Modeling and Estimation in Gaussian Graphical Models: Maximum-Entropy Methods and Walk-Sum Analysis

by

Venkat Chandrasekaran

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2007

Author . . . .
Department of Electrical Engineering and Computer Science
May 07, 2007

Certified by . . . .
Alan S. Willsky
Edwin Sibley Webster Professor of Electrical Engineering
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Modeling and Estimation in Gaussian Graphical Models: Maximum-Entropy Methods and Walk-Sum Analysis

by

Venkat Chandrasekaran

Submitted to the Department of Electrical Engineering and Computer Science
on May 07, 2007, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Graphical models provide a powerful formalism for statistical signal processing. Due to their sophisticated modeling capabilities, they have found applications in a variety of fields such as computer vision, image processing, and distributed sensor networks. In this thesis we study two central signal processing problems involving Gaussian graphical models, namely modeling and estimation. The modeling problem involves learning a sparse graphical model approximation to a specified distribution. The estimation problem in turn exploits this graph structure to solve high-dimensional estimation problems very efficiently.

We propose a new approach for learning a thin graphical model approximation to a specified multivariate probability distribution (e.g., the empirical distribution from sample data). The selection of sparse graph structure arises naturally in our approach through the solution of a convex optimization problem, which differentiates our procedure from standard combinatorial methods. In our approach, we seek the maximum entropy relaxation (MER) within an exponential family, which maximizes entropy subject to constraints that marginal distributions on small subsets of variables are close to the prescribed marginals in relative entropy. We also present a primal-dual interior point method that is scalable and tractable provided the level of relaxation is sufficient to obtain a thin graph. A crucial element of this algorithm is that we exploit sparsity of the Fisher information matrix in models defined on chordal graphs. The merits of this approach are investigated by recovering the graphical structure of some simple graphical models from sample data.

Next, we present a general class of algorithms for estimation in Gaussian graphical models with arbitrary structure. These algorithms involve a sequence of inference problems on tractable subgraphs over subsets of variables. This framework includes parallel iterations such as Embedded Trees, serial iterations such as block Gauss-Seidel, and hybrid versions of these iterations. We also discuss a method that uses local memory at each node to overcome temporary communication failures that may arise in distributed sensor network applications. We analyze these algorithms based on the recently developed walk-sum interpretation of Gaussian inference. We describe the walks "computed" by the algorithms using *walk-sum diagrams*, and show that for non-stationary iterations based on a very large and flexible set of sequences of subgraphs, convergence is achieved in walk-summable models. Consequently, we are free to choose spanning trees and subsets of variables adaptively at each iteration. This leads to efficient methods for optimizing the next iteration step to achieve maximum reduction in error. Simulation results demonstrate that these non-stationary algorithms provide a significant speedup in convergence over traditional one-tree and two-tree iterations.

Thesis Supervisor: Alan S. Willsky
Title: Edwin Sibley Webster Professor of Electrical Engineering

Acknowledgments

I am extremely fortunate to have Alan Willsky as my thesis advisor. Alan allowed me great freedom in exploring my interests, while at the same time providing guidance and direction on various research topics. During our discussions he posed an almost endless number of challenging and interesting research questions, and I continue to be amazed by his intellectual enthusiasm and curiosity. He always promptly reviewed (often over weekends, and sometimes on the same day!) drafts of our papers and this thesis with great care. I am also grateful for his advice on numerous aspects of academic life. Thanks, Alan, for this opportunity.

Al Oppenheim has been an excellent source of advice and encouragement over the last two years. I appreciate him for going far out of his way in organizing EGCS-3. I would also like to thank the other students in EGCS-3 for our weekly meetings, which helped me make a smooth transition to life at MIT.

I enjoyed collaborating with Jason Johnson on much of the research presented in this thesis. He taught me information geometry, and answered my many questions on the intricacies of graphical models. Pat Kreidl has been a good friend, and I enjoyed our conversations on research, politics, and food. Dmitry Malioutov deserves a special mention for his great sense of humor, and also for our numerous research discussions. I was lucky to have Erik Sudderth as an officemate during my first year at MIT. He helped clarify various points about the Embedded Trees algorithm and graphical models in general. Jin Choi and I entered MIT at the same time, and have shared an office for most of our time here. I would like to thank her for many interesting conversations, both research and otherwise. Sujay Sanghavi was very willing to discuss my half-baked ideas. Jason Williams provided useful references on combinatorial algorithms, and was an excellent and patient 6.432 TA. I have also benefited from discussions with Al Hero and Devavrat Shah about various aspects of this thesis, and I look forward to fruitful future collaborations with them.

Finally, I would like to thank my family for their love, support, and encouragement all these years.

Contents

List of Figures

Chapter 1

Introduction

Statistical signal processing plays an important role in a variety of applications including medical imaging [55], speech processing [17], financial data analysis [19], image and video processing [44, 83], and wireless communications [64]. Analyzing and processing signals based on their stochastic properties involves several important problems such as *generating* signal samples from a random ensemble, *modeling* the probabilistic behavior of signals based on empirically obtained statistics, and *processing* noise-corrupted signals to extract useful information. This thesis deals with the second and third problems using the framework of graph-structured probabilistic models, also known as graphical models.

Graphical models provide a powerful formalism for statistical signal processing. They offer a convenient representation for joint probability distributions and convey the Markov structure in a large number of random variables compactly. A graphical model [47, 53] is a collection of variables defined with respect to a graph; each vertex of the graph is associated with a random variable and the edge structure specifies the conditional independence (Markov) properties among the variables. An important feature of graphical models is that they can be used to succinctly specify global distributions over a large collection of variables in terms of local interactions, each involving only a small subset of variables.

Due to their sophisticated modeling capabilities, graphical models (also known as Markov random fields or MRFs) have found applications in a variety of fields including distributed processing using sensor networks [21], image processing [34, 83, 84], computer vision [75], statistical physics [61], and coding theory [58]. Our focus is on the important class of Gaussian graphical models, also known as Gauss-Markov random fields (GMRFs), which have been widely used to model natural phenomena in many large-scale applications [30, 67].

The modeling problem for graphical models essentially reduces to learning the graph (Markov) structure of a set of variables given an empirical distribution on those variables. Exploiting this graph structure is also critical in order to efficiently solve the Gaussian estimation problem of denoising a signal corrupted by additive noise. Due to the widespread use of normally distributed random variables as both prior and noise models, both the Gaussian modeling and estimation problems are of great importance. Solving these problems efficiently forms the focus of this thesis.

