal line is the true probability. The dashed curve is the variational approximation that assumes independent X_i (same as curve in left plot).

A more commonly used form, because of its simplicity, is the weaker bound

$$p(X \in C_\delta) \le e^{-np\,\delta^2/4} \tag{6.28}$$

which is valid when $\delta < 2e - 1$.

6.3.2 Example: Chernoff bounds for Markov models

One of the simplest extensions of the classical Chernoff bounds for independent Bernoulli trials is the case of a Markov or hidden Markov model. For illustration we consider a Markov model on two states, where the joint distribution for $X_1, \ldots, X_m$ with $X_i \in \{0, 1\}$ is given by

$$p(X_1, \ldots, X_n) \;\propto\; \exp\left(\sum_{i=1}^{n} \theta_{X_i} + \sum_{i=1}^{m-1} \theta_{X_i, X_{i+1}}\right) \tag{6.29}$$

Thus $\theta = (\theta_0, \theta_1, \theta_{0,0}, \theta_{0,1}, \theta_{1,0}, \theta_{1,1})$, with the case $\theta_{0,0} = \theta_{0,1} = \theta_{1,0} = \theta_{1,1} = 0$ corresponding to independent Bernoulli(p) trials with $p = e^{\theta_1}/(e^{\theta_0} + e^{\theta_1})$.

Since the random variables are not independent, the classical Chernoff bound for $p_\theta(\sum_i X_i > np\,(1+\delta))$ will be highly biased. The generalized Chernoff bound for the event $C_\delta = \{X \mid \sum_i X_i \geq np(1+\delta)\}$ is

$$p_\theta(X \in C_\delta) \leq \inf_\lambda -\lambda np\,(1+\delta) + \Phi(\theta + \bar{\lambda}) - \Phi(\theta)$$

where $\bar{\lambda} = (0, \lambda, 0, 0, 0, 0)$. In this case the log partition functions $\Phi(\theta+\lambda)$ and $\Phi(\theta)$ are easily computed in $O(n)$ time using dynamic programming. However, computing the probability $p_\theta(X \in C_\delta)$ exactly using dynamic programing requires $O(n^2)$ time—auxiliary states to count $\sum_i X_i$ must be introduced, requiring $O(n)$ states at each position. Similar statements can be made for graphical models where the underlying graph is a tree.

An example of these bounds for a simple Markov model is shown in Figure 6.1, where the bounds are compared to the classical bounds for the iid case. The left plot shows bounds for Bernoulli trials with $p = \frac{1}{2}$; the right plot shows bounds for a Markov model of the form (6.29) with $\theta_1 = \log \frac{p}{1-p}$ and $\theta_{1,1} = -1$, which discourages neighboring 1s.

Such a chain-structured graphical model is the simplest case of the generalized Chernoff bounds we consider. For more general graphical models, where dynamic programming may not be available, we must resort to more elaborate approximations.

6.4 Variational Chernoff Bounds

The exact log probability (6.21) and the generalized Chernoff bounds (6.23) require computation of log partition functions. In order to derive tractable bounds, we apply upper and lower variational bounds. Let $\Phi^{(U)}(\theta)$ and $\Phi^{(L)}(\theta)$ be upper and lower bounds on $\Phi(\theta)$, respectively. Then clearly

$$\log p_\theta(X \in C) \leq \Phi_C^{(U)}(\theta) - \Phi^{(L)}(\theta) \qquad (6.30)$$

$$\log p_\theta(X \in C) \geq \Phi_C^{(L)}(\theta) - \Phi^{(U)}(\theta) \qquad (6.31)$$

Applying the bounds to the graphical model Chernoff bounds (6.23) gives

$$\log p_\theta(X \in C) \leq \inf_{\lambda \in \mathbb{R}^n} S_{C,\phi}(-\lambda) + \Phi^{(U)}(\lambda + \theta) - \Phi^{(L)}(\theta) \qquad (6.32)$$

For upper bounds on the log-partition function, we employ (a) the log-determinant relaxations of [65] and (b) the tree-reweighted belief propagation algorithms of [63]. For the lower bounds, we use structured mean field approximations and M-best approximations using belief propagation [67].

6.4.1 Collapsing the Nested Optimization

Note however that (6.32) is a nested optimization problem: given any value of λ, the variational method solves an optimization problem to get a bound $\Phi^{(U)}(\lambda+\theta)$; the overall bound requires optimizing over λ in turn. It is possible however to collapse this nested optimization into into a single level optimization by "unraveling" the variational optimization. In this section, we investigate this for the log-determinant relaxations of [65].
[65] develop a semidefinite relaxation of $\Phi(\theta)$ which leads to a log determinant optimization problem. The idea behind this approach is to bound the dual function Φ^*, which is a negative entropy, in terms of the entropy of a Gaussian. Since the entropy of a Gaussian is a log determinant, the semidefinite upper bound follows. The analysis in [65] focuses on $\mathcal{X} = \{0, 1\}$ and vertex and pairwise interaction potentials on the complete graph K_n; this is the case we now assume, although the approach generalizes.

Recalling some of the notation of [65], for $\mu \in \mathbb{R}^m$, $M_1[\mu]$ is the $(n+1) \times (n+1)$ matrix

$$M_1[\mu] = \begin{pmatrix} 1 & \mu_1 & \cdots & \mu_n \\ \mu_1 & \mu_1 & \cdots & \mu_{1n} \\ \mu_2 & \mu_{21} & \cdots & \mu_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \mu_{n-1} & \mu_{(n-1),1} & \cdots & \mu_{(n-1),n} \\ \mu_n & \mu_{n1} & \cdots & \mu_n \end{pmatrix} \tag{6.33}$$

and $\text{SDEF}_1(K_n) = \{\mu \,|\, M_1[\mu] \succeq 0\}$. Let $M \supset \text{MARG}(K_n)$ be any convex set that contains $\text{SDEF}_1(K_n)$, and let $A(\mu) = M_1[\mu] + \frac{1}{12}\widetilde{I}$, where $\widetilde{I} = [0, I_n]$ is an $(n+1) \times (n+1)$ block diagonal matrix, and $c_n = \frac{n}{2}\log(2\pi e)$.

Theorem 1 of [65] states,

Proposition 3.

$$\Phi(\theta) \le \sup_{\mu \in M \;\, M_1[\mu] \succeq 0} \{\langle\theta, \mu\rangle + \frac{1}{2}\log\det A(\mu)\} + c_n \tag{6.34}$$

The outer-loop optimization over λ in (6.32) can be collapsed as in the following proposition, to yield a single-level optimization for the graphical model Chernoff bound,

Proposition 4.

$$\log p_\theta(X \in C) \le \sup_{\mu \in M} \left\{ \langle\theta, \mu\rangle + \frac{1}{2}\log\det A(\mu) - S^*_{C,\phi}(\mu) \right\} + c_n - \Phi(\theta) \tag{6.35}$$

The proof of this proposition follows from inequality (6.32) and the previous theorem, after observing that $S_{C,\phi}(\lambda)$, as a supremum of linear functions, is a convex function even if C is not convex, and that

$$\begin{aligned} S^*_{C,\phi}(\mu) &= \sup_\lambda \langle \lambda, \mu\rangle - S_{C,\phi}(\lambda) \qquad (6.36)\\ &= -\inf_\lambda \langle \lambda, \mu\rangle + S_{C,\phi}(-\lambda) \qquad (6.37)\end{aligned}$$

Solving the log determinant optimization problem above and replacing $\Phi(\theta)$ with any lower bound $\Phi^{(L)}(\theta)$ gives an upper bound on $p_\theta(X \in C)$.

6.5 Tightness of Chernoff Bounds

The generalized Chernoff bounds with linear approximations to the indicator function are actually *exact* expressions of event probabilities in an exponential family graphical model in certain cases. While the actual computation of the Chernoff bounds may be highly nontrivial, this result gives an indication of the power of the framework.

Proposition 5. *Let $p_\theta(X) = \exp(\langle \theta, \phi(X) - \Phi(\theta)\rangle)$ be an exponential model with $X = (X_1, \ldots, X_m)$, where $X \mapsto \phi(X) \in \mathbb{R}^n$ is a one-to-one mapping. Then for $C \subset \mathbb{R}^m$,*

$$\log p_\theta(X \in C) = \inf_{\lambda \in \mathbb{R}^n} S_{C,\phi}(-\lambda) + \Phi(\lambda + \theta) - \Phi(\theta)$$

[45] gives the detailed proof; we run through a particular example where we show the bounds are exact, and which gives some intuition.
Consider a mean-field graphical model,

$$p(X;\theta) = \exp\left(\sum_i \theta_i X_i - \Phi(\theta)\right) \qquad (6.38)$$

where the nodes X_i take values in $\{-1, +1\}$. $\Phi(\theta) = \sum_i \log(\exp(\theta_i) + \exp(-\theta_i))$ is the log-partition function. Consider the event $C = [x_1 = 1]$. The event probability $Pr(X \in C) = Pr(X_1 = 1)$ is given by,

$$\log p_\theta(X_1 = 1) = \theta_1 - \log(\exp(\theta_1) + \exp(-\theta_1)) \qquad (6.39)$$

The graphical model support function $S_C(-\lambda)$ can be calculated as,

$$S_C(-\lambda) = \sup_{x_1 = 1} -\lambda^T x = -\lambda_1 + \sum_{i=2}^n |\lambda_i| \qquad (6.40)$$

The graphical model Chernoff bound is given by,

$$\begin{aligned}
\log p_\theta(X \in C) &\leq \inf_\lambda (-\lambda_1 + \sum_{i \neq 1} |\lambda_i|) \\
&\qquad + \sum_i \log(\exp(\theta_i + \lambda_i) + \exp(-\theta_i - \lambda_i)) - \Phi(\theta) \\
&\leq \inf_{\lambda_1} -\lambda_1 + \log(\exp(\theta_1 + \lambda_1) + \exp(-\theta_1 - \lambda_1)) - \Phi(\theta) \\
&\qquad + \sum_{i \neq 1} \inf_{\lambda_i} (|\lambda_i| + \log(\exp(\theta_i + \lambda_i) + \exp(-\theta_i - \lambda_i))) \\
&\leq \theta_1 + \sum_{i \neq 1} \log(\exp(\theta_i) + \exp(-\theta_i)) - \Phi(\theta) \\
&\leq \theta_1 - \log(\exp(\theta_1) + \exp(-\theta_1)) \\
&= \log p_\theta(X \in C) \qquad (6.41)
\end{aligned}$$

which shows that the Chernoff bound is exact for this model.
The third inequality follows by by noting that

$$\inf_{\lambda_1} -\lambda_1 + \log(\exp(\theta_1 + \lambda_1) + \exp(-\theta_1 - \lambda_1)) = \theta_1 \qquad (6.42)$$

the infimum being obtained as $\lambda_1 \to \infty$; and that

$$\inf_{\lambda_i} (|\lambda_i| + \log(\exp(\theta_i + \lambda_i) + \exp(-\theta_i - \lambda_i))) \leq \log(\exp(\theta_i) + \exp(-\theta_i))$$

since the infimum is upper bounded by the value at $\lambda_i = 0$.

6.6 Experimental Results

To test the performance of the upper and lower bound methods, we performed experiments for binary random fields on both a complete graph and a 2-D nearest-neighbor grid graph, closely following the experiments in [65]. In order to be able to compare the bounds with the exact probabilities, we show results for small graphs with 9 nodes. For different qualitative characteristics of the exponential distributions (repulsive, mixed, or attractive), we construct many randomly generated models, and compute the mean error for each type of graph.

The graphical models were randomly generated according to the following specification. First, the parameters were randomly generated in the following manner:

Single node potentials: For each trial, we sample $\theta_s \sim$ Uniform$(-d_{\text{pot}}, +d_{\text{pot}})$ independently for each node, where $d_{\text{pot}} = \frac{1}{4}$.

Problem type			Average L_1 error $\pm$ std			
			Approximation method			
Graph	Coupling	Strength	MF/Tree lower	MF/SDP lower	Tree/MF upper	SDP heuristic
Grid	Repulsive	(0.25,1.0)	0.093 $\pm$ 0.003	0.297 $\pm$ 0.009	0.166 $\pm$ 0.008	0.010 $\pm$ 0.002
	Repulsive	(0.25,2.0)	0.127 $\pm$ 0.009	0.290 $\pm$ 0.007	0.327 $\pm$ 0.059	0.024 $\pm$ 0.002
	Mixed	(0.25,1.0)	0.054 $\pm$ 0.028	0.452 $\pm$ 0.047	0.070 $\pm$ 0.038	0.026 $\pm$ 0.002
	Mixed	(0.25,2.0)	0.095 $\pm$ 0.012	0.421 $\pm$ 0.053	0.138 $\pm$ 0.011	0.017 $\pm$ 0.003
	Attractive	(0.25,1.0)	0.026 $\pm$ 0.001	0.770 $\pm$ 0.019	0.025 $\pm$ 0.002	0.023 $\pm$ 0.001
	Attractive	(0.25,2.0)	0.001 $\pm$ 0.001	0.791 $\pm$ 0.026	0.001 $\pm$ 0.001	0.016 $\pm$ 0.002
Full	Repulsive	(0.25,0.25)	0.072 $\pm$ 0.010	0.290 $\pm$ 0.006	0.069 $\pm$ 0.011	0.021 $\pm$ 0.001
	Repulsive	(0.25,0.50)	0.132 $\pm$ 0.009	0.238 $\pm$ 0.007	0.156 $\pm$ 0.016	0.016 $\pm$ 0.001
	Mixed	(0.25,0.25)	0.032 $\pm$ 0.001	0.393 $\pm$ 0.014	0.029 $\pm$ 0.001	0.013 $\pm$ 0.004
	Mixed	(0.25,0.50)	0.120 $\pm$ 0.027	0.450 $\pm$ 0.037	0.127 $\pm$ 0.034	0.024 $\pm$ 0.004
	Attractive	(0.25,0.06)	0.009 $\pm$ 0.001	0.445 $\pm$ 0.009	0.007 $\pm$ 0.001	0.019 $\pm$ 0.003
	Attractive	(0.25,0.12)	0.037 $\pm$ 0.006	0.520 $\pm$ 0.023	0.033 $\pm$ 0.006	0.040 $\pm$ 0.003

Table 6.1: L_1 approximation error of single node marginals for the fully connected graph K_9 and the 4 nearest neighbour grid with 9 nodes, with varying potential and coupling strengths $(d_{\text{pot}}, d_{\text{coup}})$. Three different variational methods are compared: MF/Tree derives a lower bound with mean field approximation for Φ_C and tree-reweighted belief propagation for Φ; MF/SDP derives a lower bound with the SDP relaxation used for Φ; Tree/MF derives an upper bound using tree-reweighted belief propagation for Φ_C and mean field for Φ. SDP denotes the heuristic use of the dual parameters in the SDP relaxation, with no provable upper or lower bounds.

Edge coupling potentials: For a given coupling strength d_{coup}, three types of coupling are used:

$$\begin{aligned} \text{Repulsive:} \quad & \theta_{st} \sim \text{Uniform}(-2d_{coup}, 0) \\ \text{Mixed:} \quad & \theta_{st} \sim \text{Uniform}(-d_{\text{coup}}, +d_{\text{coup}}) \\ \text{Attractive:} \quad & \theta_{st} \sim \text{Uniform}(0, 2d_{\text{coup}}) \end{aligned}$$

For a given model, the marginal probabilities $p_\theta(X_s = 1)$ and $p_\theta(X_s = 1, X_t = 1)$ are computed exactly for each node and edge by calculating the log partition function exactly. Then, the variational Chernoff bounds on these probabilities are computed using different approximations to the log partition functions. As described in Section 6.4, we have $\log p_\theta(X \in C) = \Phi_C(\theta) - \Phi(\theta)$. In the case of the marginal at a single node, $C = \{x \in \mathbb{R}^n \,|\, x_s = 1\}$. We compute the bounds using the following methods:

MF/Tree: A lower bound on $\log p_\theta(X \in C)$ is computed by applying the structured mean field approximation to $\Phi_C(\theta)$ and the tree-reweighted belief propagation approximation to $\Phi(\theta)$.

Problem type			Average L_1 error $\pm$ std			
			Approximation method			
Graph	Coupling	Strength	MF/Tree lower	MF/SDP lower	Tree/MF upper	SDP heuristic
Grid	Repulsive	(0.25,1.0)	0.025 $\pm$ 0.003	0.118 $\pm$ 0.012	0.047 $\pm$ 0.008	0.005 $\pm$ 0.003
	Repulsive	(0.25,2.0)	0.034 $\pm$ 0.005	0.108 $\pm$ 0.010	0.101 $\pm$ 0.022	0.013 $\pm$ 0.001
	Mixed	(0.25,1.0)	0.026 $\pm$ 0.004	0.243 $\pm$ 0.022	0.037 $\pm$ 0.009	0.019 $\pm$ 0.005
	Mixed	(0.25,2.0)	0.056 $\pm$ 0.024	0.250 $\pm$ 0.035	0.087 $\pm$ 0.031	0.021 $\pm$ 0.006
	Attractive	(0.25,1.0)	0.029 $\pm$ 0.008	0.621 $\pm$ 0.076	0.043 $\pm$ 0.015	0.016 $\pm$ 0.012
	Attractive	(0.25,2.0)	0.002 $\pm$ 0.001	0.791 $\pm$ 0.012	0.003 $\pm$ 0.001	0.036 $\pm$ 0.007
Full	Repulsive	(0.25,0.25)	0.011 $\pm$ 0.002	0.081 $\pm$ 0.024	0.015 $\pm$ 0.001	0.021$\pm$ 0.004
	Repulsive	(0.25,0.50)	0.008 $\pm$ 0.005	0.046 $\pm$ 0.003	0.021 $\pm$ 0.002	0.021$\pm$ 0.003
	Mixed	(0.25,0.25)	0.040 $\pm$ 0.006	0.216 $\pm$ 0.013	0.014 $\pm$ 0.001	0.012 $\pm$ 0.007
	Mixed	(0.25,0.50)	0.068 $\pm$ 0.011	0.250 $\pm$ 0.033	0.052 $\pm$ 0.005	0.016 $\pm$ 0.011
	Attractive	(0.25,0.06)	0.020 $\pm$ 0.004	0.257 $\pm$ 0.017	0.003 $\pm$ 0.001	0.026 $\pm$ 0.007
	Attractive	(0.25,0.12)	0.061 $\pm$ 0.009	0.367 $\pm$ 0.019	0.015 $\pm$ 0.003	0.061 $\pm$ 0.005

Table 6.2: L_1 approximation error of pairwise node marginals. Approximation methods are as described for Table 1.

MF/SDP: A lower bound on $\log p_\theta(X \in C)$ is computed by the applying structured mean field approximation to $\Phi_C(\theta)$ and the semidefinite relaxation, resulting in a log determinant problem for $\Phi(\theta)$.

Tree/MF: An upper bound is derived using tree-reweighted belief propagation to upper bound $\Phi_C(\theta)$, and using structured mean field to derive a lower bound on $\Phi(\theta)$.

SDP: The semidefinite relaxation is used to heuristically estimate the marginal probability, as in [65], with no provable upper or lower bound.

To assess the accuracy of each approximation, we use the L_1 error, defined as

$$\frac{1}{n}\sum_{s=1}^{n} |p_\theta(X \in C) - \hat{p}_\theta(X \in C)| \tag{6.43}$$

where $\hat{p}_\theta$ denotes the estimated marginal. The results are shown in Table 1 for the single node case, and in Table 2 for the case of node pairs.

Chapter 7

Variational Chebyshev Bounds

The inference tasks of the earlier chapters all had a completely specified model; we were given the exponential family parameters, and we had to estimate the log-partition function or an event probability. In this chapter, we consider the case where we have to perform inference but we are given only partial information. In particular, we are given the expected values (moments) of some known functions, and we have to estimate an event probability; like in the previous chapter.
Formally, let X be a $p-$dimensional random variable, with domain $\mathcal{X}$, and distributed according to an unknown distribution p. Let $C \subseteq \mathcal{X}$ be an event; we wish to bound $p(X \in C)$. The only information we have about p is a set of moments: $E_p f_i = \sigma_i$, $i = 1, \ldots, k$.

Such a moment-matching problem arises naturally in MLE parameter estimation. Consider an exponential family distribution, $p_\theta(X) = \exp\left(\theta^\top \phi(X) - \Phi(\theta)\right)$, with feature function $\phi(X)$ and unknown parameter θ. Given data,
$D := \{x^{(1)}, \ldots, x^{(n)}\}$, the MLE parameters $\hat{\theta}$ are obtained by maximizing the log-likelihood,

$$\max_\theta \theta^\top \bar{\phi} - \Phi(\theta) \tag{7.1}$$

where $\bar{\phi} = \frac{1}{n} \sum_{j=1}^n \phi(X^{(j)})$. Setting the gradient to zero, we get the moment equations,

$$\mu(\hat{\theta}) = E_{\hat{\theta}}[\phi] = \bar{\phi} \tag{7.2}$$

In this case, the (MLE) parameters are completely specified by the moments; we investigate however the general case where such moment information need not completely specify the model.

A classical technique for this moment matching task is the Chebyshev bound. As in the Chernoff bound chapter, this is obtained by bounding the indicator function. Let $1_C(x)$ be the indicator function of the set C. Consider the following parameterized affine family of bounds on the indicator function, $\langle \lambda, f(x) \rangle + u \geq 0$ if $x \in C$. Taking an expectation of the affine expression gives,

$$\langle \lambda, \sigma \rangle + u \geq p[X \in C] \tag{7.3}$$

Thus we get the Chebyshev bound,

$$\begin{aligned} \inf_{\lambda,u} \quad & \langle \lambda, \sigma \rangle + u & (7.4) \\ s.t. \quad & \langle \lambda, f(x) \rangle + u \geq 1_C(x) & (7.5) \end{aligned}$$

This can be simplified, for instance by,

$$\inf_{\lambda} \quad \langle \lambda, \sigma \rangle + \max\{ \sup_{x \in C^c} -\lambda^\top f,\, 1 + \sup_{x \in C} -\lambda^\top f \} \tag{7.6}$$

For simple events C, such as a set given by polynomial inequalities, the constraint can be substituted with conditions for positivity of expressions. Consider an example from [7]: given the first and second moments, $\mu = E(x)$ and $\Sigma = E(xx^\top)$, the Chebyshev bound for the event probability $Pr(X \in C)$ can be written as

$$\begin{aligned} \min_{P,q,r} \quad & \mathbf{tr}(\Sigma P) + 2\mu^\top q + r \\ \text{s.t.} \quad & \mathbf{tr}(Pxx^\top) + 2q^\top x + r \geq 1_C(x) \end{aligned}$$

Consider the event $C = \mathbb{R}^p \backslash \mathcal{Y}$, where $\mathcal{Y} = \{x | a_i^\top x < b_i, i = 1, \ldots, k\}$.
[7] show that the Chebyshev bound problem can be rewritten as the following SDP,

$$\begin{array}{ll} \min & \mathbf{tr}(\Sigma P) + 2\mu^\top q + r \\ & \begin{bmatrix} P & q \\ q^\top & r-1 \end{bmatrix} \succeq \tau_i \begin{bmatrix} 0 & a_i/2 \\ a_i^T/2 & -b_i \end{bmatrix},\ i = 1, \ldots, k \\ \text{s.t.} & \tau_i \geq 0,\ i = 1, \ldots, k \\ & \begin{bmatrix} P & q \\ q^\top & r \end{bmatrix} \succeq 0 \end{array}$$

7.1 Graphical Model Chebyshev bounds

The Chebyshev bound in (7.4) bounds the event probability $p(X \in C)$ given moment information $E_p(f) = \sigma$. In the task we now consider, in addition to the moment information, we are also informed that the distribution belongs to a specified graphical model family; in particular, the graph structure of the model is known. We have already seen an example where the specification of the graphical model family would allow us to improve upon the Chebyshev bound: in MLE parameter estimation for an exponential family, given the moments of the feature functions, we can estimate the exact distribution parameters. In the general case however, we cannot estimate the parameters exactly, and we would like to incorporate the graphical model information into the Chebyshev machinery in (7.4). At first glance however, the bound in (7.4) does not have any remnant of the distribution left after taking the moments. Consider, on the other hand, its dual,

$$\sup_p \quad \sum_x p(x) 1_C(x) \tag{7.7}$$
$$\text{s.t.} \quad \sum_x p(x) f_i(x) = \sigma_i,\; i = 1, \ldots, k \tag{7.8}$$
$$\sum_x p(x) = 1 \tag{7.9}$$
$$p(x) \geq 0 \tag{7.10}$$

The dual form is very intuitive: it take the supremum of $p(X \in C)$ over all distributions p with the given moments. This thus suggests that instead of taking a supremum over *all* distribution p, we only take the supremum over distributions belonging to the specified graphical model family.