Both the modeling and estimation problems can be efficiently solved for tree-structured graphical models (i.e., graphs with no cycles, also called treewidth-1 graphs[1]). Finding the best tree-structured approximation (in the sense of Kullback-Leibler divergence) to a specified empirical distribution can be solved using a suitable maximum spanning tree formulation [16]. For the estimation problem in tree-structured MRFs, Belief Propagation (BP) [62] provides an efficient linear complexity algorithm to compute exact estimates. However, tree-structured Gaussian processes possess limited modeling capabilities, leading to blocky artifacts in the resulting covariance approximations [74]. In order to model a richer class of statistical dependencies among variables, one often requires loopy graphical models.

The situation is more complicated for both modeling and inference when the graphical models involved contain cycles. Indeed, the general graphical model selection problem of finding the best treewidth-k graphical model approximation to a specified distribution is NP-hard for $k > 1$ [49]. For the Gaussian estimation problem, computing the Bayes least-squares estimate is equivalent to solving a linear system of equations specified in terms of the information-form parameters of the conditional distribution. Due to its cubic computational complexity in the number of variables, direct matrix inversion to solve the Gaussian estimation problem is intractable in many applications in which the number of variables is very large (e.g., in oceanography problems [30] the number of variables may be on the order of 10^6).

In this thesis, we describe tractable methods to solve both these problems when the graphical models involved contain cycles.

1.1 Contributions

1.1.1 Learning Markov Structure using Maximum Entropy Relaxation

The problem of learning the Markov structure of a probability distribution has been extensively studied from the point of view of solving a combinatorial optimization problem [5, 16, 26, 49, 51, 59]. Given a distribution p^* (for example, an empirical distribution obtained from data samples), one searches over a collection of graphs in order to identify a simple graph that still provides a good approximation to p^* in the sense of Kullback-Leibler divergence. In essence, this involves projecting the distribution to each candidate graph (minimizing information divergence) and picking the closest one. Previous work has focussed on this problem for families of triangulated graphical models with bounded treewidth. In order to solve this problem using a polynomial-time algorithm, several approximate algorithms have been studied [49, 59]. These algorithms restrict the search space to subgraphs of a *given* treewidth-k graph rather than searching over all possible treewidth-k graphs. Another restriction with these methods [16, 49, 59] is that they focus on chordal graphs due to the fact the projection onto a chordal graph has a simple solution. In any case, only heuristic methods are permitted because the general graphical model selection problem is NP-hard.

[1]All the technical terminology used in this thesis is explained in detail in Chapter 2.

We propose a novel approach to solve the graphical model selection problem using a convex program as opposed to a combinatorial approach. Our formulation is motivated by the maximum entropy (ME) principle [20, 42]. The ME principle states that subject to linear constraints on a set of statistics, the entropy-maximizing distribution among *all* distributions lies in the exponential family based on those statistics used to define the constraints. Loosely, this suggests that entropy, when used as maximizing objective function, implicitly favors Markov models that possess as few conditional dependencies as possible while still satisfying the constraints. Proceeding with this point of view, we propose a maximum entropy relaxation (MER) problem in which linear constraints on marginal moments are replaced by a set of nonlinear, convex constraints that enforce closeness to the marginal moments of p^* in the sense of information divergence. Roughly speaking, we expect that when p^* is close to a family of Markov models defined on some graph, the MER problem will automatically "thin" the model, i.e., the relaxed probability distribution will be Markov on that graph. Hence, the MER problem automatically checks to see if there are any nearby lower-order Markov families without explicitly projecting onto a collection of candidate graphs. Thus, the formulation is not restricted in any manner to "search" only over subgraphs of a specified graph.

To solve the MER problem, we develop a scalable algorithm that exploits sparse computations on chordal graphs. This algorithm actually solves a sequence of MER problems based on subsets of the constraints. At each step of the procedure, we add more active constraints (the ones which have the largest constraint violation) until all the constraints that were omitted are found to be inactive. Each MER sub-problem may be formulated with respect to a chordal graph which supports the current constraint set. We solve these sub-problems using a primal-dual interior point method that exploits sparsity of the Fisher information matrix over chordal graphs. Very importantly, this incremental approach to solution of MER still finds the global MER solution in the complete model, but in a manner which exploits sparsity of the MER solution. We emphasize here that while our approach takes advantage of efficient computations with respect to chordal graphs, the solution to the MER problem can still be a non-chordal graph.

While our focus in this thesis is on the Gaussian model selection problem, our framework applies equally well to the case of discrete MRFs. Simulation results show that the underlying graph structure is recovered with few spurious or missing edges even with a moderate number of samples.

1.1.2 Estimation Algorithms based on Tractable Subgraphs: A Walk-Sum Analysis

Considerable effort has been and still is being put into developing estimation algorithms for graphs with cycles, including a variety of methods that employ the idea of performing inference computations on tractable subgraphs [68, 79]. The recently proposed Embedded Trees (ET) iteration [73, 74] is one such approach that solves a sequence of inference problems on trees or, more generally, tractable subgraphs. If ET converges, it yields the correct conditional estimates, thus providing an effective inference algorithm for graphs with essentially arbitrary structure.

For the case of *stationary* ET iterations — in which the same tree or tractable subgraph is used at each iteration — necessary and sufficient conditions for convergence are provided in [73, 74]. However, experimental results in [73] provide compelling evidence that much faster convergence can often be obtained by changing the embedded subgraph that is used from one iteration to the next. The work in [73] provided very limited analysis for such *non-stationary* iterations, thus leaving open the problem of providing easily computable broadly applicable conditions that guarantee convergence.

In related work that builds on [74], Delouille et al. [24] describe a stationary block Gauss-Jacobi iteration for solving the Gaussian estimation problem with the added constraint that messages between variables connected by an edge in the graph may occasionally be "dropped". The local blocks (subgraphs) are assumed to be small in size. Such a framework provides a simple model for estimation in distributed sensor networks where communication links between nodes may occasionally fail. The proposed solution involves the use of memory at each node to remember past messages from neighboring nodes. The values in this local memory are used if there is a breakdown in communication to prevent the iteration from diverging. However, the analysis in [24] is also restricted to the case of stationary iterations, in that the same partitioning of the graph into local subgraphs is used at every iteration.

Finally, we note that ET iterations fall under the class of *parallel* update algorithms, in that every variable must be updated in an iteration before one can proceed to the next iteration. However, *serial* schemes involving updates over subsets of variables also offer tractable methods for solving large linear systems [38, 76]. An important example in this class of algorithms is block Gauss-Seidel (GS) in which each iteration involves updating a small subset of variables.

In this thesis, we analyze non-stationary iterations based on an arbitrary sequence of embedded trees or tractable subgraphs. We present a general class of algorithms that includes the non-stationary ET and block GS iterations, and provide a general and very easily tested condition that guarantees convergence for any of these algorithms. Our framework allows for hybrid non-stationary algorithms that combine aspects of both block GS and ET. We also consider the problem of failing links and describe a method that uses local memory at each node to address this problem in general non-stationary parallel and serial iterations.

Our analysis is based on a recently introduced framework for interpreting and analyzing inference in GMRFs based on sums over walks in graphs [56]. We describe *walk-sum diagrams* that provide an intuitive interpretation of the estimates computed by each of the algorithms after every iteration. A walk-sum diagram is a graph that corresponds to the walks "accumulated" after each iteration. As developed in [56] walk-summability is an easily tested condition which, as we will show, yields a simple necessary and sufficient condition for the convergence of the algorithms. As there are broad classes of models (including attractive, diagonally-dominant, and so-called pairwise-normalizable models) that are walk-summable, our analysis shows that our algorithms provide a convergent, computationally attractive method for inference.

The walk-sum analysis and convergence results show that non-stationary iterations of our algorithms based on a very large and flexible set of sequences of subgraphs or subsets of variables converge in walk-summable models. Consequently, we are free to use any sequence of trees in the ET algorithm or any sequence of subsets of variables in the block

GS iteration, and still achieve convergence in walk-summable models. We exploit this flexibility by choosing trees or subsets of variables adaptively to minimize the error at iteration n based on the residual error at iteration $n-1$. To make these choices optimally, we formulate combinatorial optimization problems that maximize certain re-weighted walk-sums. We describe efficient methods to solve relaxed versions of these problems. For the case of choosing the “next best” tree, our method reduces to solving a maximum-spanning tree problem. Simulation results indicate that our algorithms for choosing trees and subsets of variables adaptively provide a significant speedup in convergence over traditional approaches involving a single subgraph or alternating between two subgraphs.

Our walk-sum analysis also shows that local memory at each node can be used to achieve convergence for any of the above algorithms when communication failures occur in distributed sensor networks. Our protocol differs from the description in [24], and as opposed to that work, allows for non-stationary updates. Also, our walk-sum diagrams provide a simple, intuitive representation for the propagation of information with each iteration.

1.2 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 provides a brief but self-contained background on graphical models, exponential families, and walk-sums. In Chapter 3, we discuss the MER formulation to learn the Markov structure in a collection of variables. Chapter 4 describes a rich class of algorithms for Gaussian estimation, and analyzes the convergence of these algorithms in walk-summable models. In both Chapter 3 and Chapter 4, simulation results are included to demonstrate the effectiveness of our methods. We conclude with a brief discussion and mention possible future research directions resulting from this thesis. The appendices provide additional details and proofs.

The research and results presented in Chapter 3 and Chapter 4 have been submitted previously as a conference paper [45] and a journal article [14] respectively.

Chapter 2

Background

This chapter provides a brief but self-contained background about the various technical aspects of this thesis. We begin by presenting some basics from graph theory and graphical models, before moving on to exponential family distributions, and concluding with the walk-sum interpretation of Gaussian estimation. In each section, we provide a list of references for readers who are interested in learning more about these topics.

2.1 Graph theory

We present some basic concepts from graph theory that will be useful throughout this thesis. For more details, we refer the reader to [9, 27, 46].

A *graph* $\mathcal{G} = (V, \mathcal{E})$ consists of a set of *vertices* or *nodes* V and associated *edges* $\mathcal{E} \subset \binom{V}{2}$ that link vertices together. Here, $\binom{V}{2}$ represents the set of all unordered pairs of vertices. An *edge* between nodes s and t is denoted by $\{s, t\}$. The *degree* of a vertex is the number of edges incident to it. Two vertices are said to be *neighbors* if there is an edge between them.

A *subgraph* $\mathcal{S}$ of $\mathcal{G} = (V, \mathcal{E})$ is any graph whose vertex set is $V' \subseteq V$, and whose edge set $\mathcal{E}'$ is a subset $\mathcal{E}' \subseteq \mathcal{E}(V')$ where

$$\mathcal{E}(V') \triangleq \{\{s,t\} \mid \{s,t\} \in \mathcal{E},\ \ s,t \in V'\}\,.$$

A subgraph is said to be *spanning* if $V' = V$. An *induced subgraph* $\mathcal{S}(V')$ is a subgraph with vertices V' and edges $\mathcal{E}' = \mathcal{E}(V')$. A *supergraph* $\mathcal{H}$ of $\mathcal{G}$ is any graph whose vertex set V' is a superset $V' \supseteq V$, and whose edge set $\mathcal{E}'$ is a superset $\mathcal{E}' \supseteq \mathcal{E}$.

A *path* $u_0 \cdots u_k$ between two vertices u_0 and u_k in $\mathcal{G}$ is a sequence of distinct vertices $\{u_i\}_{i=0}^{k}$ such that there exists an edge between each successive pair of vertices, i.e., $\{u_i, u_{i+1}\} \in \mathcal{E}$ for $i = 0, \ldots, k-1$. A subset $S \subset V$ is said to *separate* subsets $A, B \subset V$ if every path in $\mathcal{G}$ between any vertex in A and any vertex in B passes through a vertex in S.

A graph is said to be *connected* if there exists a path between every pair of vertices. A *clique* is a fully connected subgraph, i.e., a subgraph in which each vertex is linked to every other vertex by an edge. A clique is *maximal* if it is not contained as a proper subgraph of any other clique.

A *cycle* is the concatenation of a path $u_0 \cdots u_k$ with the vertex u_0 such that $\{u_k, u_0\} \in \mathcal{E}$. A *tree* is a connected graph that contains no cycles. The number of edges in a tree-structured graph is one less than the number of vertices. A *forest* is a graph, not necessarily connected, that contains no cycles. Trees and forests form an important class of graphs and play a central role in the estimation algorithms discussed in this thesis.

Chordal graphs and Junction trees We now describe the class of chordal graphs, and the junction tree representation for such graphs. These concepts are critical to the development of efficient algorithms to solve the maximum-entropy relaxation problem for model selection.

A graph is said to be *chordal* or *triangulated* if every cycle of length greater than three in the graph contains an edge between non-neighboring vertices in the cycle. A special representation for a chordal graph can be specified in terms of the maximal cliques of the graph.

Definition 2.1 *Let $\mathcal{C}$ be the set of maximal cliques in a connected graph $\mathcal{G}$. A* junction tree *representation of $\mathcal{G}$ is a tree, with the nodes being the elements of $\mathcal{C}$, which satisfies the following* running intersection *property: For every pair of nodes (cliques) C_i and C_j in the junction tree, every node (clique) in the unique path between C_i and C_j contains $C_i \cap C_j$.*

As the following theorem proves, valid junction trees can only be defined for chordal graphs.

Theorem 2.1 *A graph is chordal if and only if it has a junction tree representation.*

Proof: See [46].