We first run through an example where we show that modifying the Chebyshev bound according to the above programme helps. Consider a chain-structured graphical model with three nodes; $X = (X_1, X_2, X_3)$, where the nodes are binary valued, taking values in $\{0, 1\}$. The graph structure is a chain with X_2 as the middle node. We are given the following moments:

$$\mathbb{E}(x_1 x_2) = q_{12} \tag{7.11}$$
$$\mathbb{E}(x_2 x_3) = q_{23} \tag{7.12}$$
$$\mathbb{E}(x_2) = q_2 \tag{7.13}$$

We wish to calculate a bound on the event probability $p(x_i = 1,\, i = 1, 2, 3)$. Using the Chebyshev bound without graph-structural information, we get,

$$\begin{array}{ll} \inf_\lambda & \lambda_{12} q_{12} + \lambda_{23} q_{23} + \lambda_2 q_2 + \lambda_4 \\ \text{s.t.} & \lambda_{12} x_1 x_2 + \lambda_{23} x_2 x_3 + \lambda_2 x_2 + \lambda_4 \geq x_1 x_2 x_3 \end{array}$$

This yields the bound: $\min\{q_{12}, q_{23}, q_2\}$.

The graphical model information stipulates,

$$p = p_{12}(x_1, x_2)\, p_{23}(x_2, x_3)\, / p_2(x_2) \tag{7.14}$$

Modifying (7.7) to incorporate this, we get

$$\begin{array}{ll} \text{sup} & p_{12}(x_1 = x_2 = 1)\, p_{23}(x_2 = x_3 = 1)\, / p_2(x_2 = 1) \\ \text{s.t.} & p_{12}(x_1 = x_2 = 1) = q_{12},\, p_{23}(x_2 = x_3 = 1) = q_{23},\, p_2(x_2 = 1) = q_2 \end{array}$$

which trivially gives the exact value for the event probability $q_{12}\, q_{23}\, / q_2$.

7.2 Chebyshev-Chernoff Bounds

To formalize the intuition of the previous example, suppose we are informed that the distribution p belongs to an exponential family with feature function ϕ, so that, $p_\theta(X) = \exp(\theta^\top \phi(X) - \Phi(\theta))$. Let us first rewrite the dual Chebyshev bound in (7.7) as follows,

$$\sup_p \quad \log \mathbb{E}_p 1_C(x) \tag{7.15}$$

$$\text{s.t.} \quad \log \mathbb{E}_p f = \sigma \tag{7.16}$$

where σ is now the logarithm of the moments. Since we know the parametric form of p, we can rewrite this as,

$$\sup_\theta \quad \Phi(\theta; 1_C) - \Phi(\theta) \tag{7.17}$$

$$\text{s.t.} \quad \Phi(\theta; f_i) - \Phi(\theta) = \sigma_i,\, i = 1, \ldots, k \tag{7.18}$$

where $\Phi(\theta; g) = \log \sum_x \exp(\theta^\top \phi(x)) g(x)$. Its dual is given by,

$$\inf_\rho \sup_\theta \rho^\top \sigma + \Phi(\theta; 1_C) + \left(\sum_i \rho_i - 1 \right) \Phi(\theta) + \sum_i \rho_i \Phi(\theta; f_i) \tag{7.19}$$

While this form could be optimized as is for simple events C, we can use the Chernoff bound results of the previous chapter to "separate" the contribution from the event C, to give a *Chebyshev-Chernoff* bound.

$$\begin{aligned} \inf_\rho \sup_\theta \inf_\lambda \quad & \rho^\top \sigma + S_{C,\phi}(-\lambda) + \Phi(\lambda + \theta) \\ & + \left(\sum_i \rho_i - 1 \right) \Phi(\theta) + \sum_i \rho_i \Phi(\theta; f_i) \end{aligned} \tag{7.20}$$

We note that this can be solved by co-ordinate descent, alternating between ρ, θ and λ. The gradients with respect to θ involve computing the graphical model moments (for the current estimate of θ); these can be approximated by approximate inference procedures. We formally state the Chebyshev-Chernoff bound in the following proposition,

Proposition 6. *Suppose that $X = (X_1, \dots, X_m)$ is distributed according to an exponential model p with (non-minimal) sufficient statistic $\phi(x) \in \mathbb{R}^n$, and let $C \subset \mathbb{R}^m$. Suppose the distribution p satisfies the following moment constraints, $\mathbb{E}_p(f_i) = \exp(\sigma_i)$, $i = 1, \dots, k$. Let $C \subset \mathbb{R}^m$. Then,*

$$p(X \in C) \leq \inf_{\rho} \sup_{\theta} \inf_{\lambda} \quad \rho^\top \sigma + S_C^\phi(-\lambda) + \Phi(\lambda + \theta) + (\sum_i \rho_i - 1)\Phi(\theta) + \sum_i \rho_i \Phi(\theta; f_i) \tag{7.21}$$

Part II

Structure Learning

Chapter 8

Structure From Data

The graph structure is a primary ontological component of a graphical model, and consequently estimating it from data is of utmost importance. Consider a p-dimensional discrete random variable $X = (X_1, X_2, \ldots, X_p)$ whose distribution is governed by an unknown undirected graphical model. We investigate estimating the graph structure from an i.i.d. sample D of n data points $\{x^{(i)} = (x_1^{(i)}, \ldots, x_p^{(i)}\}_{i=1}^n)\}$.
The representation theory of graphical models is built on the back of conditional independences: the lack of an edge (i, j) represents the Markov independence assumption, $X_i \perp X_j | X_{V \setminus i,j}$. This motivates "constraint based approaches" which use hypothesis testing to estimate the set of conditional independences in the data, and then determine a graph that most closely represents those independences [47]. These approaches however work best in settings with more data and less nodes. An alternative approach is to view the graph structure estimation as a search problem. This has two components: (a) a graph scoring metric; that combines a goodness of fit measure of the graph to the data (likelihood of the MLE parameters given the graph for instance) and a graph complexity penalty; and (b) a heuristic search procedure that generates candidate graph structures to be scored. The number of undirected graphs with p nodes is $2^{\binom{p}{2}}$ however; Chickering [12] shows that this search problem is NP-hard. Note that there are two complexity roadblocks in search based procedures: one is the combinatorial search, and the other is the computation of the score for any graph. The computation of typical score metrics however involves computing the normalization constant of the graphical model distribution, which is intractable for general undirected models. The space of candidate structures in such scoring based approaches is thus typically restricted to directed models (Bayesian networks), for which the normalization constant is trivially one.

This depleted armory has thus restricted the estimation of graph structures in undirected models to simple graph classes such as trees [13], polytrees [17] and bounded tree-width hypertrees [48].
We now investigate the use of the "optimization" paradigm, which had proved to be of such good use in various inference tasks. This demands we parametrize the search space of graph structures: we can then cast the search procedure as a parametrized optimization problem. In the next section we investigate the use of a natural edge selection parametrization, and show how it leads to ℓ_1 regularized MLE estimation.

8.1 Parameterizing edge selection

We focus on the setting of discrete undirected graphical models. Revisiting our standard notation, let $\mathcal{X}_s = \{1, \ldots, m_s\}$ be the domain of variable X_s; for $j \in \mathcal{X}_s$ let $\mathbb{I}_j(x_s)$ be the indicator function for the event $\{x_s = j\}$. A pairwise undirected graphical model for $X = (X_1, X_2, \ldots, X_p)$ with graph structure G and indicator function potentials is then given by,

$$p(X;\theta;G) = \exp\left(\sum_{s;j} \theta_{s;j}\mathbb{I}_j(X_s) + \sum_{(s,t)\in E;j,k} \theta_{st;jk}\mathbb{I}_{j,k}(X_s, X_t) - \Phi(\theta)\right) \tag{8.1}$$

The maximum likelihood estimate of the parameters θ, given the i.i.d observations D and graph structure G, is given by

$$\hat{\theta}_G = \operatorname*{argmax}_{\theta} \frac{1}{n}\sum_{i=1}^{n} \log p(X^{(i)};\theta;G) \tag{8.2}$$

The structure estimation problem can then be written as,

$$\hat{G} = \operatorname*{argmax}_{G} \frac{1}{n}\left(\sum_{i=1}^{n} \log p(X^{(i)};\hat{\theta}_G;G)\right) + c(G) \tag{8.3}$$

where $c(G)$ is a function that penalizes the complexity of the graph. Typical penalty functions, including the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC) [9], are proportional to the number of "free parameters"; which in a pairwise graphical model is proportional to the number of edges. We thus write the penalty function as $c(G) = \lambda|E(G)|$. From equations (8.2),(8.3), the structure learning task can be written as the joint regularized

MLE estimation of (θ, G),

$$\sup_{\theta,G} \frac{1}{n}\sum_{i=1}^{n} \log p(X^{(i)};\theta;G) + c(G) \tag{8.4}$$

The mandate of this section was to parametrize the graph structure G to a form more amenable to optimization. A natural parametrization is the edge-appearance vector $\{z_{st}(G) = \mathbb{I}[(s,t) \in E(G)]\}$. Under this parametrization, the likelihood can be written as,

$$p(x;\theta;G) := p(x;\theta;\mathbf{z}) \tag{8.5}$$

$$= \exp\left(\sum_{s;j}\theta_{s;j}\mathbb{I}_j(x_s) + \sum_{s,t;j,k} z_{st}\theta_{st;jk}\mathbb{I}_{j,k}(x_s,x_t) - \Phi(\theta;z)\right) \tag{8.6}$$

where $\Phi(\theta;z)$ is the log-partition function

$$\Phi(\theta;z) = \log\sum_x \exp\left(\sum_{s;j}\theta_{s;j}\mathbb{I}_j(x_s) + \sum_{s,t;j,k} z_{st}\theta_{st;jk}\mathbb{I}_{j,k}(x_s,x_t)\right)$$

Noting that the penalty function $c(G) = \lambda|E(G)|$ becomes $c(z) = \lambda\sum_{st} z_{st}$; the structure estimation problem in Equation 8.4 can thus be written as,

$$\sup_{\theta;\, z\in\{0,1\}^{\binom{p}{2}}} \sum_{s;j}\theta_{s;j}\bar{\mu}_{s;j} + \sum_{s,t;j,k} z_{st}\theta_{st;jk}\bar{\mu}_{st;jk} \ - \Phi(\theta;z) + \lambda\sum_{st} z_{st} \tag{8.7}$$

where $\bar{\mu}$ are the average counts; $\bar{\mu}_{s;j} = \frac{1}{n}\sum_i \mathbb{I}[x_s^{(i)} = j]$.
There are two sources of complexity; the log-partition function, and the optimization of the discrete vector z over the $\{0,1\}^{\binom{p}{2}}$ hypercube. With the aim of approximating the log-partition function; substitute in the conjugate dual $\Phi^*(\bar{\mu};z) = \sup_\theta \sum_{s;j}\theta_{s;j}\bar{\mu}_{s;j} + \sum_{s,t;j,k} z_{st}\theta_{st;jk}\bar{\mu}_{st;jk} \ - \Phi(\theta;z)$ in equation 8.7, to get,

$$\sup_{\mathbf{z}} \Phi^*(\bar{\mu};z) + \lambda\sum_{st} z_{st} \tag{8.8}$$

We can then use variational approximations of the entropy function $\Phi^*(\mu)$ to obtain approximate solutions to z.
As a canonical example, consider the Bethe approximation to the dual entropy function; which is exact if the graph corresponding to z is tree-structured,

$$\Phi^*(\bar{\mu};z) = -\sum_s H_s(\bar{\mu}_s) + \sum_{st} z_{st} I_{st}(\bar{\mu}_{st}) \tag{8.9}$$

where H_s is the single node entropy and I_{st} is the mutual information,

$$H_s(\mu_s) := -\sum_j \mu_{s;j} \log \mu_{s;j} \tag{8.10}$$

$$I_{st}(\mu_{st}) := \sum_{jk} \mu_{st;jk} \log \frac{\mu_{st;jk}}{\mu_{s;j}\mu_{t;k}} \tag{8.11}$$

Equation 8.8 can then be written as,

$$\sup_{\mathbf{z}} \sum_{st} z_{st}(\lambda + I_{st}(\bar{\mu}_{st})) \tag{8.12}$$

Optimizing over $z \in \{0,1\}^{\binom{p}{2}}$ gives:

$$z_{st} = \mathbb{I}[I_{st}(\bar{\mu}_{st}) > -\lambda] \tag{8.13}$$

Optimizing over tree-structured z: Equation (8.12) suggests we weight each edge (s,t) with $I_{st}(\bar{\mu}_{st})$ and solve for the maximum spanning tree; this is also the Chow Liu algorithm [14].
More complicated variational approximations to the dual entropy function in Equation 8.8 still leaves us with the intractable optimization over the discrete valued z; so let us go back to Equation 8.7 and attempt to "relax" the discrete optimization over z. A natural relaxation is to allow optimization of z_{st} over positive reals, and scale $\{\theta_{st;jk}, j \in \mathcal{X}_s,\ k \in \mathcal{X}_t\}$ for identifiability (of z),

$$\begin{aligned} (SE:I) \sup_{\theta;z} \quad & \sum_{s;j} \theta_{s;j}\bar{\mu}_{s;j} + \sum_{s,t;j,k} z_{st}\theta_{st;jk}\bar{\mu}_{st;jk} \ - \Phi(\theta;z) + \lambda \sum_{st} z_{st} \\ s.t. \quad & \sum_{jk} \theta^2_{st;jk} = 1,\ \ z \geq 0 \end{aligned} \tag{8.14}$$

It can be seen that the constraint set in Equation 8.14 is a relaxation of the hypercube set $\{0,1\}^{\binom{p}{2}}$. The problem is convex in z and θ separately, and can be solved by an alternating ascent procedure, optimizing over θ and z in turn. Let us however simplify the problem further. Consider the following optimization problem,

$$(SE:II) \sup_{\theta} \quad \sum_{s;j} \theta_{s;j}\bar{\mu}_{s;j} + \sum_{s,t;j,k} \theta_{st;jk}\bar{\mu}_{st;jk} \ - \Phi(\theta) + \lambda \sum_{st} \sqrt{\sum_{jk} \theta^2_{st;jk}} \tag{8.15}$$

It can be seen that problems (SE: I) and (SE: II) are equivalent

$$(\{z^*_{st}\}, \{\theta^*_s\}, \{\theta^*_{st}\}) \text{ optimizes } (SE:I) \text{ implies}$$
$$(\{\theta^*_s\}, \{\beta^*_{st} = z^*_{st}\,\theta^*_{st}\}) \text{ optimizes } (SE:II);$$

$$(\{\theta^*_s\}, \{\beta^*_{st}\}) \text{ optimizes } (SE:II) \text{ implies}$$
$$\left(\{z^*_{st} = \sqrt{\textstyle\sum_{jk} \theta^2_{st;jk}}\}, \{\theta^*_s\}, \{\theta^*_{st} = \beta^*_{st}/z^*_{st}\}\right) \text{ optimizes } (SE:I)$$

The penalty term in Problem (SE: II) is a sum or ℓ_1 norm over edges of the ℓ_2 norms of parameters of a single edge. This ℓ_1 norm penalty over edges, while derived as a relaxation, possesses many sparsity (structure) recovering properties. The problem is still not tractable however; the log-partition function still needs to be approximated. In the next chapter, we focus on the Ising model, where there is a single parameter per edge, so that the penalty is a simple ℓ_1 norm of the parameters; and analyze its structure recovering properties under a pseudo-likelihood approximation to the partition function.

Chapter 9

ℓ_1 regularized regression

We now focus on the problem of estimating the graph structure of a discrete Markov random field with Ising potentials. In the previous chapter, we reduced this to an ℓ_1 penalized MLE estimation problem, by first relaxing an edge-selection parametrization, and parameterizing to an equivalent problem. The edge-selection parametrization suggests viewing structure estimation as a "sparsity recovery" problem. An unknown "signal" of parameters or weights enters into generalized linear combinations with feature functions. What we observe are just noisy samples of these generalized linear combinations, but from which we have to recover the "sparsity pattern" of the signal: the locations of its non-zeros, which are the edges.
This "signal recovery" paradigm – recovering the signal from noisy samples of linear combinations – has been used in many fields; and a technique with a long history for the estimation of these sparse models or signals is ℓ_1 regularization; we refer to Tropp [52] for a recent survey. A surge of recent work has shown that ℓ_1-regularization can lead to practical algorithms for signal recovery with strong theoretical guarantees (e.g., [19, 41, 52]).
In the allied problem of sparsity recovery – the task of recovering the zero-pattern of the signal – the use of ℓ_1 regularization has been shown to give strong guarantees [58, 71] as well. To see why separate guarantees are required for sparsity recovery, aside from those for signal recovery, consider the sets $A = \{1, 1/n\}$ and $B = \{1, 0\}$. It can be seen that $\|A - B\|_2 \rightarrow 0$, so that the "estimate" A recovers the "signal" B convergently. But, denoting by $S(H)$ the support of set H, $S(A) = \{1, 2\}$ and $S(B) = \{1\}$ and $S(A) \not\rightarrow S(B)$; so that the estimate does not recover the sparsity pattern convergently. The rest of the chapter starts on where the last chapter left off, and analyzes the use of ℓ_1 regularized MLE estimation for structure estimation in Ising graphical models. Since MLE estimation is intractable for the fully connected graphical model; we consider the equivalent problem of es-

timating the neighborhood of each node: this would then suggest maximizing the ℓ_1 regularized conditional likelihood at each node, which is tractable. We then show that under suitable conditions this neighborhood estimation recovers the true graph structure with probability one. The consistency analysis is in the high dimensional setting, where the number of nodes in the graph, as well as the maximum neighborhood size is allowed to grow to infinity with the number of samples. This mode of analysis might at first seem discomfiting; are we getting samples from a "growing" truth, and more importantly how can one be consistent to this growing truth; to clear the confusions it is helpful to think of the setup as triangular. Given a sequence of graph parameters θ_n, graph structure E_n, and samples D_n, with the subscripts denoting the number of samples n, we show that our sequence of estimates $\hat{E}_n$, satisfy $p[\hat{E}_n = E_n] \to 1$. This non-classical mode of analysis provides a theoretical framework for studying very high dimensional problems where the sample size may be large absolutely, and yet small relative to the dimension of the problem; and is of considerable contemporary interest in statistics.

9.1 Problem Formulation and Notation

Let $G = (V, E)$ denote a graph with vertex set V of size $|V| = p$ and edge set E. We denote by $\mathrm{N}(s)$ the set of neighbors of a vertex $v \in V$; that is $\mathrm{N}(s) = \{(s,t) \in E\}$. A pairwise graphical model with graph G is a family of probability distributions for a random variable $X = (X_1, X_2, \ldots, X_p)$ given by $p(x) \propto \prod_{(s,t)\in E} \psi_{st}(x_s, x_t)$. We restrict our attention to the case where each $x_s \in \{0,1\}$ is binary, and the family of probability distributions is given by the Ising model

$$p(x;\theta) = \exp\left(\sum_{s\in V} \theta_s x_s + \sum_{(s,t)\in E} \theta_{st} x_s x_t - \Psi(\theta)\right). \tag{9.1}$$

Given such an exponential family in a minimal representation, the log partition function $\Psi(\theta)$ is strictly convex, which ensures that the parameter matrix θ is identifiable.

Given n samples $x^{(i)} \in \{0,1\}^p$ drawn from an unknown distribution $p(x;\theta^*)$ of the form (9.1), let $\hat{E}_n$ be an estimated set of edges. Our set-up includes the important situation in which the number of variables p may be large relative to the sample size n. In particular, we allow the graph $G_n = (V_n, E_n)$ to vary with n, so that the number of variables $p = |V_n|$ and the sizes of the neighborhoods $d_s := |\mathrm{N}(s)|$ may vary with sample size. (For notational clarity we will sometimes omit subscripts indicating a dependence on n.) The goal is to construct an estimator $\hat{E}_n$ for which $p[\hat{E}_n = E_n] \to 1$ as $n \to \infty$. Equivalently, we consider the problem of estimating neighborhoods $\hat{\mathrm{N}}_n(s) \subset V_n$ so that $p[\,\hat{\mathrm{N}}_n(s) = \mathrm{N}(s),\ \forall\, s \in V_n] \longrightarrow 1$. For many problems of interest, the graphical model provides a compact representation where the size of the neighborhoods are typically small—say $d_s \ll p$ for all $s \in V_n$. Our goal is to use ℓ_1-regularized logistic regression to estimate these neighborhoods; the actual values of the parameters θ_{ij} is a secondary concern.

Given input data $\{(z^{(i)}, y^{(i)})\}$, where $z^{(i)}$ is a p-dimensional covariate and $y^{(i)} \in \{0,1\}$ is a binary response, logistic regression involves minimizing the negative log likelihood

$$f_s(\theta; x) = \frac{1}{n}\sum_{i=1}^{n}\left\{\log(1+\exp(\theta^T z^{(i)})) - y^{(i)}\theta^T z^{(i)}\right\}. \tag{9.2}$$

We focus on regularized version of this regression problem, involving an ℓ_1 constraint on (a subset of) the parameter vector θ. For convenience, we assume that $z_1^{(i)} = 1$ is a constant so that θ_1 is a bias term, which is not regularized; we denote by $\theta_{\setminus s}$ the vector of all coefficients of θ except the one in position s. For the graph learning task, we regress each variable X_s onto the remaining variables, sharing the same data $x^{(i)}$ across problems. This leads to the following collection of optimization problems (p in total, one for each graph node):

$$\hat{\theta}^{s,\lambda} = \arg\min_{\theta\in\mathbb{R}^p}\left\{\frac{1}{n}\sum_{i=1}^{n}\left[\log(1+\exp(\theta^T z^{(i,s)})) - x_s^{(i)}\theta^T z^{(i,s)}\right] + \lambda_n\|\theta_{\setminus s}\|_1\right\}. \tag{9.3}$$

where $s \in V$, and $z^{(i,s)} \in \{0,1\}^p$ denotes the vector where $z_t^{(i,s)} = x_t^{(i)}$ for $t \neq s$ and $z_s^{(i,s)} = 1$. The parameter θ_s acts as a bias term, and is not regularized. Thus, the quantity $\hat{\theta}_t^{s,\lambda}$ can be thought of as a penalized conditional likelihood estimate of $\theta_{s,t}$. Our estimate of the neighborhood $\mathrm{N}(s)$ is then given by

$$\hat{\mathrm{N}}_n(s) = \left\{t \in V,\ t \neq s\ :\ \hat{\theta}_t^{s,\lambda} \neq 0\right\}.$$

Our goal is to provide conditions on the graphical model—in particular, relations among the number of nodes p, number of observations n and maximum node degree $d_{\max}$—that ensure that the collection of neighborhood estimates (9.1), one for each node s of the graph, is consistent with high probability.