In general, a chordal graph may have multiple junction tree representations. However, there is a certain uniqueness about these representations. Each edge $\{C_i, C_j\}$ in the junction tree is labeled by the intersection of the cliques $C_i \cap C_j$. The running intersection property ensures that this intersection is non-empty. These edge intersections are called *separators*, and there are $|\mathcal{C}| - 1$ separators.

Theorem 2.2 *For a chordal graph $\mathcal{G}$, the set $\mathcal{S}$ of separators, with multiplicity, is the same for any junction tree representation.*

Proof: See [43].

Thus, without loss of generality, we can refer to *the* junction tree representation of a chordal graph.

The *treewidth* of a chordal graph is one less than the cardinality of the largest clique in the junction tree representation. A graph $\mathcal{G}$ is said to be *thin* if the smallest chordal supergraph of $\mathcal{G}$ (i.e., one with the least number of extra edges) has small treewidth.

The reason that chordal graphs and junction tree representations are important is that one can make strong, precise statements about the factorization of probability distributions defined over chordal graphs based on local marginal distributions on the cliques and separators in the junction tree representation. This leads to analytical formulas for the computation of the entropy and other quantities of distributions defined on chordal graphs. We discuss these points in the later sections of this chapter and in Chapter 3 of this thesis.

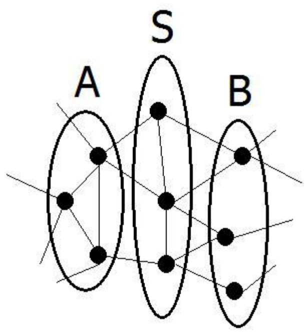


Figure 2-1: Illustration of Markov condition: The variables x_{A} are independent of x_{B} conditioned on the variables x_{S} because the subset of variables S separates A and B.

2.2 Graphical models

2.2.1 Definition and Markovianity

A *graphical model* [46, 47, 53] is a collection of random variables indexed by the vertices of a graph $\mathcal{G} = (V, \mathcal{E})$; each vertex $s \in V$ corresponds to a random variable x_s, and where for any $A \subset V$, $x_A = \{x_s | s \in A\}$. The models that we consider in this thesis are defined with respect to undirected graphs; we note that models defined on directed graphs can be converted to models on undirected graphs with some loss in structure [62].

Definition 2.2 *A distribution $p(x_V)$ is* Markov *with respect to a graph $\mathcal{G} = (V, \mathcal{E})$ if for any subsets $A, B \subset V$ that are separated by some $S \subset V$, the subset of variables x_A is conditionally independent of x_B given x_S, i.e. $p(x_A, x_B | x_S) = p(x_A|x_S) \cdot p(x_B|x_S)$.*

In this manner, graphical models generalize the concept of Markov chains, and are thus also referred to as *Markov random fields* (MRFs). Figure 2-1 provides a simple example. Any distribution p defined with respect to the graph in the figure must satisfy the condition that $p(x_A, x_B|x_S) = p(x_A|x_S) \cdot p(x_B|x_S)$. Note that this is only one of several conditional independence relations (implied by the graph structure) that p must satisfy in order to be Markov with respect the graph.

A distribution being Markov with respect to a graph implies that it can be decomposed into local functions in a very particular way. The following fundamental theorem precisely relates these two notions of Markovianity and local factorization.

Theorem 2.3 Hammersley-Clifford [53]: *Let $p(x_V) > 0$ be a strictly positive distribution that is Markov with respect to graph $\mathcal{G} = (V, \mathcal{E})$. Then,*

$$p(x_V) = \prod_{s \in V} \psi_s(x_s) \prod_{E \in \mathcal{E}} \psi_E(x_E), \tag{2.1}$$

where each $\psi_E(x_E)$ is a local function that depends only on the variables x_E, and each $\psi_s(x_s)$ depends only on variable x_s. Conversely, if $p(x_V)$ is any distribution that factorizes according to (2.1), then $p(x_V)$ is Markov with respect to $\mathcal{G}$.

The functions $\psi_s(x_s)$ and $\psi_E(x_E)$ in (2.1) are called potential functions. This theorem illustrates an important feature of graphical models: Global probability distributions involving a very large number of variables can be defined in a consistent manner using only local functions that summarize interactions among small subsets of variables.

Junction-tree factorization For distributions defined on chordal graphs, the global distribution can be factored in terms of local marginal distributions [46].

Theorem 2.4 *Let $p(x_V)$ be a distribution that is Markov with respect to a chordal graph $\mathcal{G} = (V, \mathcal{E})$. Let $\mathcal{C}$ and $\mathcal{S}$ be the cliques and separators respectively in the junction-tree representation of $\mathcal{G}$. Then,*

$$p(x_V) = \frac{\prod_{C \in \mathcal{C}} p(x_C)}{\prod_{S \in \mathcal{S}} p(x_S)}, \tag{2.2}$$

where each $p(x_C)$ is a marginal distribution over the subset of variables C, and each $p(x_S)$ is a marginal distribution over the subset S.

This decomposition into marginal distribution functions plays an important role in the solution of the maximum entropy relaxation framework for model selection presented in Chapter 3.

2.2.2 Gaussian graphical models

We consider Gaussian graphical models, also known as Gauss-Markov random fields (GMRFs), $\{x_s | s \in V\}$ parameterized by a mean vector μ and a symmetric, positive-definite covariance matrix P (denoted by $P \succ 0$): $x_V \sim \mathcal{N}(\mu, P)$ [53, 71]. Each x_s is assumed (for simplicity) to be a scalar variable. Thus, the distribution is specified as follows:

$$p(x_V) = \frac{1}{(2\pi \cdot \det P)^{\frac{|V|}{2}}} \exp\left\{-\frac{1}{2}(x_V - \mu) P^{-1} (x_V - \mu)^T\right\}, \tag{2.3}$$

where $\det P$ denotes the determinant of the matrix P [40].

An alternate natural parameterization for GMRFs is specified in terms of the *information matrix* $J = P^{-1}$ and *potential vector* $h = P^{-1}\mu$, and is denoted by $x_V \sim \mathcal{N}^{-1}(h, J)$:

$$p(x_V) \propto \exp\left\{-\frac{1}{2} x_V^T J x_V + h^T x_V\right\}. \tag{2.4}$$

This parameterization is known as the *information form* representation. The importance of this alternate representation is two-fold. First, Gaussian priors in many applications are specified in the information form, such as the thin-membrane and thin-plate models used in image processing [83]. Second, the specialization of the Hammersley-Clifford theorem (Theorem 2.3) to the case of Gaussian graphical models provides an explicit connection between the sparsity of J and the graph with respect to which the distribution is Markov.