We conclude this section with some additional notation that is used throughout the sequel. Defining the probability $p(z^{(i,s)};\theta) := [1+\exp(-\theta^T z^{(i,s)})]^{-1}$, straightforward calculations yield the gradient and Hessian, respectively, of the negative log likelihood (9.2):

$$\nabla_\theta f_s(\theta; x) \;=\; \frac{1}{n}\sum_{i=1}^{n} p(z^{(i,s)};\theta)\, z^{(i,s)} - \theta^T \left(\frac{1}{n}\sum_{i=1}^{n} x_s^{(i)} z^{(i,s)}\right) \qquad (9.4a)$$

$$\nabla^2_\theta f_s(\theta; x) \;=\; \frac{1}{n}\sum_{i=1}^{n} p(z^{(i,s)};\theta)\,[1 - p(z^{(i,s)};\theta)]\; z^{(i,s)}\,(z^{(i,s)})^T. \qquad (9.4b)$$

Finally, for ease of notation, we make frequent use the shorthand

$$Q_s(\theta) = \nabla^2 f_s(\theta; x)$$

9.2 Main Result and Outline of Analysis

In this section, we begin with a precise statement of our main result, and then provide a high-level overview of the key steps involved in its proof.

9.2.1 Statement of main result

We begin by stating the assumptions that underlie our main result. A subset of the assumptions involve the Fisher information matrix associated with the logistic regression model, defined for each node $s \in V$ as

$$Q_s^* \;=\; \mathbb{E}\Big[p_s(Z;\theta^*)\ \{1 - p_s(Z;\theta^*)\} Z Z^T\Big], \qquad (9.5)$$

Note that Q_s^* is the population average of the Hessian $Q_s(\theta^*)$. For ease of notation we use S to denote the neighborhood $\mathrm{N}(s)$, and S^c to denote the complement $V - \mathrm{N}(s)$. Our first two assumptions (A1 and A2) place restrictions on the dependency and coherence structure of this Fisher information matrix. We note that these first two assumptions are analogous to conditions imposed in previous work [38, 52, 58, 71] on linear regression. Our third assumption is a growth rate condition on the triple $(n, p, d_{\max})$.

[A1] Dependency condition: We require that the subset of the Fisher information matrix corresponding to the relevant covariates has bounded eigenvalues: namely, there exist constants $C_{min} > 0$ and $C_{max} < +\infty$ such that

$$C_{min} \;\leq\; \Lambda_{min}(Q^*_{SS}), \qquad and \qquad \Lambda_{max}(Q^*_{SS}) \;\leq\; C_{max}. \qquad (9.6)$$

These conditions ensure that the relevant covariates do not become overly dependent, and can be guaranteed (for instance) by assuming that $\hat{\theta}^{s,\lambda}$ lies within a compact set.

[A2] Incoherence condition: Our next assumption captures the intuition that the large number of irrelevant covariates (i.e., non-neighbors of node s) cannot exert an overly strong effect on the subset of relevant covariates (i.e., neighbors of node s). To formalize this intuition, we require the existence of an $\epsilon \in (0, 1]$ such that

$$\|Q^*_{S^cS}(Q^*_{SS})^{-1}\|_\infty \leq 1 - \epsilon. \tag{9.7}$$

Analogous conditions are required for the success of the Lasso in the case of linear regression [38, 52, 58, 71].

[A3] Growth rates: Our second set of assumptions involve the growth rates of the number of observations n, the graph size p, and the maximum node degree $d_{\max}$. In particular, we require that:

$$\frac{n}{d^5_{\max}} - 6d_{\max}\log(d_{\max}) - 2\log(p) \rightarrow +\infty. \tag{9.8}$$

Note that this condition allows the graph size p to grow exponentially with the number of observations (i.e., $p(n) = \exp(n^\alpha)$ for some $\alpha \in (0, 1)$. Moreover, it is worthwhile noting that for model selection in graphical models, one is typically interested in node degrees $d_{\max}$ that remain bounded (e.g., $d_{\max} = O(1)$), or grow only weakly with graph size (say $d_{\max} = o(\log p)$).

With these assumptions, we now state our main result:

Theorem 1. *Given a graphical model and triple $(n, p, d_{\max})$ such that conditions A1 through A3 are satisfied, suppose that the regularization parameter λ_n is chosen such that (a) $n\lambda_n^2 - 2\log(p) \rightarrow +\infty$, and (b) $d_{\max}\lambda_n \rightarrow 0$. Then $p[\,\hat{\mathrm{N}}_n(s) = \mathrm{N}(s),\ \forall\, s \in V_n] \rightarrow 1$ as $n \rightarrow +\infty$.*

9.2.2 Outline of analysis

We now provide a high-level roadmap of the main steps involved in our proof of Theorem 1. Our approach is based on the notion of a *primal witness*: in particular, focusing our attention on a fixed node $s \in V$, we define a constructive procedure for generating a primal vector $\widehat{\theta} \in \mathbb{R}^p$ as well as a corresponding subgradient $\widehat{z} \in \mathbb{R}^n$ that together satisfy the zero-subgradient optimality conditions associated with the convex program (9.3). We then show that this construction succeeds with probability converging to one under the stated conditions. A key fact is that the convergence rate is sufficiently fast that a simple union bound over all graph nodes shows that we achieve consistent neighborhood estimation for all nodes simultaneously.

To provide some insight into the nature of our construction, the analysis in Section 9.3 shows the neighborhood $\mathrm{N}(s)$ is correctly recovered if and only if the pair $(\widehat{\theta}, \widehat{z})$ satisfies the following four conditions: (a) $\widehat{\theta}_{S^c} = 0$; (b) $|\widehat{\theta}_t| > 0 \quad$ for all $t \in S$; (c) $\widehat{z}_S = \mathrm{sgn}(\theta^*_S)$; and (d) $\|\widehat{z}_{S^c}\|_\infty < 1$. The first step in our construction is to choose the pair $(\widehat{\theta}, \widehat{z})$ such that both conditions (a) and (c) hold. The remainder of the analysis is then devoted to establishing that properties (b) and (d) hold with high probability.

In the first part of our analysis, we assume that the dependence (A1) mutual incoherence (A2) conditions hold for the *sample Fisher information matrices* $Q_s(\theta^*)$ defined below equation (9.4b). Under this assumption, we then show that the conditions on λ_n in the theorem statement suffice to guarantee that properties (b) and (d) hold for the constructed pair $(\widehat{\theta}, \widehat{z})$. The remainder of the analysis, provided in the full-length version of this paper, is devoted to showing that under the specified growth conditions (A3), imposing incoherence and dependence assumptions on the *population version* of the Fisher information $Q^*(\theta^*)$ guarantees (with high probability) that analogous conditions hold for the sample quantities $Q_s(\theta^*)$. While it follows immediately from the law of large numbers that the empirical Fisher information $Q^n_{AA}(\theta^*)$ converges to the population version Q^*_{AA} for any *fixed* subset, the delicacy is that we require controlling this convergence over subsets of increasing size. Our analysis therefore requires the use of uniform laws of large numbers [44].

9.3 Primal-Dual Relations for ℓ_1-Regularized Logistic Regression

Basic convexity theory can be used to characterize the solutions of ℓ_1-regularized logistic regression. We assume in this section that θ_1 corresponds to the unregularized bias term, and omit the dependence on sample size n in the notation. The objective is to compute

$$\begin{aligned}\operatorname*{argmin}_{\theta \in \mathbb{R}^p} \; \mathcal{L}(\theta, \lambda) &= \operatorname*{argmin}_{\theta \in \mathbb{R}^p} \left\{ f(\theta; x) + \lambda \left(\|\theta_{\setminus 1}\|_1 - b \right) \right\} \\ &= \operatorname*{argmin}_{\theta \in \mathbb{R}^p} \left\{ f(\theta; x) + \lambda \|\theta_{\setminus 1}\|_1 \right\} \end{aligned} \tag{9.9}$$

The function $\mathcal{L}(\theta, \lambda)$ is the Lagrangian function for the problem of minimizing $f(\theta; x)$ subject to $\|\theta_{\setminus 1}\|_1 \leq b$ for some b. The dual function is $h(\lambda) = \inf_\theta \mathcal{L}(\theta, \lambda)$.

If $p \leq n$ then $f(\theta; x)$ is a strictly convex function of θ. Since the ℓ_1-norm is convex, it follows that $\mathcal{L}(\theta, \lambda)$ is convex in θ and strictly convex in θ for $p \leq n$. Therefore the set of solutions to (9.9) is convex. If $\hat{\theta}$ and $\hat{\theta}'$ are two solutions, then by convexity $\hat{\theta} + \rho(\hat{\theta}' - \hat{\theta})$ is also a solution for any $\rho \in [0, 1]$. Since the solutions minimize $f(\theta; x)$ subject to $\|\theta_{\setminus 1}\|_1 \leq b$, the value of $f(\hat{\theta} + \rho(\hat{\theta}' - \hat{\theta}))$ is independent of ρ, and $\nabla_\theta f(\hat{\theta}; x)$ is independent of the particular solution $\hat{\theta}$. These facts are summarized below.

Lemma 1. *If $p \leq n$ then a unique solution to* (9.9) *exists. If $p \geq n$ then the set of solutions is convex, with the value of $\nabla_\theta f(\hat{\theta}; x)$ constant across all solutions. In particular, if $p \geq n$ and $|\nabla_{\theta_t} f(\hat{\theta}; x)| < \lambda$ for some solution $\hat{\theta}$, then $\hat{\theta}_t = 0$ for all solutions.*

The subgradient $\partial\|\theta_{\setminus 1}\|_1 \subset \mathbb{R}^p$ is the collection of all vectors z satisfying $|z_t| \leq 1$ and

$$z_t = \begin{cases} 0 & \text{for } t = 1 \\ sign(\theta_t) & \text{if } \theta_t \neq 0. \end{cases}$$

Any optimum of (9.9) must satisfy

$$\partial_\theta \mathcal{L}(\hat{\theta}, \lambda) = \nabla_\theta f(\hat{\theta}; x) + \lambda z = 0 \tag{9.10}$$

for some $z \in \partial\|\theta_{\setminus 1}\|$. The analysis in the following sections shows that, with high probability, a primal-dual pair $(\hat{\theta}, \hat{z})$ can be constructed so that $|\hat{z}_t| < 1$ and therefore $\hat{\theta}_t = 0$ in case $\theta^*_t = 0$ in the true model θ^* from which the data are generated.

9.4 Constructing a Primal-Dual Pair

We now fix a variable X_s for the logistic regression, denoting the set of variables in its neighborhood by S. From the results of the previous section we observe that the ℓ_1-regularized regression recovers the sparsity pattern if and only if there exists a primal-dual solution pair $(\widehat{\theta}, \widehat{z})$ satisfying the zero-subgradient condition, and the conditions (a) $\widehat{\theta}_{S^c} = 0$; (b) $|\widehat{\theta}_t| > 0 \quad$ for all $t \in S$ and $\operatorname{sgn}(\widehat{\theta}_S) = \operatorname{sgn}(\theta^*{}_S)$; (c) $\widehat{z}_S = \operatorname{sgn}(\theta^*_S)$; and (d) $\|\widehat{z}_{S^c}\|_\infty < 1$.

Our proof proceeds by showing the existence (with high probability) of a primal-dual pair $(\widehat{\theta}, \widehat{z})$ that satisfy these conditions. We begin by setting $\widehat{\theta}_{S^c} = 0$, so that (a) holds, and also setting $\widehat{z}_S = \operatorname{sgn}(\widehat{\theta}_S)$, so that (c) holds. We first establish a consistency result when incoherence conditions are imposed on the sample Fisher information Q^n. The remaining analysis, deferred to the full-length version, establishes that the incoherence assumption (A2) on the population version ensures that the sample version also obeys the property with probability converging to one exponentially fast.

Theorem 2. *Suppose that*

$$\|Q^n_{S^cS}(Q^n_{SS})^{-1}\|_\infty \quad \leq \quad 1-\epsilon \tag{9.11}$$

for some $\epsilon \in (0,1]$. *Assume that* $\lambda_n \to 0$ *is chosen that* $\lambda_n^2 n - \log(p) \to +\infty$ *and* $\lambda_n d \to 0$. *Then* $\mathbb{P}\left(\hat{\mathrm{N}}(s) = \mathrm{N}(s)\right) = 1 - O(\exp(-cn^\gamma))$ *for some* $\gamma > 0$.

Proof. Let us introduce the notation

$$W^n \quad := \quad \frac{1}{n}\sum_{i=1}^n z^{(i,s)}\left(x_s^{(i)} - \frac{\exp(\theta^{*T} z^{(i,s)})}{1+\exp(\theta^{*T} z^{(i,s)})}\right)$$

Substituting into the subgradient optimality condition (9.10) yields the equivalent condition

$$\nabla f(\widehat{\theta}; x) - \nabla f(\theta; x) - W^n + \lambda_n \widehat{z} \quad = \quad 0. \tag{9.12}$$

By a Taylor series expansion, this condition can be re-written as

$$\nabla^2 f(\theta^*; x)\ [\widehat{\theta} - \theta^*] \quad = \quad W^n - \lambda_n \widehat{z} + R^n, \tag{9.13}$$

where the remainder R^n is a term of order $\|R^n\|_2 = O(\|\widehat{\theta} - \theta^*\|^2)$.

Using our shorthand $Q^n = \nabla^2_\theta f(\theta^*; x)$, we write the zero-subgradient condition (9.13) in block form as:

$$Q^n_{S^cS}\ [\hat{\theta}^{s,\lambda}_S - \theta^*_S] \quad = \quad W^n_{S^c} - \lambda_n \widehat{z}_{S^c} + R^n_{S^c}, \tag{9.14a}$$

$$Q^n_{SS}\ [\hat{\theta}^{s,\lambda}_S - \theta^*_S] \quad = \quad W^n_S - \lambda_n \widehat{z}_S + R^n_S. \tag{9.14b}$$

It can be shown that the matrix Q^n_{SS} is invertible w.p. one, so that these conditions can be rewritten as

$$Q^n_{S^cS}\ (Q^n_{SS})^{-1}\ [W^n_S - \lambda_n \widehat{z}_S + R^n_S] \quad = \quad W^n_{S^c} - \lambda_n \widehat{z}_{S^c} + R^n_{S^c}. \tag{9.15}$$

Re-arranging yields the condition

$$Q^n_{S^cS}\,(Q^n_{SS})^{-1}\,[W^n_S - R^n_S] - [W^n_{S^c} - R^n_{S^c}] + \lambda_n Q^n_{S^cS}\,(Q^n_{SS})^{-1}\widehat{z}_S \quad = \quad \lambda_n \widehat{z}_{S^c} \tag{9.16}$$

Analysis of condition (d): We now demonstrate that $\|\widehat{z}_{S^c}\|_\infty < 1$. Using triangle inequality and the sample incoherence bound (9.11) we have that

$$\|\widehat{z}_{S^c}\|_\infty \quad \leq \quad \frac{(2-\epsilon)}{\lambda_n}\ [\|W^n\|_\infty + \|R^n\|_\infty] + (1-\epsilon) \tag{9.17}$$

We complete the proof that $\|\widehat{z}_{S^c}\|_\infty < 1$ with the following two lemmas, proved in [62].

Lemma 2. *If* $n\lambda_n^2 - \log(p) \to +\infty$*, then*

$$\mathbb{P}\left(\frac{2-\epsilon}{\lambda_n}\|W^n\|_\infty \geq \frac{\epsilon}{4}\right) \quad \to \quad 0 \tag{9.18}$$

at rate $O(\exp\left(-n\lambda_n^2 + \log(p)\right))$.

Lemma 3. *If* $n\lambda_n^2 - \log(p) \to +\infty$ *and* $d_{\max}\lambda_n \to 0$*, then we have*

$$\mathbb{P}\left(\frac{2-\epsilon}{\lambda_n}\|R^n\|_\infty \geq \frac{\epsilon}{4}\right) \quad \to \quad 0 \tag{9.19}$$

at rate $O(\exp\left(-n\lambda_n^2 + \log(p)\right))$.

We apply these two lemmas to the bound (9.17) to obtain that with probability converging to one at rate $O(\exp\left\{\exp\left(n\lambda_n^2 - \log(p)\right\}\right)$, we have

$$\|\widehat{z}_{S^c}\|_\infty \quad \leq \quad \frac{\epsilon}{4} + \frac{\epsilon}{4} + (1-\epsilon) \ = \ 1 - \frac{\epsilon}{2}.$$

Analysis of condition (b): We next show that condition (b) can be satisfied, so that $\mathrm{sgn}(\widehat{\theta}_S) = \mathrm{sgn}(\theta^*{}_S)$. Define $\rho_n := \min_{i\in S}|\theta^*_S|$. From equation (9.14b), we have

$$\hat{\theta}^{s,\lambda}_S \quad = \quad \theta^*_S - (Q^n_{SS})^{-1}\ [W_S - \lambda_n\widehat{z}_S + R_S]\,. \tag{9.20}$$

Therefore, in order to establish that $|\hat{\theta}^{s,\lambda}_i| > 0$ for all $i \in S$, and moreover that $sign(\hat{\theta}^{s,\lambda}_S) = sign(\theta^*_S)$, it suffices to show that

$$\left\|(Q^n_{SS})^{-1}\ [W_S - \lambda_n\widehat{z}_S + R_S]\right\|_\infty \quad \leq \quad \frac{\rho_n}{2}.$$

Using our eigenvalue bounds, we have

$$\begin{aligned}
\left\|(Q^n_{SS})^{-1}\ [W_S - \lambda_n\widehat{z}_S + R_S]\right\|_\infty \quad &\leq \quad \|(Q^n_{SS})^{-1}\|_\infty\ [\|W_S\|_\infty + \lambda_n + \|R_S\|_\infty] \\
&\leq \quad \sqrt{d}\ \|(Q^n_{SS})^{-1}\|_2\,[\|W_S\|_\infty + \lambda_n + \|R_S\|_\infty] \\
&\leq \quad \frac{\sqrt{d}}{C_{min}}\,[\|W_S\|_\infty + \lambda_n + \|R_S\|_\infty]\,.
\end{aligned}$$

In fact, the righthand side tends to zero from our earlier results on W and R, and the assumption that $\lambda_n d \to 0$. Together with the exponential rates of convergence established by the stated lemmas, this completes the proof of the result.

9.5 Experimental Results

We briefly describe some experimental results that demonstrate the practical viability and performance of our proposed method. We generated random Ising models (9.1) using the following procedure: for a given graph size p and maximum degree $d_{\max}$, we started with a graph with disconnected cliques of size less than or equal to ten, and for each node, removed edges randomly until the sparsity condition (degree less than $d_{\max}$) was satisfied. For all edges (s,t) present in the resulting random graph, we chose the edge weight $\theta_{st} \sim \mathcal{U}[-3,3]$. We drew n i.i.d. samples from the resulting random Ising model by exact methods. We implemented the ℓ_1-regularized logistic regression by setting the ℓ_1 penalty as $\lambda_n = \mathcal{O}((\log p)^3 \sqrt{n})$, and solved the convex program using a customized primal-dual algorithm. In each case, we evaluate a given method in terms of its average *precision* (one minus the fraction of falsely included edges), and its *recall* (one minus the fraction of falsely excluded edges). Figure 9.1 shows results for the case of constant degrees ($d_{\max} \leq 4$), and graph sizes $p \in \{100, 200, 400\}$, for the AND method (respectively the OR) method, in which an edge (s,t) is included if and only if it is included in the local regressions at both node s *and* (respectively *or*) node t. Note that both the precision and recall tend to one as the number of samples n is increased.

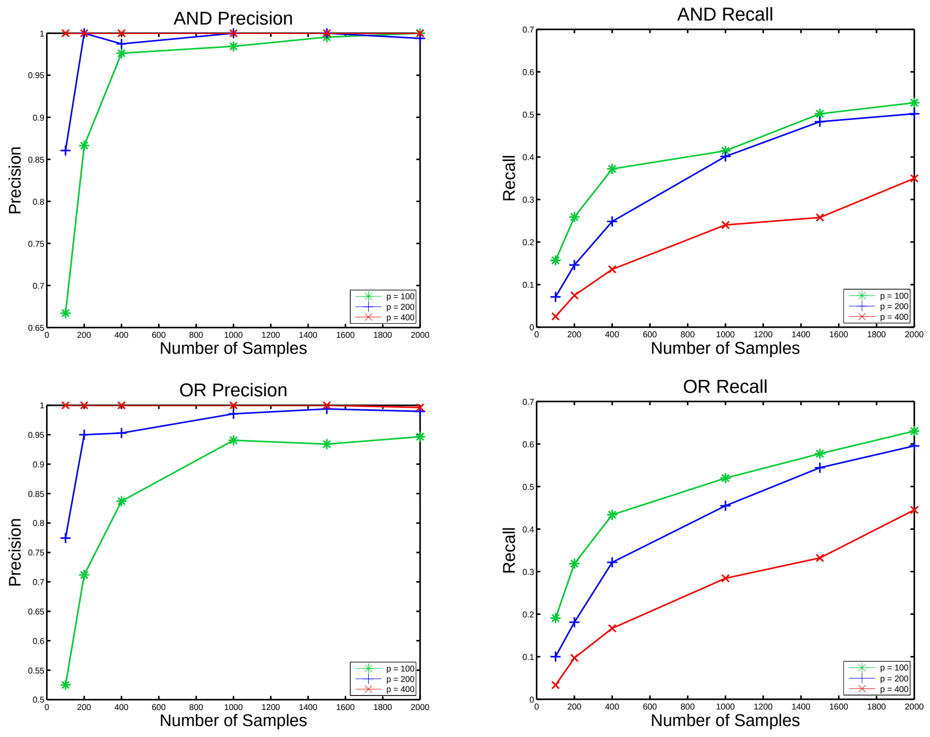


Figure 9.1: Precision/recall plots using the AND method (top), and the OR method (bottom). Each panel shows precision/recall versus n, for graph sizes $p \in \{100, 200, 400\}$.

Part III

Feature Estimation

Chapter 10

Features from data

The graphical models in previous chapters all belonged to a parametric family; in particular, the exponential family of distributions $p(X) \propto \exp(\theta^\top \phi(X))$. Given the feature functions $\phi(X)$, the structure and weights θ were then learned from data. It is typically a domain expert who specifies these feature functions; she either hand-designs them, or uses standard functions such as Ising, Potts, or indicator functions for discrete-valued models. The prediction accuracy of these models are then typically highly dependent on these expert-specified features; using the "raw" observations as features performs poorly [55]. It is thus of interest to develop techniques that can perform this feature estimation task in a data-driven fashion. Over the next two chapters, we propose techniques to (a) estimate feature functions from data, given the structure; and (b) for a restricted class of models, estimate the structure as well as feature functions simultaneously.

For the task of feature estimation given the structure, we are motivated in particular by structured prediction. Prediction tasks with known input and response variables, motivate discriminative models, which model the conditional distribution of the response given the input. In a structured prediction task, the response is multi-dimensional, with some inherent graphical structure, such as a linear chain for label sequences. It can be thought of as a multi-class problem with a large number of class labels, typically exponential in the number of variables, where for efficient estimation, the structure of the set of labels must be taken into account. For such a structured response $Y = (Y_1, \ldots, Y_T)$ then, instead of treating each Y_t as a separate prediction problem, it is important to estimate the model jointly across $\{Y_t\}$. A commonly used graphical model approach to this is conditional random fields, CRFs [33], which model the conditional distribution of the structured response Y by a Markov random field over the response variables $Y = (Y_1, \ldots, Y_T)$, globally conditioned on observation X. We extend CRFs to a new class of models, additive conditional random fields (aCRFs), which allow efficient estimation of the feature functions from data given the structure. In particular, we propose two feature-estimation procedures. One is a "dynamic" variant of boosting we call aDyBoost, which performs functional gradient descent in a smooth Hilbert space of functions. The second, called "dynamic backfitting," performs functional Gauss-Newton descent. The new methods give a flexible set of tools for nonparametric graphical models that complements those considered previously [34, 1, 53].

In Chapters 8,9, we used a technique based on ℓ_1 regularization to estimate the structure of a graphical model, given the feature functions; we focused in particular on models with Ising and indicator feature functions. We now propose a class of models, sparse additive models (SpAM) which allow us to do both structure and feature estimation simultaneously. Consider a linear regression model; $Y_i = X_i^T \beta + \epsilon_i$, for $i = 1, \ldots, n$; where Y_i is a real-valued response, X_i is a p-dimensional predictor and ϵ_i is a mean zero error term. For high-dimensional problems, where the number of predictors p is very large, for both statistical and computational performance reasons, it is necessary to first estimate the relevant set of predictors (predictor selection). Substantial progress has been made recently on this problem; in particular with the lasso [50]. The lasso estimator $\hat{\beta}$ minimizes the ℓ_1-penalized sums of squares

$$\sum_i (Y_i - X_i^T \beta) + \lambda \sum_{j=1}^{p} |\beta_j| \tag{10.1}$$

with the ℓ_1 penalty $\|\beta\|_1$ encouraging sparse solutions, where many components $\hat{\beta}_j$ are zero. In other words, the lasso performs predictor selection as well as parameter estimation simultaneously. The good empirical success of this estimator has been recently backed up by results confirming that it has strong theoretical properties; see [21, 72, 39, 56]. These models have a strong bias however – the features are linear. [25] thus introduced the class of additive models of the form

$$Y_i = \sum_{j=1}^{p} m_j(X_{ij}) + \epsilon_i \tag{10.2}$$

which extend linear models non-parametrically, but are still easy to fit and interpretable; in particular, an additive model can be estimated using a coordinate descent Gauss-Seidel procedure called backfitting (described in the next section). An extension of the additive model is the functional ANOVA model

$$Y_i = \sum_{1 \le j \le p} m_j(X_{ij}) + \sum_{j<k} m_{j,k}(X_{ij}, X_{ik}) + \sum_{j<k<\ell} m_{j,k,\ell}(X_{ij}, X_{ik}, X_{i\ell}) + \cdots + \epsilon_i \tag{10.3}$$

which allows interactions among the variables. Unfortunately, as was the case with linear models, additive models only have good statistical and computational behavior when the number of variables p is not large relative to the sample size n.

We introduce sparse additive models (SpAM), a class of models which allow predictor selection as well as component function estimation simultaneously; just as the lasso performs simultaneous predictor selection and weight estimation with (parametric) linear features. The underlying model is the same as in (10.2), but constraints are placed on the component functions $\{m_j\}_{1 \leq j \leq p}$ to simultaneously encourage smoothness of each component and sparsity across components. It also naturally extends to classification problems using generalized additive models.

The next two chapters develop and analyze aCRFs and SpAM; at the heart of these methods are the concepts of smoothing and additive and generalized additive models, which we briefly review in the next section.

10.1 Smoothing and Additive Models

A nonparametric regression model is given by,

$$Y = m(X) + \epsilon, \;\; \mathbb{E}(\epsilon) = 0 \tag{10.4}$$

here Y is a real-valued response, X is a p-dimensional predictor; and $m(x) = \mathbb{E}(Y|X = x)$ is the *regression* function; assumed to lie in a smooth function space such as a Sobolev space. The task in non-parametric regression is to estimate this regression function $m(X)$ given n observations $\{(X_i, Y_i),\, i = 1, \ldots, n\}$. Such a non-parametric estimate $\hat{m}(x)$ of the regression function is referred to as a smoother, since it "smooths" the noisy function values at given input points into a function over the entire domain. Common smoothers include basic kernel regression, local linear and local polynomial smoothing, binning, scatterplot smoothing, or even techniques such as wavelet regression.

We briefly describe cubic splines and kernel smoothers below. A cubic spline is the solution of the following penalized likelihood problem,

$$\min_{m \in \mathcal{S}} \sum_{i=1}^{n} (Y_i - m(X_i))^2 + \lambda \int (m''(x))^2 dx \tag{10.5}$$

where $\mathcal{S} = \{m : \int (m''(x))^2 dx < \infty\}$ is the sobolev space of order two. The penalty $J(m) = \int (m''(x))^2 dx$ penalizes rough functions; in particular the solution of the above optimization problem is a cubic polynomial that interpolates the given data points.

A kernel smoother is a local weighted average estimator, defined as

$$\hat{m}(x) = \frac{\sum_{i=1}^n K(\frac{x-X_i}{h}) \quad Y_i}{\sum_{i=1}^n K(\frac{x-X_i}{h})} \tag{10.6}$$

where K is a kernel function, and gives a large weight to data points X_i close to x and a small weight to points which are farther.

A smoother which when evaluated at any x is a linear combination of the training responses, $\hat{m}(x) = \sum_i s_i Y_i = \mathcal{S}_x Y$, is referred to as a linear smoother. The linear combination coefficients depend on the evaluation point x. Both cubic splines and kernel estimators are linear smoothers.

Since non-parametric smoothing procedures become challenging when X is very high dimensional, [25] introduced the class of additive models,

$$Y_i = \sum_{j=1}^p m_j(X_{ij}) + \epsilon_i \tag{10.7}$$

To see how we can fit such a model, consider the population objective function for an additive model,

$$\frac{1}{2}\mathbb{E}\left(Y - \sum_{j=1}^p m_j(X_j)\right)^2 \tag{10.8}$$

Let $R_j = Y - \sum_{k\neq j} m_k(X_k)$ be the jth residual. Then the stationary condition for minimizing the objective as a function of m_j, holding the other components m_k fixed for $k \neq j$, is simply

$$0 = \mathbb{E}\left(m_j(X_j) + \sum_{k\neq j} m_k(X_k) - Y \,\middle|\, X_j\right) \tag{10.9}$$

$$m_j(X_j) = \mathbb{E}\left(R_j \mid X_j\right) \tag{10.10}$$

The backfitting procedure replaces this population expectation by a sample version that uses a smoother (or smoothing matrix in the case of a linear smoother) $\mathcal{S}_j$: $\hat{m}_j(X_j) = \mathcal{S}_j R_j$. It can be summarized as follows:

```
Iterate until convergence:
    For each $j = 1, \dots, p$:
        Compute the residual: $R_j = Y - \sum_{k\neq j} m_k(X_k)$;
        Estimate the projection $P_j = \mathbb{E}[R_j \mid X_j]$ by smoothing: $\hat{P}_j = \mathcal{S}_j R_j$;
```

```
            Update the $j-$th component: $m_j \leftarrow \hat{P}_j$.
```

In the case of the additive logistic regression, the model takes the form

$$p(Y=1|X;\beta) = \exp\left(\sum_{j=1}^{p} m_j(X_j)\right)/1+\exp\left(\sum_{j=1}^{p} m_j(X_j)\right) \tag{10.11}$$

where the functions $\{m_j\}$ take the place of the linear combinations $\{\beta_j X_j\}$ in logistic regression. In this case, when $Y \in \{0,1\}$, the population log-likelihood is

$$\ell(m) = \mathbb{E}\left[Ym(X) - \log\left(1+\exp m(X)\right)\right] \tag{10.12}$$

The stationary condition for component function m_j is

$$\mathbb{E}\left(p - Y \mid X_j\right) = 0 \tag{10.13}$$

However, this condition is nonlinear in m, and so we linearize the gradient of the log-likelihood around a current estimate m_0. This yields the linearized condition

$$\mathbb{E}\left[w(X)(m(X) - Z) \mid X_j\right] = 0 \tag{10.14}$$

where Z is a transformed response for the current estimate m_0:

$$Z_i = m_0(X_i) + \frac{Y_i - p(X_i;m_0)}{p(X_i;m_0)(1-p(X_i;m_0))} \tag{10.15}$$

and the weights are $w(X_i) = p(X_i;m_0)(1-p(X_i;m_0))$. The weighted smooth is given by

$$\hat{P}_j = \frac{\mathcal{S}_j(wR_j)}{\mathcal{S}_j w}. \tag{10.16}$$

which is a backfitting of (Z, X) with weights w. This yields a local scoring algorithm which runs the backfitting procedure within Newton's method.

```
Iterate until convergence:
    Evaluate as above the transformed response and weight $(Z, W)$ around
    current estimate.
    Iterate until convergence:
        For each $j = 1, \dots, p$:
            Compute the residual: $R_j = Z - \sum_{k \neq j} m_k(X_k)$;
            Estimate the weighted projection $P_j = \frac{\mathbb{E}[wR_j \mid X_j]}{\mathbb{E}[w|X_j]}$ by smooth-
            ing: $\hat{P}_j = \frac{\mathcal{S}_j(wR_j)}{\mathcal{S}_j w}$.
```

Update the $j-$th component: $m_j \leftarrow \hat{P}_j$

Another technique commonly used to fit additive logistic regression models is AdaBoost and its variants; see [20].

The smoothing matrices $\mathcal{S}_j$ can be derived from any of a wide range of non-parametric smoothers. The backfitting procedure can be viewed as a Gauss-Seidel coordinate descent algorithm. It can be seen as trading off convergence speed for ease of implementation and scalability to high dimensions. We refer to [25] for an extensive introduction.

An extension of the additive model is the functional ANOVA model where the functions take the form

$$m(X) = \sum_{1 \leq j \leq p} m_j(X_{ij}) + \sum_{j<k} m_{j,k}(X_{ij}, X_{ik}) + \sum_{j<k<\ell} m_{j,k,\ell}(X_{ij}, X_{ik}, X_{i\ell}) + \cdots \tag{10.17}$$

Such an extension is naturally applicable to the aCRF models that we discuss next.

Chapter 11

Additive Conditional Random Fields

11.1 Introduction

In this chapter, we propose a new class of models, additive conditional random fields (aCRFs), which allow efficient estimation of the feature functions from data given the structure. The motivating task is structured prediction. In a structured prediction task, the response is multi-dimensional, with some inherent graphical structure, such as a linear chain for label sequences. It can be thought of as a multi-class problem with a large number of class labels, typically exponential in the number of variables, where for efficient estimation, the structure of the set of labels must be taken into account. Problems such as speech recognition, image denoising, object recognition, natural language parsing, information extraction, handwriting recognition, gene prediction, machine translation and many others can be naturally cast as structured prediction problems. A small sample of recent work in this direction includes [15, 43, 33, 37, 32, 46, 49, 1, 53].

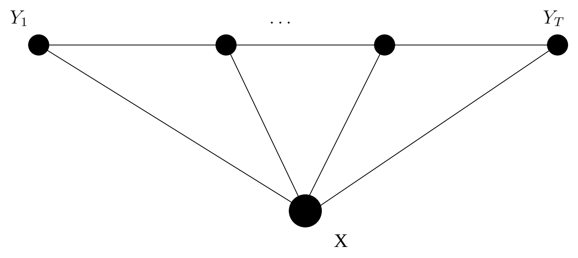
A commonly used graphical model approach to structured prediction is conditional random fields, CRFs [33], which model the conditional distribution of the structured response Y by a Markov random field over the response variables $Y = (Y_1, \ldots, Y_T)$, globally conditioned on observation X. We extend CRFs to a new class of models, additive conditional random fields (aCRFs), which as stated previously, allow efficient estimation of feature functions from data. In particular, we propose two feature estimation procedures. One is a "dynamic" variant of boosting we call aDyBoost, which performs functional gradient descent in a smooth Hilbert space. The second is an extension of additive model backfitting, which we call "dynamic backfitting," and which performs functional

Gauss-Newton descent. These procedures allow the use of arbitrary smoothing techniques, and are easy to implement. While we focus here on sequence models, for which the underlying graph is a chain, the methods generalize naturally to general (conditional) graphical models. The new methods give a flexible set of tools for nonparametric graphical models that complements those considered previously [34, 1, 53].

11.2 Additive Conditional Random Fields

Let $X = \{X_1, \ldots, X_T\}$ be a sequence of observation variables, and $Y = \{Y_1, \ldots, Y_T\}$ be the corresponding sequence of label variables. A conditional random field models the conditional distribution of the label sequence Y given the observation sequence X by an undirected graphical model over the label variables. The feature functions over cliques of the label graph are allowed to depend on the entire observation sequence X. In what follows, we start off with a canonical CRF formulation and then extend it to our aCRF formulation.

Consider a CRF where the graph over the labels is a chain as in HMMs, so that the neighbors of Y_t are Y_{t-1} and Y_{t+1}.



$$p(\mathbf{Y} \,|\, \mathbf{X}) \propto \exp \sum_t \sum_j w_{j,t} g_{j,t}(Y_t, \mathbf{X}) + \sum_t \sum_k w_{k,t} f_{k,t}(Y_{t-1}, Y_t, \mathbf{X}) \quad (11.1)$$

where $\{g_{j,t}, f_{j,t}\}$ are the features, and $\{w_{j,t}, w_{k,t}\}$ are the corresponding weights. Assuming the features are time-independent, and absorbing the weights into the features, we can write

$$p(\mathbf{Y} \,|\, \mathbf{X}) \propto \exp \sum_t \sum_j g_j(Y_t, h_{j,t}(X)) + \sum_t \sum_k f_k(Y_{t-1}, Y_t, h_{k,t}(X)) \quad (11.2)$$

where $h_{l,t}(X)$ is a feature-specific history window of the observation sequence, e.g. (X_t, X_{t-1}). Consider further a "tensor product" assumption, namely that the feature functions can be written as a product of functions of Y and functions of X as

$$p(\mathbf{Y}\,|\,\mathbf{X}) \propto \exp \sum_t \sum_j w_j\, q_s(Y_t) g_j(h_{j,t}(X)) + \sum_t \sum_k w_k\, q_p(Y_{t-1}, Y_t) f_k(h_{k,t}(X)) \tag{11.3}$$

Consider now an ANOVA type additive expansion of the observation feature functions. Let $m_k(X_{t,k})$, $m_{p,kk'}(X_{t-1,k}, X_{t,k'})$, $m_{s,kk'}(X_{t,k}, X_{t,k'})$ be the first and second-order feature functions, where k ranges over the dimension of covariates X_t.

$$p(\mathbf{Y}\,|\,\mathbf{X}) \propto \exp \sum_t \sum_k q_s(Y_t) m_k(X_{t,k}) + \sum_t \sum_{k,k'} q_s(Y_t) m_{s,kk'}(X_{t,k}, X_{t,k'}) + \sum_t \sum_{k,k'} q_p(Y_{t-1}, Y_t) m_{p,kk'}(X_{t-1,k}, X_{t,k'}) \tag{11.4}$$

This, then, is our aCRF formulation. In CRFs, the functional form of the features is fixed and assumed, and only multiplicative weights are learned. In aCRFs, we learn the features themselves from data. The additive nature of the observation feature functions $m(X_t) = \sum_k m_k(X_{tk})$ allows us to finesse the curse of dimensionality and retain applicability in high dimensions. Note that since the label domain is categorical, learning the label functions q reduces to parametric learning of weights q_{ij}, which can be seen by letting $q(y, y') = \sum_{ij} q_{ij} I_{ij}(y, y')$.

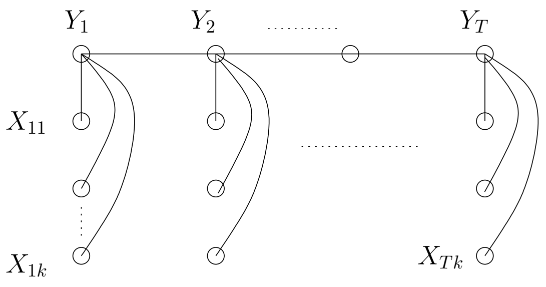


Figure 11.1: First order aCRF

Figure (11.1) shows a first order chain aCRF, with observation functions $m_k(X_{t,k})$; we now formalize the above descriptions for this first order chain aCRF; the details generalize to higher order aCRFs.

For $j = 1, \ldots, p$ ranging over the dimension of observations; let $\{X_{tj}, t = 1, \ldots, T\}$, be random variables with a common domain $\mathcal{X}_j \subseteq \mathbb{R}$. Let $\mathcal{H}_{tj}$ be smooth Hilbert spaces of measurable functions $m_{tj}(X_{tj})$, with $\mathbb{E}(m_{tj}(X_{tj})) = 0$, and $\mathbb{E}(m_{tj}^2(X_{tj})) < \infty$, and inner product $\left\langle m_{tj}, m'_{tj} \right\rangle = \mathbb{E}(m_{tj}(X_{tj}) m'_{tj}(X_{tj}))$. Let $\mathcal{H}_{t+} = \oplus_{j=1}^{p} \mathcal{H}_{tj}$ denote the Hilbert space of functions of $(\mathbf{X}_{t,1}, \ldots, \mathbf{X}_{t,p})$ with an additive form: $m_t(X_t) = \sum_{j=1}^{p} m_{tj}(X_{tj})$. Denote $\mathcal{H}_j = \cap_{t=1}^{T} \mathcal{H}_{tj}$, and $\mathcal{H}_+ = \cap_{t=1}^{T} \mathcal{H}_{t+} = \oplus_{j=1}^{p} \mathcal{H}_j$. Then, the first order aCRF is given by

$$p(\mathbf{Y} \mid \mathbf{X}) \propto \exp \sum_t \alpha(Y_{t-1}, Y_t) + \sum_t q(Y_{t-1}, Y_t) m(X_t) \tag{11.5}$$

where $m(X_t) = \sum_j m_j(X_{tj})$ lies in $\mathcal{H}_+$.

We remark that the above formulation is primarily intended for the case where the inputs X_j are real valued, or categorical with many values. In the case where the X_j are binary, the standard linear CRF is already effectively nonparametric.

11.3 Backfitting and Boosting

Let $X = (X_1, \ldots, X_T)$ be the observation sequence, where each $X_t = (X_{t1}, \ldots, X_{tp}) \in \mathbb{R}^p$ and $Y = (Y_1, \ldots, Y_T)$ be the label sequence, where each $Y_t \in \mathcal{C}$, for a categorical set $\mathcal{C}$. Let $m_t(X_t) = \sum_{j=1}^{p} m_{tj}(X_{tj})$ be the observation feature function at time t. We assume that the feature functions are time-independent, so that $m_t = m$, $m_{tj} = m_j$, $t = 1, \ldots, T$.

Let $\ell(m, X, Y)$ denote the aCRF log-likelihood functional at a sequence instance (X, Y). Note that the aCRF log-likelihood uses the same observation feature function m at T time steps, and thus can also be written as $\ell(m, \ldots, m, X, Y)$ where m is used as the first T arguments. We present a canonical example below for illustrating this form of the aCRF log-likelihood

$$l(m, X, Y) = \sum_t \alpha_{Y_{t-1}, Y_t} m(X_t) - \log Z(m, X) \tag{11.6}$$

where $Z(m, X) = \log \sum_Y \exp \left(\sum_t \alpha_{Y_{t-1}, Y_t} m(X_t) \right)$ We require an estimate of $\{m_j\}$ from n training sequences $\{X^{(1)}, \ldots, X^{(n)}\}$. We propose two estimation procedures for aCRF. One is a dynamic variant of boosting we call aDyBoost, and the second is a backfitting procedure we call "dynamic backfitting". In the next section, we derive these as the sample analogues of gradient descent and Gauss-Newton descent of the population aCRF log-likelihood over an additive Hilbert space of functions.

Figure 11.2 details the aDyBoost algorithm for a general aCRF. We illustrate the computation of the gradient with the following example, where the model is simplified for clarity:

$$p(\mathbf{Y} \mid \mathbf{X}) \propto \exp\left(\sum_t \alpha_{Y_{t-1},Y_t} + Y_t\left(\sum_j m_j(X_{tj})\right)\right) \tag{11.7}$$

The (functional) gradient contribution at time step t for this model is then

$$V_t = \frac{\partial \ell}{\partial m_t} = Y_t - P(Y_t = 1). \tag{11.8}$$

As [20] showed, additive logistic regression models can be fit by Adaboost, which performs functional gradient descent. Here, we estimate the feature functions shared across time steps by a dynamic variant of boosting. The gradient is computed by a double smoothing procedure. At each time step, the contribution for that time step is computed by smoothing the functional gradient, then the kernel average of the contributions from all time steps is computed to obtain the gradient.

Figure (11.3) describe a “backfitting” procedure that uses second-order information. Called dynamic backfitting, it is an extension of the local scoring based backfitting procedure for generalized additive models. The Gauss-Newton descent direction–as with aDyBoost–is computed by a two-fold smoothing procedure. Weighted contributions from each time step is computed by smoothing the weighted gradients, and finally the contributions from all time steps are smoothed by kernel averaging to yield the descent direction.

11.4 Derivation of aDyBoost and Dynamic Backfitting

Let $\ell : \otimes_{t=1}^{T} \mathcal{H}_{t+} \otimes_{j=1}^{p} \mathcal{X}_j^T \times \mathcal{Y} \rightarrow \mathbb{R}$ be the aCRF log-likelihood whose first T arguments are functions in $\mathcal{H}_{t+}$, for $t = 1, \dots, T$.

Now consider the expected log-likelihood $\mathbb{E}\ell : \mathcal{H}_+ \rightarrow \mathbb{R}$,

$$\mathbb{E}\ell(m) = \mathbb{E}[\ell(m, X, Y)] = \int \ell(m, \dots, m, X, Y)\, p(X, Y)\, dX dY \tag{11.9}$$

where $X = \{X_{tj}, t = 1, \dots, T,\ j = 1, \dots, p\}$, and $p(X,Y)$ is the population density over (X,Y). Let $Z = (X,Y)$. For $\hat{m} = m + \epsilon \eta_j$, where $\eta_j \in \mathcal{H}_j$,

$$\mathbb{E}\ell(\hat{m}) = \mathbb{E}\ell(m) + \epsilon \int \sum_t \frac{\partial \ell}{\partial m_t}\Big|_{m_t = m} \eta_j(X_{tj}) p(Z)\, dZ + O(\epsilon^2) \tag{11.10}$$

Thus the first variation of $\mathbb{E}\ell$ is given by

$$\delta\mathbb{E}\ell(m,\eta_j) = \sum_t \int \mathbb{E}\left[\frac{\partial\ell}{\partial m_t}\Big|_{m_t=m}\middle| X_{tj}=x_j\right] p(X_{tj}=x_j)\eta_j(x_j)\,dx_j$$

and the gradient $\frac{\partial\mathbb{E}\ell}{\partial m}$ is given by

$$\frac{\partial\mathbb{E}\ell}{\partial m}(x_j) = \sum_t \mathbb{E}\left[\frac{\partial\ell}{\partial m_t}\Big|_{m_t=m}\middle| X_{tj}=x_j\right] p(X_{tj}=x_j). \tag{11.11}$$

In the finite sample case, the conditional expectation of the gradient given X_{tj} at each time step is estimated by a smoother, while $p(X_{tj}=x_j)$ by any density estimator. Using the Nadaraya-Watson kernel density estimator yields a kernel averaging of the contributions from different steps. Thus, we get the dual-smoothing procedure of aDyBoost.