Theorem 2.5 *Let $x_V \sim \mathcal{N}^{-1}(h, J)$ be a collection of Gaussian random variables, with joint distribution $p(x_V)$ defined according to (2.4). Let this distribution be Markov with*

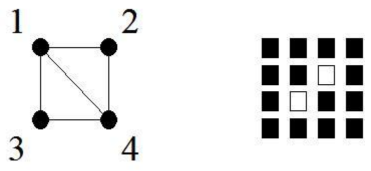


Figure 2-2: Gauss-Markov random field sparsity example: The graph on the left serves as the underlying model. The matrix on the right represents the sparsity pattern of the information matrix J of the corresponding distribution with solid squares representing non-zero entries and empty squares representing zero entries. The nodes 2 and 3 are not connected by an edge; hence, the corresponding entries in the J matrix are zero.

respect to graph $\mathcal{G} = (V, \mathcal{E})$. Then, $J_{s,t} \neq 0$ if and only if the edge $\{s, t\} \in \mathcal{E}$ for every pair of vertices $s, t \in V$.

Proof: See [71].

The example in Figure 2-2 provides a simple illustration of this theorem.

Interpreted in a different manner, the elements of the information matrix J are also related to so-called *partial correlation coefficients*. The partial correlation coefficient $\rho_{t,s}$ is the correlation coefficient of variables x_t and x_s conditioned on knowledge of all the other variables [53]:

$$\rho_{t,s} \triangleq \frac{\text{cov}(x_t; x_s | x_{\setminus t,s})}{\sqrt{\text{var}(x_t | x_{\setminus t,s}) \text{var}(x_s | x_{\setminus t,s})}} = -\frac{J_{t,s}}{\sqrt{J_{t,t} J_{s,s}}}. \tag{2.5}$$

Hence, $J_{t,s} = 0$ implies that x_t and x_s are conditionally independent given all the other variables $x_{\setminus t,s}$.

2.3 Exponential families

We describe an important class of probability distributions known as exponential families [15]. These families possess a very rich and elegant geometric structure, and the associated tools that exploit this structure fall in the area of information geometry [3, 22]. Exponential families have played a significant role in the development of new lines of research in the graphical models community [80].

2.3.1 Definition

Let $\mathbb{X}$ be either a continuous or discrete sample space. We consider parametric families of probability distributions with support $\mathbb{X}^{|V|}$ defined by

$$p_\theta(x) = \exp\{\theta^T \phi(x) - \Phi(\theta)\}, \tag{2.6}$$

where $\phi : \mathbb{X}^{|V|} \rightarrow \mathbb{R}^d$ are the *sufficient statistics*, θ are the *exponential parameters*, and $\Phi(\theta) = \log \int \exp(\theta^T \phi(x))dx$ is the *cumulant generating function* [1](also known as the *log-partition function*). The family is defined by the set $\Theta \subset \mathbb{R}^d$ of all normalizable θ:

$$\Theta \triangleq \left\{\theta \in \mathbb{R}^d : \Phi(\theta) < \infty\right\}$$

The connection to graphical models is established by the fact that the statistics $\phi(x)$ are usually features over small subsets of variables, for example $\phi(x) = \left\{x_s x_t : \{s,t\} \in \binom{V}{2}\right\}$. More precisely, the exponential family distributions that we consider are related to so-called Gibbs distributions [35] in that the statistics ϕ are functions over the edges $\mathcal{E}$ and the vertices V. Thus, by virtue of the exponential, the focus shifts from a product of local edge-wise potentials as in (2.1) to a linear combination of features over edges. This leads directly to a specialization of the Hammersley-Clifford theorem that relates the Markovianity of a distribution p_θ to the sparsity of the corresponding exponential parameter θ.

Theorem 2.6 *Let x_V be some collection of variables, and let*

$$\phi(x) = \{\phi_s(x_s) : s \in V\} \bigcup \left\{\phi_E(x_E) : E \in \binom{V}{2}\right\}$$

be some set of statistics such that each $\phi_E(x_E)$ and $\phi_s(x_s)$ depend only on the corresponding subsets of variables x_E and x_s respectively. Then, $p_\theta(x)$ is Markov with respect to the graph $\mathcal{G} = (V, \mathcal{E})$ for some $\mathcal{E} \subset \binom{V}{2}$ if and only if the collection of exponential parameters $\left\{\theta_E : E \in \binom{V}{2} \backslash \mathcal{E}\right\}$ is zero.

The statistics ϕ are *minimal* if they are linearly independent. We note here that linearly dependent statistics have also played a role in recent work [78, 79]. An exponential family is said to be *regular* if Θ is a non-empty open set in $\mathbb{R}^d$. Our focus here is on *marginalizable* exponential families, in which the marginal distribution $p(x_S) = \int_{x_{V \backslash S}} p_\theta(x)dx$ for $S \subset V$ is also an exponential family distribution based on the subset of statistics $\phi_{\bar{S}}(x_S) \triangleq \{\phi_{S'}(x_{S'}) : S' \subseteq S\}$ with support inside S. Letting ζ be the exponential parameter of $p(x_S)$, we have that $p(x_S) = p_\zeta(x_S) \propto \exp\{\zeta^T \phi_{\bar{S}}(x_S)\}$. Note that $\zeta \neq \theta_{\bar{S}}$ in general, where $\theta_{\bar{S}}$ is defined analogous to $\phi_{\bar{S}}$.

2.3.2 Log-partition function and moment parameterization

The log-partition function $\Phi(\theta)$ possesses a number of important properties [3, 80]. We begin with the following theorem, which justifies the use of the term cumulant-generating function for $\Phi(\theta)$.

Theorem 2.7 *Let $\{\phi_\alpha(x)\}$ be a collection of statistics, and let $\theta = \{\theta_\alpha\}$ be the corresponding exponential parameters of an exponential family of distributions. The derivatives*

[1]Since the focus of this thesis is on continuous Gaussian models, we use integrals. These must be replaced by sums for discrete models.

of the log-partition function $\Phi(\theta)$ can be computed as follows:

$$\begin{aligned}\frac{\partial \Phi(\theta)}{\partial \theta_\alpha} &= \mathbb{E}_{p_\theta}\left\{\phi_\alpha(x)\right\} \\ \frac{\partial^2 \Phi(\theta)}{\partial \theta_\alpha \partial \theta_\beta} &= \mathbb{E}_{p_\theta}\left[(\phi_\alpha(x) - \mathbb{E}_{p_\theta}\{\phi_\alpha(x)\})(\phi_\beta(x) - \mathbb{E}_{p_\theta}\{\phi_\beta(x)\})\right].\end{aligned}$$

The second derivative corresponds to the entry $\mathbb{E}_{p_\theta}\left\{-\frac{\partial^2 \log p_\theta(x)}{\partial\theta_\alpha\partial\theta_\beta}\right\}$ of the Fisher information matrix with respect to the exponential parameters θ.