For the Gauss-Newton update, note that for m to be an optimum it is required that

$$\frac{\partial\ell}{\partial m} = \sum_t \mathbb{E}\left[\frac{\partial\ell}{\partial m_t}\Big|_{m_t=m}\middle| X_{tj}=x_j\right] p_{tj}(x_j) = 0 \tag{11.12}$$

A partial linearization around a current guess $m=m_0$, restricted to each time step, gives

$$0 = \sum_t p_{tj}(x_j)\mathbb{E}\left[\frac{\partial\ell}{\partial m_{t0}} + \frac{\partial^2 l}{\partial m_{t0}^2}\left(m(X_t)-m_0(X_t)\right)\middle| X_{tj}=x_j\right]$$

Letting $W_{t0} = -\frac{\partial^2\ell}{\partial m_{t0}^2}$, and $V_{t0} = m_0(X_t) + W_{t0}^{-1}\frac{\partial\ell}{\partial m_{t0}}$, the linearization becomes

$$0 = \sum_t p_{tj}(x_j)\mathbb{E}\left[W_{t0}\left(V_{t0} - \sum_{k\neq j} m_k(X_{tk}) - m_j(x_j)\right)\middle| X_{tj}=x_j\right]$$

We have thus derived the dynamic backfitting procedure, since

$$m_j(x_j) = \frac{\sum_t p_{tj}(x_j)\mathbb{E}\left[W_{t0}\left(V_{t0} - \sum_{k\neq j} m_k(X_{tk})\right)\middle| X_{tj}=x_j\right]}{\sum_t p_{tj}(x_j)\mathbb{E}\left[W_{t0}\middle| X_{tj}=x_j\right]} \tag{11.13}$$

11.5 Experiments

In this section, we present three sets of experiments. First, we provide a synthetic example to motivate the use of a nonparametric model for structured classification tasks. In the second experiment, we use synthetic data to demonstrate that our back fitting procedure can accurately recover a generative model from sequences sampled from the generative model. In the third set of experiments, we present results using speech data from the UCI KDD archive to contrast aCRFs with traditional CRFs on a more realistic data set.

11.5.1 Multi-modal observations

We sampled sequences from a hidden Markov model with two hidden states and single, continuous emission term. The transition dynamics of the Markov chain were biased towards self-transitions, specifically $p(Y_t = i \,|\, Y_{t-1} = j) = .75$ when $i = j$. The emissions were drawn from two component Gaussian mixture models as shown in Figure (11.4) (left). The key property of the emission model is that the components corresponding to different states are interleaved. This represents a pathological case for simple sequence models because the mode of the observations produced by each state falls inside of a mixture component corresponding to the other state.

We compared an aCRF to a parametric CRF having first-, second-, or third-order features of the observed variable. For example, the second-order CRF includes the features $\{f_1 = Y_t,\, f_2 = Y_tX_t,\, f_3 = Y_tX_t^2\}$ in addition to the usual features associated with state transitions. The additive CRF fit m using a kernel smoother with an Epanechnikov kernel, with a bandwidth $h = .25$. The results of fitting these models to a sequence of 500 training points are presented in Figure 11.4 (right). Specifically, the figure shows fitted functions and the values of $\beta^T f$ for the relevant features of the various CRFs as X_t is varied and Y_t is held fixed at 1. The CRFs with linear and quadratic features perform poorly, producing error rates of 30.46% and 30.54% respectively. The nonparametric aCRF and the CRF that includes third-order terms both perform well and produce error rates of 2.46% and 2.56% respectively.

11.5.2 Reconstructing known functions

In this experiment, we generated synthetic data from an aCRF using known functions. The goal is to recover an accurate approximation to the true functions used to generate data from a sequence sampled from the true model. The generative model contained two states and included features equivalent to a transition model of $p(Y_t = i \,|\, Y_{t-1} = j) = .75$ when $i = j$. To sample sequences from an aCRF, we generated a sequence of observations X, where in each time step X_t consisted of two covariates drawn from *Uniform*$(-2, 2)$. We then used these observed values in conjunction with the model containing known m_j to sample a label sequence consistent with the observation sequence. Figure 11.5 shows the resulting learned $\hat{m}_j$ when an aCRF is trained on a sequence of 5,000 time steps sampled from the generative model versus the true functions used in the generative model. There is a good correspondence between the true and estimated functions.

11.5.3 Speaker identification

Our third set of experiments tested the aCRF in a speaker identification task using speech data from the UCI KDD archive that were donated by [30]. The data are a collection of discrete utterances made by nine male Japanese speakers. A single utterance consists of the vowels *ae* spoken together, and each utterance is represented as a series of twelve cepstral coefficients. To generate longer sequences with transition dynamics between speakers, we defined a Markov chain over speakers and sampled from this chain to identify a speaker. The training set contained 30 utterances per speaker. A single transition between speakers in the Markov chain produces observations across multiple time steps—sampling 10 speakers from the Markov chain would produce a sequence of approximately 150 label/observation pairs where the label identifies the speaker at any given moment and each observation consists of 12 cepstral coefficients. Figure 11.6 presents results comparing an aCRF with a CRF built with first order features over the covariates in X as the length of the training sequence was varied. For these experiments, we reduced the classification problem described above to the binary case by remapping the labels Y_t to $Y_t' = Y_t \mod 2$. Each training sequence length was tested 50 times. On average, the nonparametric model produced a lower error rate on test data across all of the training lengths we considered.

```
*Input*: Data $(X^{(i)}, Y^{(i)})$.

*Initialize* $m_j = m^{(0)}j$, for $j = 1, \ldots, p$.

*Iterate* until convergence:

    For each dimension $j = 1, \ldots, p$:
        *Smoothing at each time step:*
        For each time step $t = 1, \ldots, T$
            Compute the gradient contribution of time $t$: $V_t = \frac{\partial \ell}{\partial m_t}$.
            Estimate the conditional expectation $G_{tj} = \mathbb{E}[V_t \mid X_{tj} = x_j]$
            by smoothing: $\hat{G}_{tj} = \mathcal{S}_{tj} V_t$.

        *Smoothing across time steps:*
        For each time step $t = 1, \ldots, T$
            Estimate the density $p(X_{tj} = x_j)$ at time $t$ by a smoothed
            kernel density estimate: $\hat{p}_{tj}(x_j) = \mathcal{K}_{tj}(x_j)$

        Compute the gradient for $j$ by adding weighted contributions from
        all time steps:
        $g_j(x_j) = \sum_{t=1}^{T} \hat{p}_{tj}(x_j) \hat{G}_{tj}$
        Descend along gradient: $m_j \leftarrow m_j - \alpha g_j$.

*Output*: Component functions $m_j$ and estimator $\hat{m}(X_t^{(i)}) = \sum_j m_j(X_{tj}^{(i)})$.
```

Figure 11.2: ADyBoost

```
*Input*: Data $(X^{(i)}, Y^{(i)})$.

*Initialize* $m_j = m^{(0)}j$, for $j = 1, \ldots, p$.

*Iterate* until convergence:

    For each time step $t = 1, \ldots, T$
        Compute the gradient contribution of time $t$: $V_t = \frac{\partial \ell}{\partial m_t}$.
        Compute the Hessian contribution of time $t$: $W_t = \frac{\partial^2 \ell}{\partial m_t^2}$

    For each dimension $j = 1, \ldots, p$:
        *Smoothing at each time step:*
        For each time step $t = 1, \ldots, T$
            Compute the weighted residual $R_{jt} = W_t(V_t - \sum_{k \neq j} m_{kt})$.
            Estimate the conditional expectation $G_{tj} = \mathbb{E}[R_{tj} \mid X_{tj} = x_j]$
            by smoothing: $\hat{G}_{tj} = \mathcal{S}_{tj} G_{tj}$.
            Estimate the expected weights $H_{tj} = \mathbb{E}[W_t \mid X_{tj} = x_j]$ by
            smoothing: $\hat{H}_{tj} = \mathcal{S}_{tj} H_{tj}$.

        *Smoothing across time steps:*
        For each time step $t = 1, \ldots, T$
            Estimate the density $P(X_{tj} = x_j)$ at time $t$ by a smoothed
            density estimate: $\hat{P}_{tj}(x_j) = \mathcal{K}_{tj}(x_j)$

        Compute the Gauss-Newton update for $j$ by adding weighted con-
        tributions from all time steps:
        $m_j(x_j) = \frac{\sum_{t=1}^T \hat{P}_{tj}(x_j)\hat{G}_{tj}}{\sum_{t=1}^T \hat{P}_{tj}(x_j)\hat{W}_{tj}}$.

*Output*: Component functions $m_j$ and estimator $\hat{m}(X_t^{(i)}) = \sum_j m_j(X_{tj}^{(i)})$.
```

Figure 11.3: DYNAMIC BACKFITTING

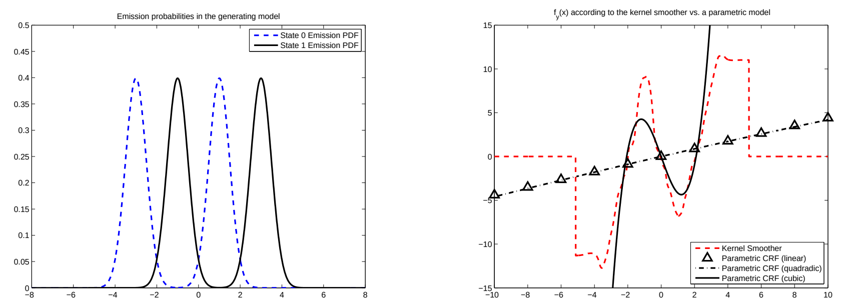


Figure 11.4: Experiment 1: interleaved mixture components (left) and fitted feature functions $\hat{m}_y$ (right) learned using a kernel smoother in an aCRF. The corresponding values of $\beta^T f$ from the relevant features in traditional CRFs that include first-, second-, and third-order terms are also shown.

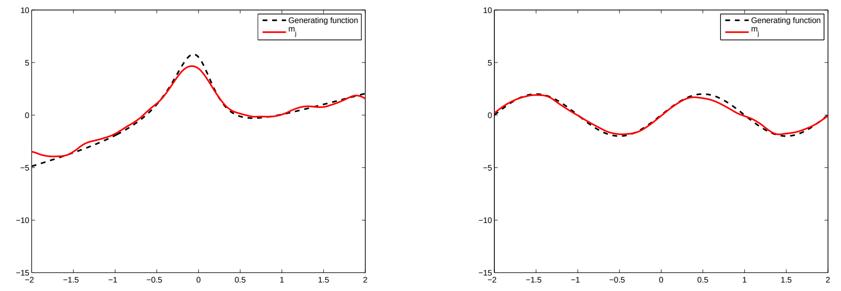


Figure 11.5: Experiment 2: m_1 (left) and m_2 (right) and the estimated functions $\hat{m}_1$ and $\hat{m}_2$.

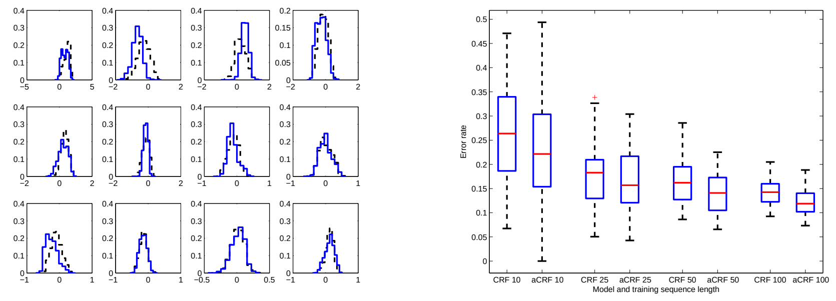


Figure 11.6: Experiment 3: (left) histograms comparing the distributions of the 12 covariates for the classes $Y_t = 0$ and $Y_t = 1$; (right) error rates comparing the traditional CRF and aCRF.

Chapter 12

SpAM: Sparse Additive Models

12.1 Introduction

In this chapter, we introduce sparse additive models (SpAM), a class of models which allow simultaneous structure and feature estimation. We restate the motivating discussion of Chapter 10 for continuity. Consider a linear regression model, $Y_i = X_i^T\beta + \epsilon_i$, for $i = 1, \ldots, n$; where Y_i is a real-valued response, X_i is a p-dimensional predictor and ϵ_i is a mean zero error term. For high-dimensional problems, where the number of predictors p is very large, for both statistical and computational performance reasons, it is necessary to first estimate the relevant set of predictors (predictor selection). Substantial progress has been made recently on this problem; in particular with the lasso [50]. The lasso estimator $\hat{\beta}$ minimizes the ℓ_1-penalized sums of squares

$$\sum_i (Y_i - X_i^T\beta) + \lambda \sum_{j=1}^{p} |\beta_j| \tag{12.1}$$

with the ℓ_1 penalty $\|\beta\|_1$ encouraging sparse solutions, where many components $\hat{\beta}_j$ are zero. In other words, the lasso performs predictor selection as well as parameter estimation simultaneously. The good empirical success of this estimator has been recently backed up by results confirming that it has strong theoretical properties; see [21, 72, 39, 56]. These models have a strong bias however – the features are linear. [25] thus introduced the class of additive models of the form

$$Y_i = \sum_{j=1}^{p} m_j(X_{ij}) + \epsilon_i \tag{12.2}$$

which extend linear models non-parametrically, but are still easy to fit and interpretable; in particular, an additive model can be estimated using a coordinate descent Gauss-Seidel procedure called backfitting (described in the next section). An extension of the additive model is the functional ANOVA model

$$Y_i = \sum_{1\le j\le p} m_j(X_{ij}) + \sum_{j<k} m_{j,k}(X_{ij}, X_{ik}) + \sum_{j<k<\ell} m_{j,k,\ell}(X_{ij}, X_{ik}, X_{i\ell}) + \cdots + \epsilon_i \tag{12.3}$$

which allows interactions among the variables. Unfortunately, as was the case with linear models, additive models only have good statistical and computational behavior when the number of variables p is not large relative to the sample size n.

We introduce sparse additive models (SpAM), a class of models which allow predictor selection as well as component function estimation simultaneously; just as the lasso performs simultaneous predictor selection and weight estimation with (parametric) linear features. The underlying model is the same as in (12.2), but constraints are placed on the component functions $\{m_j\}_{1\le j\le p}$ to simultaneously encourage smoothness of each component and sparsity across components. The SpAM estimation procedure we introduce allows the use of arbitrary nonparametric smoothing techniques, and in the case where the underlying component functions are linear, it reduces to the lasso. It naturally extends to classification problems using generalized additive models. Our main results are (i) the formulation of a convex optimization problem for estimating a sparse additive model, (ii) an efficient backfitting algorithm for constructing the estimator, (iii) simulations showing the estimator has excellent behavior on some simulated and real data, even when p is large, and (iv) a statistical analysis of the theoretical properties of the estimator that support its good empirical performance.

12.2 The SpAM Optimization Problem

In this section we describe the key idea underlying SpAM. We first present a population version of the procedure that intuitively suggests how sparsity is achieved. We then present an equivalent convex optimization problem. In the following section we derive a backfitting procedure for solving this optimization problem in the finite sample setting.

To motivate our approach, we first consider a formulation that scales each component function g_j by a scalar β_j, and then imposes an ℓ_1 constraint on $\beta = (\beta_1, \ldots, \beta_p)^T$. For $j \in \{1, \ldots, p\}$, let $\mathcal{H}_j$ denote the Hilbert space of measurable functions $f_j(x_j)$ of the single scalar variable x_j, such that $\mathbb{E}(f_j(X_j)) = 0$ and $\mathbb{E}(f_j(X_j)^2) < \infty$, furnished with the inner product

$$\langle f_j, f_j' \rangle = \mathbb{E}\left(f_j(X_j) f_j'(X_j)\right). \tag{12.4}$$

Let $\mathcal{H}^{\text{add}} = \mathcal{H}_1 + \mathcal{H}_2 + \ldots, \mathcal{H}_p$ denote the Hilbert space of functions of $(x_1, \ldots, x_p)$ that have an additive form: $f(x) = \sum_j f_j(x_j)$. The standard additive model optimization problem, in the population setting, is

$$\min_{f_j \in \mathcal{H}_j,\, 1 \le j \le p} \mathbb{E}\left(Y - \textstyle\sum_{j=1}^p f_j(X_j)\right)^2 \tag{12.5}$$

and $m(x) = \mathbb{E}(Y \mid X = x)$ is the unknown regression function. Now consider the following modification of this problem that imposes additional constraints:

$$\begin{aligned} (P) \qquad & \min_{\beta \in \mathbb{R}^p, g_j \in \mathcal{H}_j} && \mathbb{E}\left(Y - \textstyle\sum_{j=1}^p \beta_j g_j(X_j)\right)^2 && \text{(12.6a)} \\ & \text{subject to} && \sum_{j=1}^p |\beta_j| \le L && \text{(12.6b)} \\ & && \mathbb{E}\left(g_j^2\right) = 1,\ j = 1, \ldots, p && \text{(12.6c)} \\ & && \mathbb{E}\left(g_j\right) = 0,\ j = 1, \ldots, p && \text{(12.6d)} \end{aligned}$$

noting that g_j is a function while β is a vector. Intuitively, the constraint that β lies in the ℓ_1-ball $\{\beta : \|\beta\|_1 \le L\}$ encourages sparsity of the estimated β, just as for the parametric lasso. When β is sparse, the estimated additive function $f(x) = \sum_{j=1}^p f_j(x_j) = \sum_{j=1}^p \beta_j g_j(x_j)$ will also be sparse, meaning that many of the component functions $f_j(\cdot) = \beta_j g_j(\cdot)$ are identically zero. The constraints (12.6c) and (12.6c) are imposed for identifiability; without (12.6c), for example, one could always satisfy (12.6a) by rescaling.

While this optimization problem makes plain the role ℓ_1 regularization of β to achieve sparsity, it has the unfortunate drawback of not being convex. More specifically, while the optimization problem is convex in β and $\{g_j\}$ separately, it is not convex in β and $\{g_j\}$ jointly.

However, consider the following related optimization problem:

$$\begin{aligned} (Q) \qquad & \min_{f_j \in \mathcal{H}_j} && \mathbb{E}\left(Y - \textstyle\sum_{j=1}^p f_j(X_j)\right)^2 && \text{(12.7a)} \\ & \text{subject to} && \sum_{j=1}^p \sqrt{\mathbb{E}(f_j^2(X_j))} \ \le\ L && \text{(12.7b)} \\ & && \mathbb{E}(f_j) = 0,\ j = 1, \ldots, p. && \text{(12.7c)} \end{aligned}$$

This problem is convex in $\{f_j\}$, as a quadratically constrained quadratic program (QCQP). Moreover, the solutions to problems (P) and (Q) are equivalent:

$$\left(\beta^*, \left\{g_j^*\right\}\right) \text{ optimizes } (P) \text{ implies } \left\{f_j^* = \beta_j^* g_j^*\right\} \text{ optimizes } (Q);$$

$\left\{f_j^*\right\}$ optimizes (Q) implies $\left(\beta^* = (\|f_j\|_2)^T, \left\{g_j^* = f_j^*/\|f_j^*\|_2\right\}\right)$ optimizes (P).

While optimization problem (Q) has the important virtue of being convex, the way it encourages sparsity is not intuitive; the following observation provides some insight. Consider the set $C \subset \mathbb{R}^4$ defined by

$$C = \left\{(f_{11}, f_{12}, f_{21}, f_{22})^T \in \mathbb{R}^4 \, : \, \sqrt{f_{11}^2 + f_{12}^2} + \sqrt{f_{21}^2 + f_{22}^2} \le L\right\} \qquad (12.8)$$

Then the projection $\pi_{12}C$ onto the first two components is an ℓ_2 ball. However, the projection $\pi_{13}C$ onto the first and third components is an ℓ_1 ball. In this way, it can be seen that the constraint $\sum_j \|f_j\|_2 \le L$ acts as an ℓ_1 constraint across components to encourage sparsity, while it acts as an ℓ_2 constraint within components to encourage smoothness, as in a ridge regression penalty. It is thus crucial that the norm $\|f_j\|_2$ appears in the constraint, and not its square $\|f_j\|_2^2$. For the purposes of sparsity, this constraint could be replaced by $\sum_j \|f_j\|_q \le L$ for any $q \ge 1$. In case each f_j is linear, $(f_j(x_{1j}), \ldots, f(x_{nj})) = \beta_j(x_{1j}, \ldots, x_{nj})$, the optimization problem reduces to the lasso.

The use of scaling coefficients together with a nonnegative garrote penalty, similar to our problem (P), is considered by [69]. However, the component functions g_j are fixed, so that the procedure is not asymptotically consistent. The form of the optimization problem (Q) is similar to that of the COSSO for smoothing spline ANOVA models [36]; however, our method differs significantly from the COSSO, as discussed below. In particular, our method is scalable and easy to implement even when p is much larger than n.

12.3 A Backfitting Algorithm for SpAM

We now derive a coordinate descent algorithm for fitting a sparse additive model. We assume that we observe $Y = m(X) + \epsilon$, where ϵ is mean zero Gaussian noise. We write the Lagrangian for the optimization problem (Q) as

$$\mathcal{L}(f, \lambda, \mu) = \frac{1}{2}\mathbb{E}\left(Y - \textstyle\sum_{j=1}^p f_j(X_j)\right)^2 + \lambda \sum_{j=1}^p \sqrt{\mathbb{E}(f_j^2(X_j))} + \sum_j \mu_j \mathbb{E}(f_j). \qquad (12.9)$$

Let $R_j = Y - \sum_{k \ne j} f_k(X_k)$ be the jth residual. The stationary condition for minimizing $\mathcal{L}$ as a function of f_j, holding the other components f_k fixed for $k \ne j$, is expressed in terms of the Frechet derivative $\delta\mathcal{L}$ as

$$\delta\mathcal{L}(f, \lambda, \mu; \delta f_j) = \mathbb{E}\left[(f_j - R_j + \lambda v_j)\delta f_j\right] = 0 \qquad (12.10)$$

for any $\delta f_j \in \mathcal{H}_j$ satisfying $\mathbb{E}(\delta f_j) = 0$, where $v_j \in \partial\sqrt{\mathbb{E}(f_j^2)}$ is an element of the subgradient, satisfying $\sqrt{\mathbb{E}v_j^2} \leq 1$ and $v_j = f_j \Big/ \sqrt{\mathbb{E}(f_j^2)}$ if $\mathbb{E}(f_j^2) \neq 0$. Therefore, conditioning on X_j, the stationary condition (12.10) implies

$$f_j + \lambda v_j = \mathbb{E}(R_j \mid X_j). \tag{12.11}$$

Letting $P_j = \mathbb{E}[R_j \mid X_j]$ denote the projection of the residual onto $\mathcal{H}_j$, the solution satisfies

$$\left(1 + \frac{\lambda}{\sqrt{\mathbb{E}(f_j^2)}}\right) f_j \;=\; P_j \;\text{ if }\; \mathbb{E}(P_j^2) > \lambda \tag{12.12}$$

and $f_j = 0$ otherwise. Condition (12.12), in turn, implies

$$\left(1 + \frac{\lambda}{\sqrt{\mathbb{E}(f_j^2)}}\right) \sqrt{\mathbb{E}(f_j^2)} = \sqrt{\mathbb{E}(P_j^2)} \quad \text{or} \quad \sqrt{\mathbb{E}(f_j^2)} = \sqrt{\mathbb{E}(P_j^2)} - \lambda. \tag{12.13}$$

Thus, we arrive at the following multiplicative soft-thresholding update for f_j:

$$f_j = \left[1 - \frac{\lambda}{\sqrt{\mathbb{E}(P_j^2)}}\right]_+ P_j \tag{12.14}$$

where $[\cdot]_+$ denotes the positive part. In the finite sample case, as in standard backfitting [25], we estimate the projection $\mathbb{E}[R_j \mid X_j]$ by a smooth of the residuals:

$$\hat{P}_j = \mathcal{S}_j R_j \tag{12.15}$$

where $\mathcal{S}_j$ is a linear smoother, such as a local linear or kernel smoother. Let $\hat{s}_j$ be an estimate of $\sqrt{\mathbb{E}[P_j^2]}$. A simple but biased estimate is

$$\hat{s}_j = \frac{1}{\sqrt{n}}\|\hat{P}_j\|_2 = \sqrt{\text{mean}(\hat{P}_j^2)}. \tag{12.16}$$

More accurate estimators are possible; an example is given in the appendix. We have thus derived the SpAM backfitting algorithm given in Figure 12.1.

While the motivating optimization problem (Q) is similar to that considered in the COSSO [36] for smoothing splines, the SpAM backfitting algorithm decouples smoothing and sparsity, through a combination of soft-thresholding and smoothing. In particular, SpAM backfitting can be carried out with any nonparametric smoother; it is not restricted to splines. Moreover, by iteratively estimating over the components and using soft thresholding, our procedure is simple to implement and scales to high dimensions.

```
Input: Data $(X_i, Y_i)$, regularization parameter $\lambda$.

Initialize $f_j = f_j^{(0)}$, for $j = 1, \ldots, p$.

Iterate until convergence:

    For each $j = 1, \ldots, p$:
        Compute the residual: $R_j = Y - \sum_{k \neq j} f_k(X_k)$;
        Estimate the projection $P_j = \mathbb{E}[R_j \mid X_j]$ by smoothing: $\hat{P}_j = \mathcal{S}_j R_j$;
        Estimate the norm $s_j = \sqrt{\mathbb{E}[P_j]^2}$ using, for example, (12.16) or (12.38);
        Soft-threshold: $f_j = \left[1 - \frac{\lambda}{\hat{s}_j}\right]_+ \hat{P}_j$;
        Center: $f_j \leftarrow f_j - \text{mean}(f_j)$.

Output: Component functions $f_j$ and estimator $\hat{m}(X_i) = \sum_j f_j(X_{ij})$.
```

Figure 12.1: THE SPAM BACKFITTING ALGORITHM

12.3.1 SpAM for Nonparametric Logistic Regression

The SpAM backfitting procedure can be extended to nonparametric logistic regression for classification. The additive logistic model is

$$p(Y = 1 \mid X) \equiv p(X; f) = \frac{\exp\left(\sum_{j=1}^{p} f_j(X_j)\right)}{1 + \exp\left(\sum_{j=1}^{p} f_j(X_j)\right)} \tag{12.17}$$

where $Y \in \{0, 1\}$, and the population log-likelihood is

$$\ell(f) = \mathbb{E}\left[Y f(X) - \log\left(1 + \exp f(X)\right)\right] \tag{12.18}$$

Recall that in the local scoring algorithm for generalized additive models [25] in the logistic case, one runs the backfitting procedure within Newton's method. Here one iteratively computes the transformed response for the current estimate f_0

$$Z_i = f_0(X_i) + \frac{Y_i - p(X_i; f_0)}{p(X_i; f_0)(1 - p(X_i; f_0))} \tag{12.19}$$

and weights $w(X_i) = p(X_i; f_0)(1 - p(X_i; f_0)$, and carries out a weighted backfitting of (Z, X) with weights w. The weighted smooth is given by

$$\hat{P}_j = \frac{\mathcal{S}_j(wR_j)}{\mathcal{S}_j w}. \tag{12.20}$$

To incorporate the sparsity penalty, we first note that the Lagrangian is given by

$$\mathcal{L}(f, \lambda, \mu) = \mathbb{E}\left[\log\left(1 + \exp f(X)\right) - Y f(X)\right] + \lambda \sum_{j=1}^{p} \sqrt{\mathbb{E}(f_j^2(X_j))} + \sum_j \mu_j \mathbb{E}(f_j) \tag{12.21}$$

and the stationary condition for component function f_j is $\mathbb{E}\left(p - Y \mid X_j\right) + \lambda v_j = 0$ where v_j is an element of the subgradient $\partial\sqrt{\mathbb{E}(f_j^2)}$. As in the unregularized case, this condition is nonlinear in f, and so we linearize the gradient of the log-likelihood around f_0. This yields the linearized condition

$$\mathbb{E}\left[w(X)(f(X) - Z) \mid X_j\right] + \lambda v_j = 0 \tag{12.22}$$

When $\mathbb{E}(f_j^2) \neq 0$, this implies the condition

$$\left(\mathbb{E}\left(w \mid X_j\right) + \frac{\lambda}{\sqrt{\mathbb{E}(f_j)^2}}\right) f_j(X_j) = \mathbb{E}(wR_j \mid X_j). \tag{12.23}$$

In the finite sample case, in terms of the smoothing matrix $\mathcal{S}_j$, this becomes

$$f_j = \frac{\mathcal{S}_j(wR_j)}{\mathcal{S}_j w + \lambda \Big/ \sqrt{\mathbb{E}(f_j^2)}}. \tag{12.24}$$

If $\|\mathcal{S}_j(wR_j)\|_2 < \lambda$, then $f_j = 0$. Otherwise, this implicit, nonlinear equation for f_j cannot be solved explicitly, so we propose to iterate until convergence:

$$f_j \leftarrow \frac{\mathcal{S}_j(wR_j)}{\mathcal{S}_j w + \lambda\sqrt{n}\,/\|f_j\|_2}. \tag{12.25}$$

When $\lambda = 0$, this yields the standard local scoring update (12.20). An example of logistic SpAM is given in Section 12.5.

12.4 Properties of SpAM

12.4.1 SpAM is Persistent

The notion of risk consistency, or persistence, was studied by [27] and [21] in the context of linear models. Let (X, Y) denote a new pair (independent of the observed data) and define the predictive risk when predicting Y with $f(X)$ by

$$R(f) = \mathbb{E}(Y - f(X))^2. \tag{12.26}$$

Since we consider predictors of the form $f(x) = \sum_j \beta_j g_j(x_j)$ we also write the risk as $R(\beta, g)$ where $\beta = (\beta_1, \ldots, \beta_p)$ and $g = (g_1, \ldots, g_p)$. Following [21], we say that an estimator $\hat{m}_n$ is *persistent* relative to a class of functions $\mathcal{M}_n$ if

$$R(\hat{m}_n) - R(m_n^*) \xrightarrow{P} 0 \tag{12.27}$$

where $m_n^* = \operatorname{argmin}_{f \in \mathcal{M}_n} R(f)$ is the predictive oracle. [21] showed that the lasso is persistent for the class of linear models $\mathcal{M}_n = \{f(x) = x^T\beta : \|\beta\|_1 \le L_n\}$ if $L_n = o((n/\log n)^{1/4})$. We show a similar result for SpAM.

Theorem 5. *Suppose that $p_n \le e^{n^\varepsilon}$ for some $\varepsilon < 1$. Then SpAM is persistent relative to the class of additive models $\mathbb{A}_n = \left\{ f(x) = \sum_{j=1}^p \beta_j g_j(x_j) \ : \ \|\beta\|_1 \le L_n \right\}$ if $L_n = o\left(n^{(1-\varepsilon)/4}\right)$.*

12.4.2 SpAM is Sparsistent

In the case of linear regression, with $m_j(X_j) = \beta_j^T X_j$, [56] shows that under certain conditions on n, p, $s = |\text{supp}(\beta)|$, and the design matrix X, the lasso recovers the sparsity pattern asymptotically; that is, the lasso estimator $\hat{\beta}_n$ is *sparsistent*: $p\left(\text{supp}(\beta) = \text{supp}(\hat{\beta}_n)\right) \to 1$. We show a similar result for SpAM with the sparse backfitting procedure.

For the purpose of analysis, we use orthogonal function regression as the smoothing procedure. For each $j = 1, \ldots, p$ let ψ_j be an orthogonal basis for $\mathcal{H}_j$. We truncate the basis to finite dimension d_n, and let $d_n \to \infty$ such that $d_n/n \to 0$. Let Ψ_j denote the $n \times d$ matrix $\Psi_j(i,k) = \psi_{jk}(X_{ij})$. If $A \subset \{1, \ldots, p\}$, we denote by Ψ_A the $n \times d|A|$ matrix where for each $i \in A$, Ψ_i appears as a submatrix in the natural way. The SpAM optimization problem can then be written as

$$\min_\beta \frac{1}{2n}\left(Y - \textstyle\sum_{j=1}^p \Psi_j\beta_j\right)^2 + \lambda_n \sum_{j=1}^p \sqrt{\frac{1}{n}\beta_j^T \Psi_j^T \Psi_j \beta_j} \tag{12.28}$$

where each β_j is a d-dimensional vector. Let S denote the true set of variables $\{j : m_j \ne 0\}$, with $s = |S|$, and let S^c denote its complement. Let $\hat{S}_n = \{j : \hat{\beta}_j \ne 0\}$ denote the estimated set of variables from the minimizer $\hat{\beta}_n$ of (12.28).

Theorem 6. *Suppose that* Ψ *satisfies the conditions*

$$\Lambda_{max}\left(\frac{1}{n}\Psi_S^T\Psi_S\right) \leq C_{max} < \infty \quad \textit{and} \quad \Lambda_{min}\left(\frac{1}{n}\Psi_S^T\Psi_S\right) \geq C_{min} > 0 \quad (12.29)$$

$$\left\|\left(\tfrac{1}{n}\Psi_{S^c}^T\Psi_S\right)\left(\tfrac{1}{n}\Psi_S^T\Psi_S\right)^{-1}\right\|_2^2 s\log s \leq 1-\delta < 1. \quad (12.30)$$

Let the regularization parameter $\lambda_n \to 0$ *be chosen to satisfy*

$$\lambda_n\sqrt{s} \to 0, \quad \frac{s}{d_n\lambda_n} \to 0, \quad and \quad \frac{d_n(\log d_n + \log(p-s))}{n\lambda_n^2} \to 0. \quad (12.31)$$

Then SpAM is sparsistent: $p\left(\hat{S}_n = S\right) \longrightarrow 1.$

12.5 Experiments

In this section we present experimental results for SpAM applied to both synthetic and real data, including regression and classification examples that illustrate the behavior of the algorithm in various conditions. We first use simulated data to investigate the performance of the SpAM backfitting algorithm, where the true sparsity pattern is known. We then apply SpAM to some real data. If not explicitly stated otherwise, the data are always rescaled to lie in a d-dimensional cube $[0,1]^d$, and a kernel smoother with Gaussian kernel is used. To tune the penalization parameter λ, we use a C_p statistic, which is defined as

$$C_p(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \textstyle\sum_{j=1}^{p}\widehat{f}_j(X_j)\right)^2 + \frac{2\hat{\sigma}^2}{n}\sum_{j=1}^{p}\text{trace}(\mathcal{S}_j)\,\mathbf{1}[\hat{f}_j \neq 0] \quad (12.32)$$

where $\mathcal{S}_j$ is the smoothing matrix for the j-th dimension and $\hat{\sigma}^2$ is the estimated variance.

12.5.1 Simulations

We first apply SpAM to an example from [24]. A dataset with sample size $n = 150$ is generated from the following 200-dimensional additive model:

$$Y_i = f_1(x_{i1}) + f_2(x_{i2}) + f_3(x_{i3}) + f_4(x_{i4}) + \epsilon_i \quad (12.33)$$

$$f_1(x) = -2\sin(2x),\ f_2(x) = x^2 - \tfrac{1}{3},\ f_3(x) = x - \tfrac{1}{2},\ f_4(x) = e^{-x} + e^{-1} - 1 \quad (12.34)$$

and $f_j(x) = 0$ for $j \geq 5$ with noise $\epsilon_i \sim \mathcal{N}(0,1)$. These data therefore have 196 irrelevant dimensions. The results of applying SpAM with the plug-in bandwidths are summarized in Figures (12.2),(12.3),(12.4).

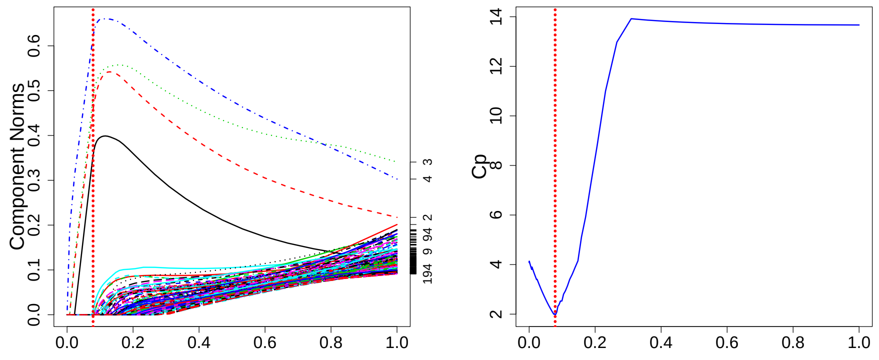


Figure 12.2: (Simulated data) left: The empirical ℓ_2 norm of the estimated components as plotted against the tuning parameter λ; the value on the x-axis is proportional to $\sum_j \|\hat{f}_j\|_2$. right: The C_p scores against the tuning parameter λ; the dashed vertical line corresponds to the value of λ which has the smallest C_p score.

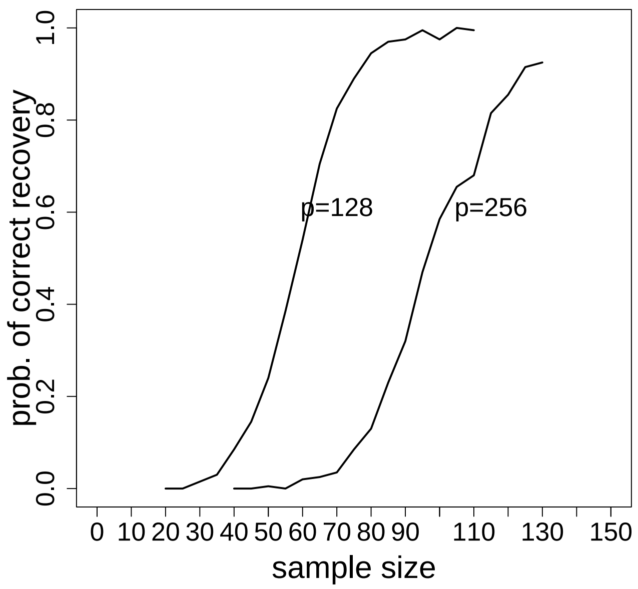


Figure 12.3: (Simulated data) The proportion of 200 trials where the correct relevant variables are selected, as a function of sample size n.

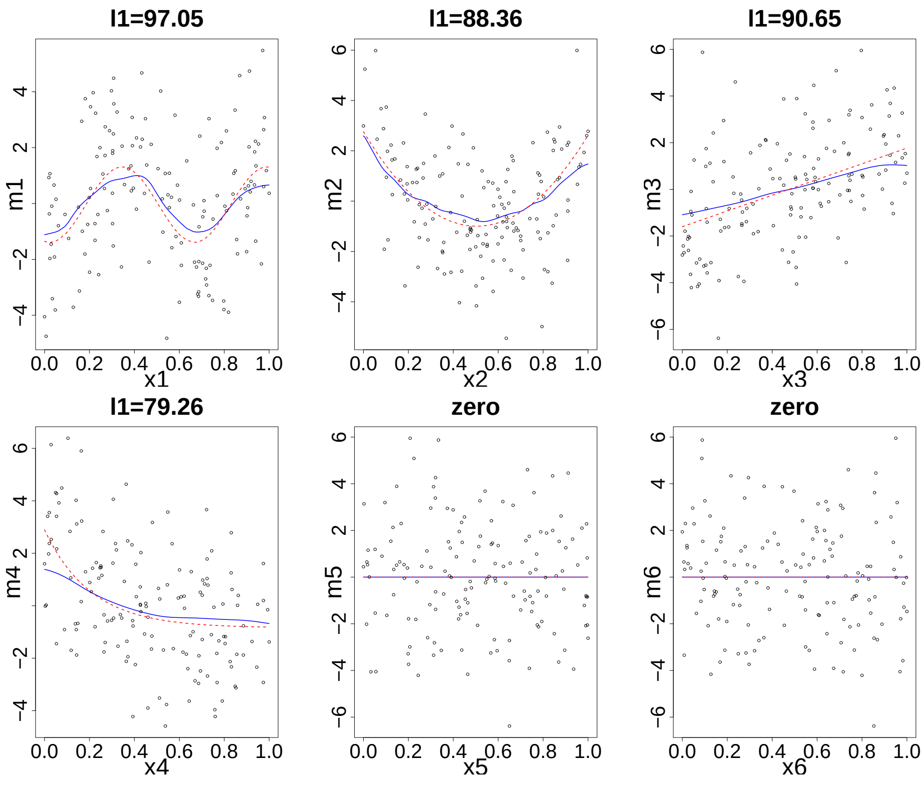


Figure 12.4: (Simulated data) Estimated (solid lines) versus true additive component functions (dashed lines) for the first 6 dimensions; the remaining components are zero.

12.5.2 Boston Housing

The Boston housing data was collected to study house values in the suburbs of Boston; there are altogether 506 observations with 10 covariates. The dataset has been studied by many other authors [24, 36], with various transformations proposed for different covariates. To explore the sparsistency properties of our method, we add 20 irrelevant variables. Ten of them are randomly drawn from Uniform$(0, 1)$, the remaining ten are a random permutation of the original ten covariates, so that they have the same empirical densities.

The full model (containing all 10 chosen covariates) for the Boston Housing data is:

$$\begin{aligned}\texttt{medv} &= \alpha + f_1(\texttt{crim}) + f_2(\texttt{indus}) + f_3(\texttt{nox}) + f_4(\texttt{rm}) + f_5(\texttt{age}) \\ &\quad + f_6(\texttt{dis}) + f_7(\texttt{tax}) + f_8(\texttt{ptratio}) + f_9(\texttt{b}) + f_{10}(\texttt{lstat}) \qquad (12.35)\end{aligned}$$

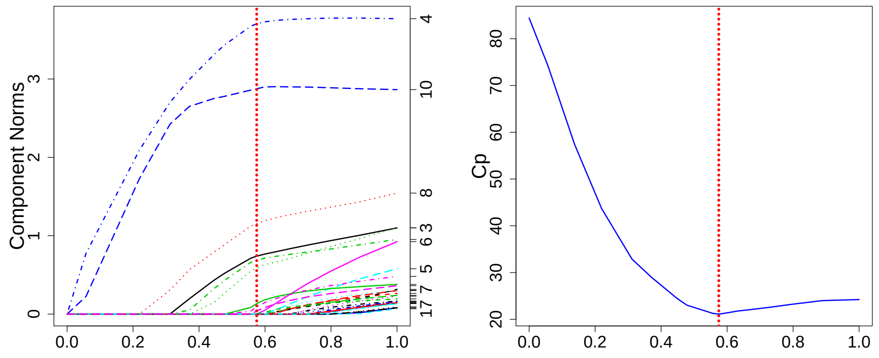


Figure 12.5: (Boston housing) Left: The empirical ℓ_2 norm of the estimated components versus the regularization parameter λ. Right: The C_p scores against λ; the dashed vertical line corresponds to best C_p score.

The result of applying SpAM to this 30 dimensional dataset is shown in Figures (12.5),(12.6). SpAM identifies 6 nonzero components. It correctly zeros out both types of irrelevant variables. From the full solution path, the important variables are seen to be `rm`, `lstat`, `ptratio`, and `crim`. The importance of variables `nox` and `b` are borderline. These results are basically consistent with those obtained by other authors [24]. However, using C_p as the selection criterion, the variables `indux`, `age`, `dist`, and `tax` are estimated to be irrelevant, a result not seen in other studies.

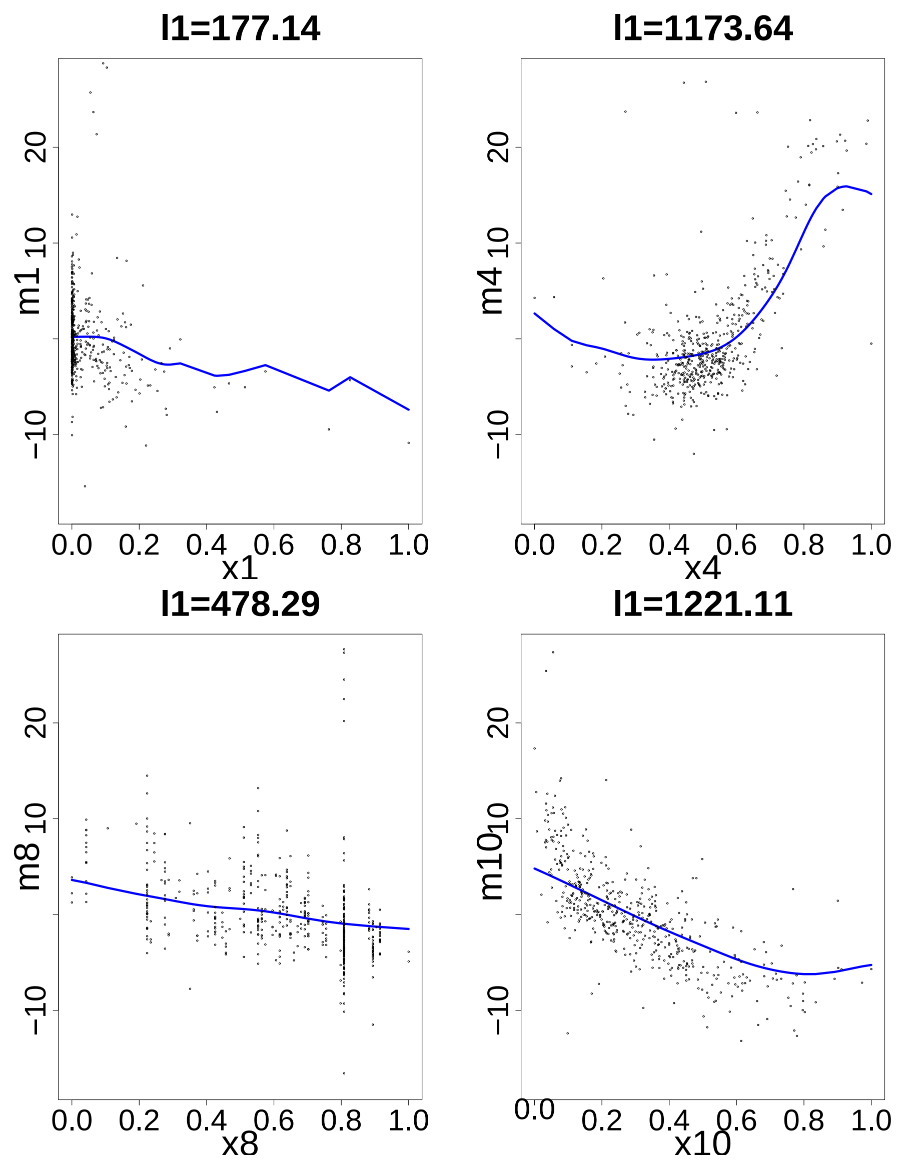


Figure 12.6: (Boston Housing) Additive fits for four relevant variables.