Proof: See [3].

The Fisher information matrix [69] with respect to the θ parameters is positive-definite for a minimal set of statistics. In other words, the Hessian of $\Phi(\theta)$ is positive-definite.

Corollary 2.1 *Let $\{\phi_\alpha(x)\}$ be a collection of minimal statistics, and let $\theta = \{\theta_\alpha\}$ be the corresponding exponential parameters of an exponential family of distributions. Then, the log-partition function $\Phi(\theta)$ is strictly convex.*

The convexity of the log-partition function plays a critical role in the development of an alternate parameterization for exponential family distributions. This is achieved through the following Legendre transformation [65]:

$$\Psi(\eta) = \sup_\theta\{\eta^T\theta - \Phi(\theta)\}. \tag{2.7}$$

The vector η has the same dimension as θ. By definition [65], we have that $\Psi(\eta)$ is also a strictly convex function of η.

In this maximization, one can differentiate the right-hand-side and check that the optimal θ^*, if it exists[2], is the exponential parameter such that:

$$\mathbb{E}_{p_{\theta^*}}\left\{\phi(x)\right\} = \eta. \tag{2.8}$$

The vector η specifies the *moment parameterization* of the exponential family. Not every $\eta \in \mathbb{R}^d$ can be realized as the expectation of the statistics ϕ with respect to some parameter $\theta \in \Theta$. Thus, we have the following definition for the set of realizable moment parameters:

$$\mathcal{M} = \left\{\eta \in \mathbb{R}^d : \exists\theta \in \Theta \text{ such that } \mathbb{E}_{p_\theta}\left\{\phi(x)\right\} = \eta\right\}. \tag{2.9}$$

Let $p_\theta(x)$ be an exponential family distribution with moment parameters $\mathbb{E}_{p_\theta}\left\{\phi(x)\right\} = \eta$ in a marginalizable exponential family. One can check that the moments of the marginal distribution $p_\zeta(x_S) \propto \exp\{\zeta^T\phi_{\bar{S}}(x_S)\}$ of subset S are determined by the corresponding subset of moments $\eta_{\bar{S}} \triangleq \{\eta_{S'} : S' \subseteq S\}$ with support inside S.

For exponential families with minimal statistics ϕ there exists a bijective map $\Lambda : \Theta \to \mathcal{M}$ that converts exponential parameters to moment parameters:

$$\Lambda(\theta) = \mathbb{E}_{p_\theta}\left\{\phi(x)\right\}. \tag{2.10}$$

[2]If there exists no finite optimal θ^*, the value of $\Psi(\eta)$ is taken to be ∞.

Thus, the optimal θ in (2.7) is given by $\Lambda^{-1}(\eta)$. Since the map Λ is bijective, each $\eta \in \mathcal{M}$ uniquely specifies a distribution parameterized by the exponential parameter $\theta = \Lambda^{-1}(\eta)$.

Due to the convexity of $\Psi(\eta)$ with respect to η, the Legendre transformation can also be applied to $\Psi(\eta)$ to recover $\Phi(\theta)$ for exponential families with minimal statistics:

$$\Phi(\theta) = \sup_{\eta}\{\theta^T \eta - \Psi(\eta)\}, \tag{2.11}$$

with the optimal value, if it exists, being attained when $\eta = \Lambda(\theta)$. Analogous to Theorem 2.7, we have the following result for the derivatives of $\Psi(\eta)$.

Theorem 2.8 *Let $\{\phi_\alpha(x)\}$ be a collection of minimal statistics, and let $\theta = \{\theta_\alpha\}$ be the corresponding exponential parameters of an exponential family of distributions. The first derivative of the dual function $\Psi(\eta)$ of the log-partition function $\Phi(\theta)$ can be computed as follows:*

$$\frac{\partial \Psi(\eta)}{\partial \eta_\alpha} = \Lambda^{-1}(\eta)_\alpha.$$

The second derivative $\frac{\partial^2 \Psi(\eta)}{\partial \eta_\alpha \partial \eta_\beta}$ corresponds to the (α, β) entry of the Fisher information matrix with respect to the moment parameters η.

To simplify notation, we refer to the moment parameter corresponding to an exponential parameter θ by $\eta(\theta)$, and the exponential parameter corresponding to a moment parameter η by $\theta(\eta)$. Letting $G(\theta)$ and $G^*(\eta)$ denote the Fisher information matrices with respect to the exponential and moment parameterizations, we have that [3]

$$G^*(\eta) = G(\theta(\eta))^{-1}. \tag{2.12}$$

Interpretation of $\Psi(\eta)$ Evaluating the function $\Psi(\eta)$, we have that

$$\begin{aligned}\Psi(\eta) = \Psi(\eta(\theta)) &= \eta(\theta)^T\theta - \Phi(\theta) \\ &= \int p_\theta(x)\left[\phi(x)^T\theta - \Phi(\theta)\right] dx \\ &= \mathbb{E}_{p_\theta}\{\log p_\theta(x)\}.\end{aligned}$$

Hence, $\Psi(\eta)$ is the negative entropy [20] of the distribution parameterized by $\theta(\eta)$. This interpretation, along with the maximum entropy principle discussed in the next section, plays an important role in the development of the maximum-entropy formulation for model selection in Chapter 3.

2.3.3 Gaussian models as exponential families

The exponential family representation of this model is $p(x) \propto \exp\{-\frac{1}{2}x^T J x + h^T x\}$ based on the *information form* parameters $J = P^{-1}$ and $h = P^{-1}\mu$ (Section 2.2.2). Defining sufficient statistics ϕ as

$$\phi(x) = \left(\begin{pmatrix} x_s^2 \\ x_s \end{pmatrix}, \forall s \in V\right) \cup \left(x_s x_t, \forall\{s,t\} \in \binom{V}{2}\right) \tag{2.13}$$

we obtain θ and η parameters that are respectively given by elements of the (J, h) and (P, μ) representations:

$$\theta = \left(\begin{pmatrix} -\frac{1}{2}J_{s,s} \\ h_s \end{pmatrix}, \forall s\right) \cup (-J_{s,t}, \forall\{s,t\}) \tag{2.14}$$

$$\eta = \left(\begin{pmatrix} P_{s,s} \\ \mu_s \end{pmatrix}, \forall s\right) \cup (P_{s,t}, \forall\{s,t\}) \tag{2.15}$$

Converting between the moment and exponential parameters is equivalent to converting between P and J (and between μ and h; but the critical transformation is between P and J). This can be achieved by matrix inversion, which, in general, is an $O(|V|^3)$ computation. Note, also, that marginalization is simplest in the moment parameterization since the marginal density for a subset of variables E is determined by the corresponding subset of the moment parameters (a principle submatrix of P).