12.5.3 SpAM for Spam

Here we consider an email spam classification problem, using the logistic SpAM backfitting algorithm from Section 12.3.1. This dataset has been studied by [26], using a set of 3,065 emails as a training set, and conducting hypothesis tests to choose significant variables; there are a total of 4,601 observations with $p = 57$ attributes, all numeric. The attributes measure the percentage of specific words or characters in the email, the average and maximum run lengths of upper case letters, and the total number of such letters. To demonstrate how SpAM performs well with sparse data, we only sample $n = 300$ emails as the training set, with the remaining 4301 data points used as the test set. We also use the test data as the hold-out set to tune the penalization parameter λ.

$\lambda(\times 10^{-3})$	Error	# zeros	selected variables
5.5	0.2009	55	{ 8,54}
5.0	0.1725	51	{ 8, 9, 27, 53, 54, 57}
4.5	0.1354	46	{7, 8, 9, 17, 18, 27, 53, 54, 57, 58}
4.0	0.1083 (√)	20	{4, 6–10, 14–22, 26, 27, 38, 53–58}
3.5	0.1117	0	all
3.0	0.1174	0	all
2.5	0.1251	0	all
2.0	0.1259	0	all

Figure 12.7: (Email spam) Classification accuracies and variable selection for logistic SpAM.

The results of a typical run of logistic SpAM are summarized in Figures (12.7) and (12.8) using plug-in bandwidths. It is interesting to note that even with this relatively small sample size, logistic SpAM recovers a sparsity pattern that is consistent with previous authors' results. For example, in the best model chosen by logistic SpAM, according to error rate, the 33 selected variables cover 80% of the significant predictors as determined by [26].

12.6 Estimating $\sqrt{\mathbb{E}[P_j^2]}$

To construct a more accurate of estimator of $\sqrt{\mathbb{E}[P_j^2]}$, let

$$\mathcal{S}_j(x) = (S(x, X_{1j}), \ldots, S(x, X_{nj}))^T$$

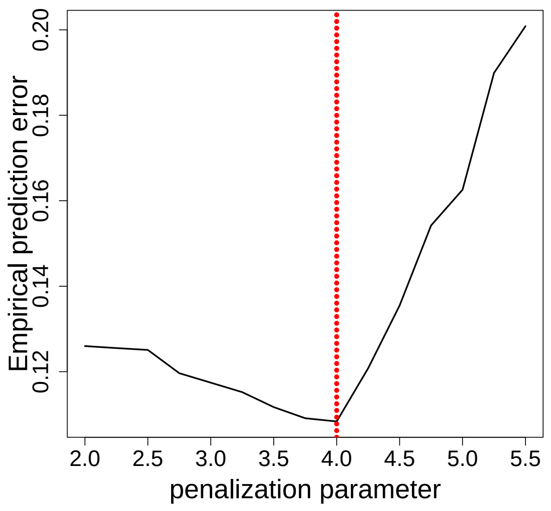


Figure 12.8: (Email Spam) The C_p scores against λ; the dashed vertical line corresponds to best C_p score.

denote the xth column of the smoothing matrix, and $\mathcal{G}_x = \mathcal{S}_j(x)\mathcal{S}_j(x)^T$. Then $\hat{P}_j(x) = \mathcal{S}_j(x)^T R_j$, and $\hat{P}_j(x)^2 = R_j^T \mathcal{G}_x R_j$. To estimate $\mathbb{E}[P_j^2(x)]$, we use the quadratic form identity

$$\mathbb{E}(X^T Q X) = \text{tr}(\Sigma Q) + \mu^T Q \mu \;\text{ in case }\; X \sim N(\mu, \Sigma). \tag{12.36}$$

Thus, if the noise $\epsilon_i \sim N(0, \sigma^2)$ is Gaussian, then

$$\mathbb{E}(R_j^T \mathcal{G}_x R_j^T \mid X_j) = \sigma^2 \text{tr}(\mathcal{G}_x) + \mathbb{E}(R_j \mid X_j)^T \mathcal{G}_x \,\mathbb{E}(R_j \mid X_j). \tag{12.37}$$

Defining $\mathbb{G} = \frac{1}{n}\sum_i \mathcal{G}_{X_i}$ and plugging in our estimate $\hat{P}_j$ for $\mathbb{E}(R_j \mid X_j)$ yields the estimator

$$\hat{s}_j = \sqrt{\sigma^2 \text{tr}(\mathbb{G}) + \hat{P}_j^T \mathbb{G} \hat{P}_j}. \tag{12.38}$$

12.7 Proof of Theorem 5

Proof of Theorem 5.

Let $(Y_1, X_1), \ldots, (Y_n, X_n)$ be n data points where $X_i \in \mathbb{R}^p$ and $Y_i \in \mathbb{R}$. The model is

$$Y_i = \alpha + m(X_i) + \epsilon_i \tag{12.39}$$

where

$$m \in \mathbb{A}_n(L_n) = \left\{ m = \sum_{j=1}^{p} \beta_j m_j(x_j),\ \ m_j \in \mathcal{T}_j,\ \ \sum_{j=1}^{p} |\beta_j| \le L_n \right\}, \tag{12.40}$$

$$\mathcal{T}_j = \left\{ m_j \in \mathcal{H}_j :\ \int m_j(x_j)dx_j = 0,\ \ \int m_j^2(x_j)dx_j = 1,\ \ \sup_x |m_j(x)| \le C \right\} \tag{12.41}$$

and $\mathcal{H}_j$ is a class of smooth functions such as the Sobolev space:

$$\mathcal{H}_j = \left\{ m_j :\ \int m_j''(x_j)^2\, dx_j < \infty,\ m_j, m_j' \text{ are absolutely continuous} \right\}. \tag{12.42}$$

We begin with some notation. If $\mathcal{M}$ is a class of functions then the L_∞ bracketing number $N_{[\,]}(\epsilon)$ is smallest number of pairs $B = \{(\ell_1, u_1), \ldots, (\ell_k, u_k)\}$ such that $\|u_j - \ell_j\|_\infty \le \epsilon$, $1 \le j \le k$, and such that for every $m \in \mathcal{M}$ there exists $(\ell, u) \in B$ such that $\ell \le m \le u$. For the Sobolev space $\mathcal{T}_j$,

$$\log N_{[\,]}(\epsilon, \mathcal{T}_j) \le K \left(\frac{1}{\epsilon}\right)^{1/2} \tag{12.43}$$

for some $K > 0$. The bracketing integral is defined to be

$$J_{[\,]}(\delta) = \int_0^\delta \sqrt{\log N_{[\,]}(u)}du. \tag{12.44}$$

A useful empirical process inequality (see Corollary 19.35 of van der Vaart 1998, for example), is

$$\mathbb{E}\left(\sup_{g \in \mathcal{M}} |\hat{\mu}(g) - \mu(g)| \right) \le \frac{C\, J_{[\,]}(\|F\|_\infty)}{\sqrt{n}} \tag{12.45}$$

for some $C > 0$, where $F(x) = \sup_{g \in \mathcal{M}} |g(x)|$, $\mu(g) = \mathbb{E}(g(X))$ and $\hat{\mu}(g) = n^{-1} \sum_{i=1}^n g(X_i)$.

Set $Z \equiv (Z_0, \ldots, Z_p) = (Y, X_1, \ldots, X_p)$ and note that

$$R(\beta, g) = \sum_{j=0}^{p} \sum_{k=0}^{p} \beta_j \beta_k \mathbb{E}(g_j(Z_j) g_k(Z_k)) \tag{12.46}$$

where we define $g_0(x) = z_0$ and $\beta_0 = -1$. Also define

$$\hat{R}(\beta, g) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=0}^{p} \sum_{k=0}^{p} \beta_j \beta_k g_j(Z_{ij}) g_k(Z_{ik}). \tag{12.47}$$

Note that the SpAM estimator is the minimizer of $\hat{R}(\beta, g)$ subject to $\sum_j \beta_j g_j(x_j) \in \mathbb{A}_n(L_n)$. For all (β, g),

$$|\hat{R}(\beta, g) - R(\beta, g)| \le \|\beta\|_1^2 \max_{jk} \sup_{g_j \in \mathcal{T}_j, g_k \in \mathcal{T}_k} |\hat{\mu}_{jk}(g) - \mu_{jk}(g)| \qquad (12.48)$$

where

$$\hat{\mu}_{jk}(g) = n^{-1} \sum_{i=1}^{n} \sum_{jk} \mathbb{E}(g_j(Z_{ij}) g_k(Z_{ik}))$$
$$\mu_{jk}(g) = \mathbb{E}(g_j(Z_j) g_k(Z_k))$$

From (12.43) it follows that

$$\log N_{[\,]}(\epsilon, \mathbb{A}_n) \le 2 \log p_n + K \left(\frac{1}{\epsilon}\right)^{1/2}. \qquad (12.49)$$

Hence, $J_{[\,]}(C, \mathbb{A}) = O(\sqrt{\log p_n})$ and it follows from (12.45) that

$$\max_{jk} \sup_{g_j \in \mathcal{T}_j, g_k \in \mathcal{T}_k} |\hat{\mu}_{jk}(g) - \mu_{jk}(g)| = O\left(\sqrt{\frac{\log p_n}{n}}\right) = O\left(\frac{1}{n^{(1-\varepsilon)/2}}\right). \qquad (12.50)$$

We conclude that

$$\sup_{g \in \mathbb{A}} |\hat{R}(g) - R(g)| = O\left(\frac{L_n^2}{n^{(1-\varepsilon)/2}}\right) = o(1). \qquad (12.51)$$

Therefore,

$$\begin{aligned} R(m^*) &\le R(\hat{m}_n) \le \hat{R}(\hat{m}_n) + o_P(1) \\ &\le \hat{R}(m^*) + o_P(1) \le R(m^*) + o_P(1) \end{aligned}$$

and the conclusion follows. □

12.8 Proof of Theorem 6

Proof of Theorem 6.

There exists an orthonormal basis (the Fourier basis) $\mathcal{B}_j = \{\psi_{j1}, \psi_{j2}, \ldots\}$ for the second-order Sobolev space $\mathcal{H}_j$ such that $m_j \in \mathcal{H}_j$ if and only if

$$m_j = \sum_{k=1}^{\infty} \beta_{jk} \psi_{jk} \qquad (12.52)$$

and $\sum_{k=1}^{\infty} \beta_{jk}^2 k^4 < C^2$, for some $C < \infty$. The basis is bounded, with

$$\sup_X |\psi_{jk}(X)| \leq B, \text{ for a constant } B < \infty.$$

Thus we can write

$$Y_i = \sum_{j=1}^{p} \sum_{k=1}^{\infty} \beta_{jk}^* \psi_{jk}(X_{ij}) + \epsilon_i \tag{12.53}$$

It can be shown that the sparse backfitting procedure with an orthogonal function regression smoother, with a truncated basis of size d_n, solves the following optimization problem,

$$\min_\beta \sum_{i=1}^{n} \frac{1}{2n} \left(Y_i - \sum_{j=1}^{p} \sum_{k=1}^{d_n} \beta_{jk} \psi_{jk}(X_{ij}) \right)^2 + \lambda \sum_{j=1}^{p} \sqrt{\frac{1}{n} \sum_i \sum_{k,k'} \beta_{jk} \beta_{jk'} \psi_{jk}(X_{ij}) \psi_{jk'}(X_{ij})} \tag{12.54}$$

To simplify notation, let β_j be the d_n dimensional vector $\{\beta_{jk},\ k = 1, \ldots, d_n\}$, and Ψ_j the $n \times d_n$ matrix, $\Psi_j[i,k] = \psi_{jk}(X_{ij})$. If $A \subset \{1, \ldots, p\}$, we denote by Ψ_A the $n \times d|A|$ matrix where for each $i \in A$, Ψ_i appears as a submatrix in the natural way.

The optimization task can then be written as

$$\min_\beta \frac{1}{2n} \left(Y - \sum_{j=1}^{p} \Psi_j \beta_j \right)^2 + \lambda \sum_{j=1}^{p} \sqrt{\frac{1}{n} \beta_j^\top \Psi_j^\top \Psi_j \beta_j} \tag{12.55}$$

We now state assumptions on the design and design parameters. Let S be the true sparsity pattern, and let $S^c = \{1, \ldots, p\} \backslash S$ be the set of irrelevant covariates.

(A1) *Dependence conditions:*

$$\Lambda_{\max} \left(\frac{1}{n} \Psi_S^\top \Psi_S \right) \leq C_{\max} < \infty \tag{12.56}$$

$$\Lambda_{\min} \left(\frac{1}{n} \Psi_S^\top \Psi_S \right) \geq C_{\min} > 0 \tag{12.57}$$

(A2) *Incoherence conditions:*

$$\left\| \left(\frac{1}{n} \Psi_{S^c}^\top \Psi_S \right) \left(\frac{1}{n} \Psi_S^\top \Psi_S \right)^{-1} \right\|_2^2 s \log s \leq (1 - \epsilon) \tag{12.58}$$

for some $\epsilon > 0$.

(A3) *Truncation conditions:*

$$d_n \to \infty \tag{12.59}$$

$$\frac{d_n}{n} \to 0 \tag{12.60}$$

(A4) *Penalty conditions:*

$$\lambda_n \sqrt{s} \to 0 \tag{12.61}$$

$$\frac{d_n(\log d_n + \log(p-s))}{n\lambda_n^2} \to 0 \tag{12.62}$$

$$\frac{s}{d_n}\frac{1}{\lambda_n} \to 0 \tag{12.63}$$

Theorem 7. *Given the model in* (12.53)*, and design settings* (n, p, s, d, λ) *such that conditions (A1) through (A4) are satisfied, then* $\mathbb{P}\left(\hat{S}_n = S\right) \to 1.$

A commonly used basis truncation size is $d_n = n^{1/5}$, which achieves the minimax error rate in the one-dimensional case. The theorem under this design setting gives,

Corollary 12.8.1.. *Given the model in* (12.53)*, penalty settings such that*

$$\lambda_n \sqrt{s} \to 0, \quad \frac{\log n(p-s)}{n^{4/5}\lambda_n^2} \to 0, \quad \frac{s}{n^{1/5}}\lambda_n \to 0 \tag{12.64}$$

and design settings such that $d = O(n^{1/5})$ *and conditions (A1) and (A2) are satisfied, then* $\mathbb{P}\left(\hat{S}_n = S\right) \to 1.$

Let $F(\beta)$ denote the objective function of the optimization problem in (12.54), and let $G(\beta) = \sum_{j=1}^{p} \sqrt{\frac{1}{n}\beta_j^\top \Psi_j^\top \Psi_j \beta_j}$ denote the penalty part, Then a vector $\hat{\beta} \in \mathbb{R}^{d_n p}$ is an optimum of the above objective function if and only if there exists a subgradient $\hat{g} \in \partial G(\hat{\beta})$, such that

$$\frac{1}{n}\Psi^\top \left(\sum_j \Psi_j \beta_j - Y \right) + \lambda_n \hat{g} = 0 \tag{12.65}$$

The subdifferential $\hat{g}$ of $G(\beta)$ takes the form

$$\hat{g}_j = \frac{\frac{1}{n}\Psi_j^\top \Psi_j \hat{\beta}_j}{\sqrt{\frac{1}{n}\hat{\beta}_j^\top \Psi_j^\top \Psi_j \hat{\beta}_j}}, \qquad \text{for } \hat{\beta}_j \neq \mathbf{0} \tag{12.66}$$

$$\hat{g}_j^\top \left[\frac{1}{n}\Psi_j^\top \Psi_j\right]^{-1} \hat{g}_j \leq 1 \qquad \text{for } \hat{\beta}_j = \mathbf{0} \tag{12.67}$$

We now proceed by a "witness" proof technique. We set $\hat{\beta}_{S^c} = 0$ and $\hat{g}_S = \partial G_S(\hat{\beta}_S)$, and, obtaining $\hat{\beta}_S$ and $\hat{g}_{S^c}$ from the stationary condition in (12.65), we show that with high probability,

$$\hat{g}_j^\top \left(\frac{1}{n}\Psi_j^\top \Psi_j\right)^{-1} \hat{g}_j \leq 1,\ j \in S^c \tag{12.68}$$

$$\hat{\beta}_S \neq 0 \tag{12.69}$$

This shows that there exists a optimal solution to the optimization problem in (12.54) which has the same sparsity pattern as the model. Since it can be shown that every solution to the optimization problem has the same sparsity pattern, this will prove the required result.

Setting $\hat{\beta}_{S^c} = 0$ and $\hat{g}_S = \partial G_S(\hat{\beta}_S)$ in the stationary condition for β_S gives

$$\frac{1}{n}\Psi_j^\top \left(\Psi_S \beta_S - Y\right) + \lambda_n g_j = 0,\ j \in S \tag{12.70}$$

which can be summarized as

$$\frac{1}{n}\Psi_S^\top \left[\Psi_S \beta_S - Y\right] + \lambda_n g_j = 0 \tag{12.71}$$

Let $V_n = Y - \Psi_S \beta^*_S - W_n$, where W_n denotes the noise vector. Then

$$\begin{aligned}
|V_i| &= |\sum_{j\in S}\sum_{k=d+1}^{\infty} \beta_{jk}\Psi_{jk}(X_{ij})| && (12.72)\\
&\leq B\sum_{j\in S}\sum_{k=d+1}^{\infty} |\beta_{jk}| && (12.73)\\
&= B\sum_{j\in S}\sum_{k=d+1}^{\infty} \frac{|\beta_{jk}|k^2}{k^2} && (12.74)\\
&\leq B\sum_{j\in S}\sqrt{\sum_{k=d+1}^{\infty}\beta_{jk}^2 k^4}\sqrt{\sum_{k=d+1}^{\infty}\frac{1}{k^4}} && (12.75)\\
&\leq sBC\sqrt{\sum_{k=d+1}^{\infty}\frac{1}{k^4}} && (12.76)\\
&\leq \frac{sB'}{d_n^{3/2}} \text{ for some constant } B' > 0 && (12.77)
\end{aligned}$$

Therefore

$$\|V_n\|_\infty \leq B' s d^{-3/2}, \qquad (12.78)$$

Letting $\Sigma_{SS} = \frac{1}{n}\left[\Psi_S^\top \Psi_S\right]$, we have

$$\beta_S - \beta^*_S = \Sigma_{SS}^{-1}\left[\frac{1}{\Psi}{}_S^{\top} W_n\right] + \Sigma_{SS}^{-1}\left[\frac{1}{n}\Psi_S^{-1}V_n\right] - \lambda_n \Sigma_{SS}^{-1} g_S \qquad (12.79)$$

This allows us to get the ℓ_∞ bound,

$$\begin{aligned}
\|\beta_S - \beta^*_S\|_\infty = & \left\|\Sigma_{SS}^{-1}\left[\frac{1}{n}\Psi_S^\top W_n\right]\right\|_\infty + \|\Sigma_{SS}^{-1}\|_\infty \left|\frac{1}{n}\Psi_S^\top V_n\right|_\infty \\
& + \lambda_n \left\|\Sigma_{SS}^{-1} g_S\right|_\infty \qquad (12.80)
\end{aligned}$$

Let $\rho_n = \min_{j\in S}\max_{k\in\{1,\ldots,d_n\}} |\beta^*_{jk}| > 0$. It suffices to show that $\|\beta_S - \beta^*_S\|_\infty < \frac{\rho_n}{2}$ to ensure that $\beta_j \not\equiv 0,\ j \in S$. We now proceed to bound the quantities in (12.80).

$$\begin{aligned}
\|\Sigma_{SS}^{-1}\|_\infty &\leq \|\Sigma_{SS}^{-1}\|_2\sqrt{sd} && (12.81)\\
&\leq \frac{\sqrt{sd}}{C_{\min}} && (12.82)
\end{aligned}$$

$$
\begin{aligned}
1 &\geq g_j^\top \left[\frac{1}{n} \Psi_j^\top \Psi_j \right]^{-1} g_j && (12.83)\\
&\geq \frac{1}{C_{\max}} \|g_j\|_2^2 && (12.84)\\
\|g_j\|_2 &\leq \sqrt{C_{\max}} && (12.85)
\end{aligned}
$$

This gives the following bounds,

$$
\begin{aligned}
\|g_S\|_\infty &= \max_{j \in S} \|g_j\|_\infty && (12.86)\\
&\leq \max_{j \in S} \|g_j\|_2 && (12.87)\\
&\leq \sqrt{C_{\max}} && (12.88)
\end{aligned}
$$

Also,

$$
\begin{aligned}
\left\|\Sigma_{SS}^{-1} g_S\right\|_\infty &\leq \left\|\Sigma_{SS}^{-1} g_S\right\|_2 && (12.89)\\
&\leq \left\|\Sigma_{SS}^{-1}\right\|_2 \|g_S\|_2 && (12.90)\\
&\leq \frac{\sqrt{C_{\max} s}}{C_{\min}} && (12.91)
\end{aligned}
$$

From (12.78), we get

$$
\begin{aligned}
\frac{1}{n} \Psi_{jk}^\top V_n &\leq \left| \frac{1}{n} \sum_i \Psi_{jk}(X_{ij}) \right| \|V_n\|_\infty && (12.92)\\
&\leq B'^2 s d^{-3/2} && (12.93)\\
& && (12.94)
\end{aligned}
$$

Finally, consider $Z := \Sigma_{SS}^{-1} \left[\frac{1}{n} \Psi_S^\top W_n\right]$. Note that $W_n \sim N(0, \sigma^2 I)$, so that Z is Gaussian as well, with mean zero. Consider its l-th component, $Z_l = e_l^\top Z$. Then $\mathbb{E}[Z_l] = 0$, and $\text{Var}[Z_l] = \frac{\sigma^2}{n} e_l^\top \Sigma_{SS}^{-1} e_l \leq \frac{\sigma^2}{C_{\min} n}$. It can then be shown (Ledoux and Talagrand, 1991) that

$$
\begin{aligned}
\mathbb{E}[\|Z\|_\infty] &\leq 3 \sqrt{\log(sd) \max_l \text{Var}[Z_l]} && (12.95)\\
&\leq 3 \sqrt{\frac{\sigma^2 \log(sd)}{n C_{\min}}} && (12.96)
\end{aligned}
$$

An application of Markov's inequality then proves the desired result,

$$\begin{aligned} p\left[\|\beta_S - \beta^*_S\|_\infty > \frac{\rho_n}{2}\right] &\leq p\left[\|Z\|_\infty + \lambda_n \frac{\sqrt{C_{\max} s}}{C_{\min}} + \sqrt{s} B'^2 s d^{-1} \geq \frac{\rho_n}{2}\right] \\ &\leq \frac{1}{\rho_n}\left\{\mathbb{E}\left[\|Z\|_\infty\right] + \lambda_n \frac{\sqrt{C_{\max} s}}{C_{\min}} + B'^2(s^{3/2}/d)\right\} \\ &\leq \frac{1}{\rho_n}\left\{3\sqrt{\frac{\sigma^2 \log(sd)}{nC_{\min}}} + \lambda_n \frac{\sqrt{C_{\max} s}}{C_{\min}} + B'^2(s^{3/2}/d)\right\} \end{aligned}$$

which converges to zero under the given assumptions.