2.4 Information geometry of exponential families

In this section, we discuss a kind of projection onto sub-manifolds of exponential family distributions. In order to define projections, we begin by describing an appropriate notion of "distance".

2.4.1 Kullback-Leibler divergence

The *Kullback-Leibler* (KL) divergence between two distributions p_1 and p_2 [20] is given by:

$$D(p_1||p_2) = \int p_1(x) \log \frac{p_1(x)}{p_2(x)} dx.$$

It is a non-negative measure of the contrast between distributions p_1 and p_2, and is zero if and only if $p_1(x) = p_2(x)$ (a.e.).

For exponential family distributions the KL divergence possesses a special form. Let p_{η_1} and p_{θ_2} be distributions parameterized by moment and exponential parameters respectively. Then, the KL divergence $D(p_{\eta_1}||p_{\theta_2})$ is given by

$$D(p_{\eta_1}||p_{\theta_2}) = \Psi(\eta_1) + \Phi(\theta_2) - \langle \eta_1, \theta_2 \rangle,$$

where the functions Ψ and Φ are as defined in Section 2.3. We abuse notation slightly by referring to $D(p_{\eta_1}||p_{\theta_2})$ by $D(\eta_1||\theta_2)$, with the implicit understanding that the divergence is with respect to the distributions p_{η_1} and p_{θ_2}.

Keeping the second argument fixed, KL divergence is a convex function with respect to the moment parameters in the first argument:

$$D(\eta||\eta^*) = \Psi(\eta) + \Phi(\theta(\eta^*)) - \langle \eta, \theta(\eta^*) \rangle.$$

The convexity follows from the fact that $\Psi(\eta)$ is convex with respect to η and $\langle \eta, \theta(\eta^*) \rangle$ is simply a linear function of η. Note here that while the divergence is between two distributions parameterized by η and η^*, the convexity is with respect to η in the first argument, i.e., $D(\eta||\eta^*)$ is convex when viewed as a function of η in the first argument with the second argument fixed. A related point is that $D(\eta||\eta^*)$ is also the *Bregman distance* [11] induced by the function $\Psi(\eta)$:

$$D(\eta||\eta^*) = \Psi(\eta) - \Psi(\eta^*) - \langle \eta - \eta^*, \theta(\eta^*) \rangle.$$

A similar analysis can be carried out with respect to the exponential parameters, and by switching the order of the arguments in $D(\cdot||\cdot)$. With the first argument fixed, KL divergence is a convex function with respect to the exponential parameters in the second argument:

$$D(\theta^*||\theta) = \Psi(\eta(\theta^*)) + \Phi(\theta) - \langle \eta(\theta^*), \theta \rangle.$$

Again, the convexity follows from the fact that $\Phi(\theta)$ is convex with respect to θ and $\langle \eta(\theta^*), \theta \rangle$ is simply a linear function of θ. A Bregman distance interpretation can also be provided in this case, with $D(\theta^*||\theta)$ being the Bregman distance induced by the function $\Phi(\theta)$:

$$D(\theta^*||\theta) = \Phi(\theta) - \Phi(\theta^*) - \langle \eta(\theta^*), \theta - \theta^* \rangle.$$

2.4.2 I-projections in exponential families

An *e-flat manifold* is a sub-manifold of probability distributions defined by an affine subspace of exponential parameters. An important example of an e-flat manifold is the set of distributions that are Markov with respect to a graph $\mathcal{G} = (V, \mathcal{E})$:

$$\Theta(\mathcal{G}) = \{p_\theta(x) : \theta_E = 0 \text{ for } E \notin \mathcal{E}\}.$$

One can define m-flat manifolds in an analogous manner as the set of distributions that are defined by an affine subspace of moment parameters.

An *information projection*, or *I-projection*, of a distribution parameterized by θ^* onto an e-flat manifold $\mathcal{M}_e$ is defined as the closest point to p_{θ^*} in $\mathcal{M}_e$ in the sense of KL divergence:

$$\arg \min_{p_\theta \in \mathcal{M}_e} D(p_{\theta^*}||p_\theta).$$

Abusing notation slightly by denoting the elements of $\mathcal{M}_e$ as exponential parameters rather than the distributions represented by these parameters, the above problem becomes:

$$\arg \min_{\theta \in \mathcal{M}_e} D(\theta^*||\theta).$$

We have from the previous section that KL divergence is a convex function with respect to exponential parameters in the second argument when the first argument is fixed. Further, an e-flat manifold is a convex set of exponential parameters (because it is defined by an affine set of exponential parameters). Thus, the above formulation is a convex optimization problem with a unique global minimizer [10].

A particular Pythagorean identity holds for I-projections onto e-flat manifolds.

Theorem 2.9 *Let p_{θ^*} represent some distribution with I-projection $p_{\theta'}$ into an e-flat mani-*

fold $\mathcal{M}_e$. Then, for any $p_\theta \in \mathcal{M}_e$, we have that:

$$D(p_{\theta^*}||p_\theta) = D(p_{\theta^*}||p_{\theta'}) + D(p_{\theta'}||p_\theta).$$

An I-projection onto an m-flat manifold, and the associated Pythagorean identity, can be developed in an analogous manner (with a switching of the arguments in KL divergence).

2.4.3 Maximum entropy principle and duality to I-projection

We begin by stating the abstract maximum entropy principle with respect to general distributions before specializing with respect to exponential family distributions. Let ϕ be a set of features, and let $H(p) = -\mathbb{E}_p\{\log p(x)\}$ denote the entropy of distribution p [20]. Consider the following *maximum-entropy* problem:

$$\begin{aligned} \arg \quad \max_p \quad & H(p) \\ \text{s.t.} \quad & \mathbb{E}_p\{\phi(x)\} = \alpha^*. \end{aligned}$$

Theorem 2.10 Maximum-entropy principle*: The solution to the above maximum-entropy problem, if it exists, has the following form:*

$$p(x) \propto \exp\{\lambda^T \phi(x)\}.$$

Proof: See [20] for an elementary proof. Also see [42].

Thus, the maximum-entropy principle states that subject to linear constraints on a set of statistics, the entropy-maximizing distribution among *all* distributions lies in the exponential family based on those statistics used to define the constraints.