We now analyze $\hat{g}_{S^c}$. The stationary condition for $q \in S^c$ is given by

$$\frac{1}{n}\Psi_q^\top \left[\Psi_S \beta_S - \Psi_S \beta^*_S - W_n - V_n\right] + \lambda_n g_q = 0 \qquad (12.97)$$

Thus,

$$\begin{aligned} \lambda_n g_q &= -\left[\frac{1}{n}\Psi_q^\top \Psi_S\right]\left[\beta_S - \beta^*_S\right] + \left(\frac{1}{n}\Psi_q^\top\right)\left[W_n + V_n\right] && (12.98) \\ &= -\Sigma_{S^cS}\left[\beta_S - \beta^*_S\right] + \left(\frac{1}{n}\Psi_q^\top\right)\left[W_n + V_n\right] && (12.99) \\ &= \Sigma_{S^cS}\Sigma_{SS}^{-1}\lambda_n g_S + \Sigma_{S^cS}\Sigma_{SS}^{-1}\left[\left(\frac{1}{n}\Psi_S^\top\right)(W_n + V_n)\right] \\ &\quad + \left(\frac{1}{n}\Psi_q^\top\right)\left[W_n + V_n\right] && (12.100) \end{aligned}$$

Letting $\Sigma_{qq} = \frac{1}{n}\Psi_q^\top \Psi_q$, we want $g_q^\top \Sigma_{qq}^{-1} g_q \leq 1$. Since

$$\begin{aligned} \sqrt{g_q^\top \Sigma_{qq}^{-1} g_q} &\leq \sqrt{\lambda_{\max}\left[\Sigma_{qq}^{-1}\right]\|g_q\|^2} && (12.101) \\ &\leq \frac{1}{\sqrt{l_{\min}}}\|g_q\|_2 && (12.102) \end{aligned}$$

it suffices to show that $\sup_{q \in S^c}\|g_q\|_2 \leq \sqrt{l_{\min}}$. From (12.98), we have,

$$\begin{aligned} \lambda_n \|g_q\|_2 &\leq \left\|\Sigma_{S^cS}\Sigma_{SS}^{-1}\right\|_2 \lambda_n \|g_S\|_2 + \left\|\Sigma_{S^cS}\Sigma_{SS}^{-1}\right\|_2 \\ &\quad \left\{\left\|\frac{1}{n}\Psi_S^\top W_n\right\|_2 + \left\|\frac{1}{n}\Psi_S^\top V_n\right\|_2\right\} + \left\{\left\|\frac{1}{n}\Psi_q^\top W_n\right\|_2 + \left\|\frac{1}{n}\Psi_q^\top V_n\right\|_2\right\} \\ &\leq \lambda_n \delta \sqrt{sC_{\max}} + \delta\sqrt{sd}\left\{\left\|\frac{1}{n}\Psi_S^\top W_n\right\|_\infty + \left\|\frac{1}{n}\Psi_S^\top V_n\right\|_\infty\right\} \\ &\quad + \sqrt{d}\left\{\left\|\frac{1}{n}\Psi_q^\top W_n\right\|_\infty + \left\|\frac{1}{n}\Psi_q^\top V_n\right\|_\infty\right\} \\ &\leq \lambda_n \delta \sqrt{sC_{\max}} + \delta\sqrt{sd} Z_1 + \sqrt{d} Z_{2q} + \delta\sqrt{sd} R_1 + \sqrt{d} R_2 && (12.103) \end{aligned}$$

where $Z_1 := \left\|\frac{1}{n}\Psi_S^\top W_n\right\|_\infty$ and $Z_{2q} := \left\|\frac{1}{n}\Psi_q^\top W_n\right\|_\infty$ are the Gaussian residuals, and $R_1 = \left\|\frac{1}{n}\Psi_S^\top V_n\right\|_\infty$, and $R_2 := \left\|\frac{1}{n}\Psi_q^\top V_n\right\|_\infty$ are the basis residuals.

From (12.78), we have $R_1, R_2 \le \sqrt{2}B^2 sd^{-3/2}$. Thus,

$$\begin{aligned}
\frac{1}{\lambda_n}\left[\delta\sqrt{sd}R_1 + \sqrt{d}R_2\right] &\le \frac{\sqrt{d}B'^2 sd^{-3/2}(\delta\sqrt{s}+1)}{\lambda_n} && (12.104)\\
&\le \frac{B'^2 d^{-1}s(\delta\sqrt{s}+1)}{\lambda_n} && (12.105)\\
&\to 0 && (12.106)
\end{aligned}$$

For the Gaussian residuals, proceeding as earlier,

$$\begin{aligned}
\mathbb{E}\left[Z_1\right] &\le 3\sqrt{\frac{\sigma^2\log(sd)C_{\max}}{n}} && (12.107)\\
\mathbb{E}\left[\sup_{q\in S^c} Z_{2q}\right] &\le 3\sqrt{\frac{\sigma^2\log(d(p-s))C_{\max}}{n}} && (12.108)
\end{aligned}$$

Thus,

$$\begin{aligned}
&\mathbb{P}\left[\frac{1}{\lambda_n}\sqrt{d}\left(\delta\sqrt{s}Z_1 + \sup_{q\in S^c} Z_{2q}\right) > \frac{\epsilon}{2}\right] && (12.109)\\
&\le \frac{\sqrt{d}}{\epsilon\lambda_n}\left(\delta\sqrt{s}\mathbb{E}\left[Z_1\right] + \mathbb{E}[\sup_{q\in S^c} Z_{2q}]\right) && (12.110)\\
&\le \frac{\sqrt{d}}{\epsilon\lambda_n}\left(\delta\sqrt{s}3\sqrt{\frac{\sigma^2\log(sd)C_{\max}}{n}} + 3\sqrt{\frac{\sigma^2\log(d(p-s))C_{\max}}{n}}\right)\\
&\le \frac{1}{\epsilon}\left[3\sqrt{\frac{\sigma^2 C_{\max}\delta^2 sd\log(sd)}{n\lambda_n^2}} + 3\sqrt{\frac{\sigma^2 C_{\max} d\log(d(p-s))}{n\lambda_n^2}}\right]\\
&\to 0 && (12.111)
\end{aligned}$$

Thus, $p\left(\sup_{q\in S^c}\|g_q\|_2 \le \sqrt{l_{\min}}\right) \to 1$, which proves the result. $\square$

Chapter 13

Concluding thoughts

This thesis is about using the tool of the Markov random field framework for the goal of prediction. To use a built graphical model for prediction is the task of inference. To build said graphical model from data comprises the tasks of feature estimation – which fixes the underlying form of the MRF – and structure and parameter learning – which then instantiates a particular MRF. The thesis proposes techniques for all three of those stages.

Inference: A primary inference task is to estimate the log partition function – the normalization constant of the graphical model distribution. For a discrete-valued model, this requires computing the sum of the unnormalized probabilities of exponentially many configurations. Graph-theoretic techniques such as clique-tree elimination reduce this complexity to being exponential only in the tree-width of the graph. This is small for sparse graphs such as trees, but even a 2D grid graph has a tree-width of $O(\sqrt{n})$. Computing the normalization constant is thus tractable only for sparse graphs. The projection paradigm starts off with this observation and reduces the log-partition function computation to an optimization problem: that of finding the "optimal" sparse graphical model from a candidate set of sparse models. The "optimal" model minimizes, over the candidate set of sparse models, a divergence measure with the given complex model. The divergence measure used is the KL divergence measure; however minimizing this over even simple graph classes like trees is difficult and typically a non-convex problem. Variational and free-energy based approximations [60, 68] tackle this by first approximating the KL divergence measure; and then minimizing this approximated divergence measure. We propose preconditioner approximations, which minimize, in place of the KL divergence measure, a "graphical model condition number" instead. Recent scientific computing developments have made possible the efficient computation of ultra-sparse preconditioners, which we are able to leverage for our preconditioner class of approximations.

Another key inference task is to compute the configuration with the most probability – the MAP configuration. The problem can be cast as an integer linear program; the relaxation of this to a linear program was analyzed in detail by [10] who gave rounding schemes with good approximation guarantees for specific feature functions and negative weights. For general problem settings, with general features functions, and general weights, [57] proposed tree-reweighted max product, which was showed to solve for the dual of the linear program under certain conditions. There is a lacuna with the LP relaxation approach however: the number of variables is quadratic $O(|E|K^2)$, where K is the number of labels, and $|E| = O(n^2)$ for dense graphs. For even small image applications, n is in the tens of thousands, and K is in the hundreds. The number of variables are so large as to necessitate the use of iterative algorithms to solve the LP, such as the message passing updates for tree-reweighted max-product. We propose formulating the MAP problem as a quadratic integer program instead; this reduces the number of variables from quadratic $O(|E|K^2)$ to linear $O(nK)$. We also show that relaxing the quadratic integer program to a quadratic program – with a simple quadratic objective and linear box constraints – does not introduce any gap, and that the relaxation is tight. That MAP is not in P suggests we would not always be able to tractably solve the QP – the reason is that the QP might be non-convex. To address this, we propose a convex approximation to the QP, which can be solved tractably; and for which we give an additive approximation guarantee. Finally, we show that the QP is equivalent to solving the MAP problem under a mean field relaxation of the marginal polytope; and that it can be extended to structured mean-field relaxations and other inner polytope approximations of the marginal polytope.

The third inference task this thesis is concerned with, is rigorous upper and lower bounds for general event probabilities. Not just inference, but also constant factor approximation guarantees for inference, have been shown to be intractable for general graphical models. In this thesis, we propose instead to provide interval guarantees: we provide the left and right intervals within which the true event probability is guaranteed to lie in. Variational methods do not provide much help here; they provide bounds for the partition function and very simple events like marginals. In variational Chernoff bounds, we extend the machinery of classical Chernoff bounds – which provide event probability bounds for i.i.d random variables – to the graphical model framework. At a high level, it involves using parameterized exponential family bounds for the event indicator function. An important subtask arises when we do not have complete distribitional information: we know only that the distribution belongs to a particular graphical model family (that is we know only the feature functions, not the parameters), as well as the expected values (moments) of a given set of functions. This arises naturally for instance in MLE parameter estimation, which involves matching moments to the empirical averages of the feature functions. We propose variational Chebyshev and Chebyshev-Chernoff bounds, which port the Chebyshev bound machinery to the graphical model framework. As it stands, the Chebyshev bound is distribution independent, but its dual optimizes over distributions satisfying the moment constraints; adding in a graphical model family constraint, and then taking its dual gives us the variational Chebyshev bounds.

Structure learning: This is the task of estimating the graph structure of the graphical model from i.i.d. samples. Score based approaches involve two components: a score metric, which is typically a sum of a goodness of fit measure of the graph to the data, and a graph complexity penalty; and a search procedure which searches through the candidate space of graphs with the goal to ouput the graph with the highest score. [12] show that the search procedure is hard; while for undirected models, even the score computation is hard since it involves computing the partition function. The score-based approaches are thus restricted to searching through very simple graph classes like trees, when used for learning undirected models. We seek to transform the search over the space of objects as structured as graphs, into a real-valued optimization problem; for this we need to transform the graph structure variable into a form more amenable to optimization. We thus propose edge-appearance relaxations, which parameterize the graph structure into an indicator vector over all node-pairs for an edge-set. We then show how a natural relaxation leads to a penalized likelihood maximization, with the penalty being a sum over edges of the ℓ_2 norms of the parameters of a single edge. For a model with a single parameter per edge, this leads to an ℓ_1 penalty on edge parameters. An Ising model is just such a model: the feature function over edges (s,t) is X_sX_t, and the edges (s,t) have a single parameter θ_{st}. We thus analyze ℓ_1 penalized maximum likelihood for the completely connected Ising model. But the likelihood of the completely connected Ising model involves the partition function – to finesse this, we propose, as in [38], to estimate the graph structure as a consistent union of the neighborhood of each node. The counterpart of ℓ_1 penalized maximum likelihood for estimating the local neighborhood of a node is ℓ_1 regularized logistic regression, of the node on the rest of the nodes; the support of the parameter vector so obtained gives the node neighborhood. We show that this procedure consistently recovers the graph structure even in high dimensional settings where the number of samples could even be logarithmic in the number of nodes.

Feature estimation: In this thesis, we also provide the tools to automate the feature estimation task in MRFs; normally the preserve of the MRF domain expert. We propose additive conditional random fields (aCRFs), a non-parametric extension of conditional random fields (CRFs) in which we learn the feature functions themselves from data. This is in contrast to learning from data the weights of given features in CRFs. We also propose sparse additive models (SpAM), a class of models which allow simultaneous predictor selection and feature estimation. For predictor selection in linear regression, ℓ_1 penalized linear regression, or lasso, has enjoyed good empirical success and has been shown to have strong theoretical properties [21, 72]. Nonparametric additive regression is a non-parametric extension of linear regression, where instead of learning the weights of linear features, we learn the additive functions themselves. In SpAM, we perform predictor selection in non-parametric (generalized) additive models in addition to learning the component functions; just as the lasso did for parametric linear regression. In fact, when the underlying component functions are linear, SpAM reduces to the lasso. We demonstrate excellent empirical behavior on some simulated as well as real data, and give a statistical analysis of the theoretical properties of the estimator that support its empirical behavior.

13.1 Future Work

For structure learning with given features (parametric graphical models), we proposed using ℓ_1 regularized regressions to recover the graph structure. We extended sparse generalized linear regression to the non-parametric setting with SpAM. It remains then to "complete the story" of "non-parametric graphical models" by using SpAM as the local sparse regressors to recover the graph structure. It also remains to complete the combining of structure and feature estimation for discriminative MRFs, by extending aCRFs to include learning the graph structure over the discrete label variables, and learning the additive features under sparsity assumptions.

Lateral extensions include learning features simultaneously across tasks, and formulating hierarchically structured features. The latter can be motivated by our own biology since humans are suspected to use hierarchical feature filters for various cognitive processes like vision.

13.2 Futurer work

Inspite of all recent advances in approximate inference for Markov random fields, it remains a fact that it is intractable to obtain arbitrarily close estimates for event probabilities. Until recently, even structure learning was formulated as an intractable problem, which was then approximated by heuristic search procedures. The recent explosion in the understanding of ℓ_1 penalties have allowed us to formulate a tractable structure learning task which, we show, consistently recovers the true graph structure. A further "organic" approach was reflected in our presented procedures for feature estimation; there we proposed additive subclasses of

Markov random fields where it is possible to estimate both features and the structure simultaneously. These raise the question of whether we can construct from the ground up a class of random fields that allow efficient inference in addition to efficient model creation. At the moment this niche has been occupied by simple graphical models like trees, which in turn have a strong model bias, which could manifestly not be reflected by the data and the domain. Note however, that “real world” biases need not take the form of simple models like trees. The entire latter part of this thesis, on structure and feature estimation, formulates and milks biases that make things tractable. We have seen how a sparsity bias allows us to estimate the graph structure tractably. Similarly, non-parametric techniques make smoothness assumptions that make feature estimation tractable. It is hoped that an increasing understanding of incorporating “real-world” biases into non-parametric approaches would allow us to construct such tractable random fields, and unify tractable inference with tractable model creation.

Bibliography

[1] Yasemin Altun, Thomas Hofmann, and Alexander J. Smola. Gaussian process classification for segmenting and annotating sequences. In *ICML-04, 21sth International Conference on Machine Learning*, 2004.

[2] Owe Axelsson and Vincent A. Barker. *Finite element solution of boundary value problems: theory and computation*. Society for Industrial and Applied Mathematics, 2001.

[3] Marshall Bern, John R. Gilbert, Bruce Hendrickson, Nhat Nguyen, and Sivan Toledo. Support-graph preconditioners. Submitted to *SIAM J. Matrix Anal. Appl.*, 2001.

[4] Dimitris Bertsimas and John Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

[5] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B*, 1986.

[6] Erik G. Boman and Bruce Hendrickson. Support theory for preconditioning. *SIAM Journal on Matrix Analysis and Applications*, 25, 2003.

[7] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[8] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11), 2001.

[9] K. P. Burnham and D. R. Anderson. *Model Selection and Multimodel Inference: A Practical-Theoretic Approach*. Springer-Verlag, 2002.

[10] Chandra Chekuri, Sanjeev Khanna, Joseph Naor, and Leonid Zosin. A linear programming formulation and approximation algorithms for the metric

labeling problem. *SIAM Journal on Discrete Mathematics*, 18(3):608–625, 2005.

[11] Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23(4):493–507, December 1952.

[12] D. Chickering. Learning Bayesian networks is NP-complete. *Proceedings of AI and Statistics*, 1995.

[13] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. Info. Theory*, 14(3):462–467, 1968.

[14] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory, IT-14(3)*, pages 462–467, 1968.

[15] Michael Collins. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP*, 2002.

[16] P. Dagum and M. Luby. Approximating probabilistic inference in bayesian belief networks is np-hard. *Artificial Intelligence 60(1)*, pages 141–153, 1993.

[17] S. Dasgupta. Learning polytrees. In *Uncertainty on Artificial Intelligence*, pages 134–14, 1999.

[18] P. Domingos. *What's Missing in AI: The Interface Layer, In P. Cohen (ed.), Artificial Intelligence: The First Hundred Years*. AAAI Press, Menlo Park, CA, 2006.

[19] D. Donoho and M. Elad. Maximal sparsity representation via ℓ_1 minimization. *Proc. Natl. Acad. Sci.*, 100:2197–2202, March 2003.

[20] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Ann. Statist.*, 28:337–407, 2000.

[21] E. Greenshtein and Y. Ritov. Persistency in high dimensional linear predictor-selection and the virtue of over-parametrization. *Journal of Bernoulli*, 10:971–988, 2004.

[22] D.M. Greig, B.T. Porteous, and A.H. Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society, Series B*, 51, 1989.

[23] K. Gremban. Combinatorial preconditioners for sparse, symmetric, diagonally dominant linear systems. *Ph.D. Thesis, Carnegie Mellon University, 1996*, 1996.

[24] Wolfgang Härdle, Marlene Müller, Stefan Sperlich, and Axel Werwatz. *Nonparametric and Semiparametric Models*. Springer-Verlag Inc, 2004.

[25] Trevor Hastie and Robert Tibshirani. *Generalized additive models*. Chapman & Hall Ltd., 1999.

[26] Trevor Hastie, Robert Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, 2001.

[27] A. Juditsky and A. Nemirovski. Functional aggregation for nonparametric regression. *Ann. Statist.*, 28:681–712, 2000.

[28] J. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric partitioning and markov random fields. *IEEE Symposium on the Foundations of Computer Science*, 1999.

[29] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *AISTATS*, 2005.

[30] M. Kudo, J. Toyama, and M. Shimbo. Multidimensional curve classification using passing-through regions. *Pattern Recognition Letters*, 20(11):1103–1111, 1999.

[31] P. Kumar, P.H.S. Torr, and A. Zisserman. Solving markov random fields using second order cone programming. *IEEE Conference of Computer Vision and Pattern Recognition*, 2006.

[32] Sanjiv Kumar and Martial Hebert. Discriminative fields for modeling spatial dependencies in natural images. In *Advances in Neural Information Processing Systems 16*, 2003.

[33] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289, 2001.

[34] John Lafferty, Xiaojin Zhu, and Yan Liu. Kernel conditional random fields: Representation and clique selection. In *Proceedings of ICML-04, 21st International Conference on Machine Learning*, 2004.

[35] S. Lauritzen. *Graphical Models*. Oxford Press, 1996.

[36] Yi Lin and Hao Helen Zhang. Component selection and smoothing in multivariate nonparametric regression. *Ann. Statist.*, 34(5):2272–2297, 2006.

[37] Andrew McCallum. Efficiently inducing features of conditional random fields. In *Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI03)*, 2003.

[38] N. Meinshausen and P. Bühlmann. High dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3), 2006.

[39] N. Meinshausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. Technical Report 720, Department of Statistics, UC Berkeley, 2006.

[40] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[41] Andrew Y. Ng. Feature selection, L1 vs. L2 regularization, and rotational invariance. In *21st International Conference on Machine Learning, ICML-04*, pages 404–412, 2004.

[42] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., 1988.

[43] David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. Table extraction using conditional random fields. In *Proceedings of the 26th Annual International ACM SIGIR conference on Research and Development in Informaion Retrieval*, pages 235–242. ACM Press, 2003.

[44] D. Pollard. *Convergence of stochastic processes*. Springer-Verlag, New York, 1984.

[45] Pradeep Ravikumar and John Lafferty. Variational Chernoff bounds for graphical models. *Proceedings of Uncertainty in Artificial Intelligence (UAI)*, 2004.

[46] Fei Sha and Fernando Pereira. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL*. Association for Computational Linguistics, 2003.

[47] P. Spirtes, C. Glymour, and R. Scheines. Causation, prediction and search. *MIT Press*, 2000.

[48] Nathan Srebro. Maximum likelihood bounded tree-width markov networks. *Artificial Intelligence*, 143:123–138, 2003.

[49] Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems 16*, 2003.

[50] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[51] P.H.S. Torr. Solving markov random fields using semidefinite programming. *Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[52] Joel A. Tropp. Just relax: Convex programming methods for identifying sparse signals. *IEEE Trans. Inform. Theory*, 51(3):1030–1051, March 2006.

[53] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Jour. Mach. Learn. Res.*, 6:1453–1484, 2005.

[54] P. M. Vaidya. Solving linear equations with symmetric diagonally dominant matrices by constructing good preconditioners. *Unpublished manuscript, UIUC*, 1990.

[55] Douglas L. Vail, John D. Lafferty, and Manuela M. Veloso. Conditional random fields for activity recognition. In *International Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, 2007.

[56] M. Wainwright. Sharp thresholds for high-dimensional and noisy recovery of sparsity. Technical Report 709, Department of Statistics, UC Berkeley, May 2006.

[57] M. Wainwright, T. Jaakkola, and A. Willsky. Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Transactions on Information Theory*, 51(11):3697–3717, November 2005.

[58] M. J. Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programs. In *Proc. Allerton Conference on Communication, Control and Computing*, October 2006.

[59] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Tree-reweighted belief propagation and approximate ML estimation by pseudo-moment matching. *9th Workshop on Artificial Intelligence and Statistics*, 2003.

[60] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *UC Berkeley, Dept. of Statistics, Technical Report 649*, 2003.

[61] M. J. Wainwright and M. I. Jordan. Variational inference in graphical models: The view from the marginal polytope. *Allerton Conference on Communication, Control, and Computing*, 2003.

[62] Martin Wainwright, Pradeep Ravikumar, and John Lafferty. High dimensional graphical model selection using l1-regularized logistic regression. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

[63] Martin J. Wainwright, Tommi S. Jaakkola, and Allan S. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation via pseudo-moment matching. In *Workshop on Artificial Intelligence and Statistics*, January 2003.

[64] Martin. J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. Technical report, University of California, Berkeley, January 2003. Department of Statistics, Technical Report 649.

[65] Martin J. Wainwright and Michael I. Jordan. Semidefinite relaxations for approximate inference on graphs with cycles. Technical report, University of California, Berkeley, September 2003. CS Division technical report UCB/CSD-03-1226.

[66] Yair Weiss and William T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, 47, 2001.

[67] C. Yanover and Y. Weiss. Finding the M most probable configurations using loopy belief propagation. In *Advances in Neural Information Processing Systems*, volume 16, 2003.

[68] J. S Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. *IJCAI 2001 Distinguished Lecture track*, 2001.

[69] Ming Yuan. Nonnegative garrote component selection in functional ANOVA models. In *Proceedings of AI and Statistics, AISTATS*, 2007.

[70] N. L. Zhang and D. Poole. A simple approach to bayesian network computations. *In Proc. of the Tenth Canadian Conference on Artificial Intelligence*, pages 171–178, 1994.

[71] P. Zhao and B. Yu. Model selection with the lasso. Technical report, UC Berkeley, Department of Statistics, March 2006. Accepted to Journal of Machine Learning Research.

[72] P. Zhao and B. Yu. On model selection consistency of lasso. *J. of Mach. Learn. Res.*, 7:2541–2567, 2007.

MACHINE LEARNING
DEPARTMENT

Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

Carnegie Mellon