We now specialize the above formulation to exponential family distributions. Let η be the moment parameters of an exponential family, and let η_V and $\eta_{\mathcal{E}}$ represent the subset of moments corresponding to the vertices V and a set of edges $\mathcal{E}$ respectively on which constraints are specified:

$$\begin{aligned} \arg \quad \max_{\eta \in \mathcal{M}} \quad & H(\eta) \\ \text{s.t.} \quad & \eta_{\mathcal{E}}\{\phi(x)\} = \eta^*_{\mathcal{E}}, \;\; \eta_V = \eta^*_V. \end{aligned}$$

Here, $H(\eta)$ refers to the entropy $H(p_\eta)$. The maximum-entropy principle states that the entropy-maximizing distribution, if it exists, is Markov with respect to the graph $\mathcal{G} = (V, \mathcal{E})$. Note that the constraint space is restricted to exponential family distributions rather than all distributions. However, as can be easily seen from the maximum-entropy principle, this doesn't affect the solution to the problem. The above optimization is over a convex set of moment parameters (indeed, an affine set), and the objective function is concave with respect to η because $H(\eta) = -\Psi(\eta)$. Thus, the maximum-entropy problem parameterized with respect to moment parameters is a convex optimization program [10].

The maximum-entropy relaxation formulation in Chapter 3 introduces a relaxation of the equality moment constraints, leading to an effective framework for model selection.

Duality with I-projection Let $\Theta(\mathcal{G})$ represent an e-flat manifold consisting of distributions that are Markov with respect to graph $\mathcal{G} = (V, \mathcal{E})$. Let η^* be a distribution whose I-projection onto $\Theta(\mathcal{G})$ is the distribution parameterized by θ'. Next, consider the maximum-entropy problem with constraints $\eta_V = \eta_V^*$ and $\eta_\mathcal{E} = \eta_\mathcal{E}^*$. Let the entropy maximizing distribution be given by η'. Then, the pair (θ', η') are Legendre dual pairs, i.e., $\eta' = \Lambda(\theta')$.

This duality provides an interesting perspective about I-projections onto e-flat manifolds. Consider an alternative representation for η^* specified in terms of the *mixed coordinates* $(\eta_V^*, \eta_\mathcal{E}^*, \theta(\eta^*)_{\mathcal{E}^c})$, where $\mathcal{E}^c$ represents the complement of the set $\mathcal{E}$. With respect to these mixed coordinates, the I-projection onto $\Theta(\mathcal{G})$ can be viewed as a Euclidean projection with the resulting mixed coordinates being $(\eta_V^*, \eta_\mathcal{E}^*, 0)$.

2.5 Modeling and estimation in graphical models

In this section, we describe the modeling and estimation problems in graphical models. We discuss efficient algorithms to solve these problems in trees. For the general estimation problem in graphs with cycles, these tree-based algorithms provide the building block for the efficient algorithms discussed in Chapter 4.

2.5.1 Modeling

Model selection in graphical models is the problem of learning graph (Markov) structure given empirical statictics. In the context of exponential families, the problem is one of converting a specified set of empirical moments η^* to a $\theta \in \mathfrak{F}$, where $\mathfrak{F}$ represents a collection of tractable distributions. If $\mathfrak{F}$ is the collection of tree-structured distributions (i.e., treewidth-1 graphical models), the modeling problem can be solved efficiently using a maximum spanning tree formulation proposed by Chow and Liu [16]. If $\mathfrak{F}$ is chosen to be the collection of treewidth-k distributions for $k > 1$, the problem becomes NP-hard [49]. The maximum-entropy relaxation framework proposed in Chapter 3 is an alternative formulation, with roughly the same intent, for which a tractable solution exists.

In the rest of this section, we briefly describe the Chow-Liu algorithm. Let $\mathfrak{F} = \{p(x) : p(x) \text{ Markov on some tree}\}$. Given some empirical distribution p^*, the model selection problem reduces to:

$$\arg\min_{p \in \mathfrak{F}} D(p^* || p). \tag{2.16}$$

This problem involves both a variational aspect (the best approximation for a given tree), and a combinatorial aspect (the best tree). Letting $p^*(x_S)$ denote the marginal distribution of variable subset S, we compute the weights between every pair of variables $s, t \in V$ as follows:

$$w_{s,t} = D\left(p^*(x_s, x_t) || p^*(x_s) p^*(x_t)\right).$$

The weights correspond to the mutual information between variables x_s and x_t. Chow and Liu show that finding the maximum spanning tree with these edge weights solves the problem (2.16).

2.5.2 Estimation

In many applications one is provided with noisy observations y of a hidden state x, and the goal is to *estimate* x from y [69]. Taking the Bayesian point of view, this reduces to computing the mean, or in general the marginals, of the conditional distribution $p_{x|y}$. From the point of view of exponential families, estimation, also referred to as *inference*, reduces to computing all or a subset of the moments η corresponding to the exponential parameters θ of the conditional distribution. We assume that this conditional distribution is provided in advance, and our discussion is with respect to the graphical structure of the conditional distribution.

Estimation in trees The estimation problem can be solved with linear complexity in tree-structured graphical models using the Belief Propagation (BP) algorithm [62]. BP is a generalization of the recursive forwards-backwards algorithms originally developed for the special case of inference in Markov chains [48]. BP has an interpretation as a "message-passing" algorithm in that messages are passed locally between variables along the edges of the tree, thus providing an efficient method to compute exact estimates in a distributed manner. We refer the reader to a several publications that provide detailed derivations and new interpretations of BP [79], and only provide a brief summary of a particular parallel form of the algorithm here.

The BP messages are passed between each pair of variables connected by an edge, and can be computed as follows:

$$m_{t\to s}^{(n)}(x_s) \propto \sum_{x_t'} \psi_{st}(x_s, x_t')\psi_t(x_t') \prod_{u\in N(t)\backslash s} m_{u\to t}^{(n-1)}(x_t'),$$

where $N(t)$ refers to the neighbors of t. These messages can also be computed in a serial manner, so that a message is sent along an edge in each direction only once. For example, some arbitrary node $s \in V$ can be assigned as the "root" node, and messages can be passed in an up-down sweep from leaves (degree-1 nodes) to the root and back to the leaves. For tree-structured graphs, the BP messages converge in a finite number of iterations proportional to the diameter of the graph (length of the longest path) [62]. The (approximate) marginals at each iteration can be computed as follows:

$$p^{(n)}(x_s) \propto \psi_s(x_s) \prod_{t\in N(s)} m_{t\to s}^{(n)}(x_s).$$

When the messages converge in tree-structured models, these marginals are exact [62]. For a node s, each of its neighbors $t \in N(s)$ corresponds to the root of a subtree. The messages from each $t \in N(s)$ to s can be interpreted as the summary of the information in the subtree rooted at t.

In tree-structured Gaussian graphical models with $x|y \sim \mathcal{N}^{-1}(h, J)$, we obtain the following parametric BP updates [56, 63]:

$$\begin{aligned}
J_{t\backslash s}^{(n)} &= J_{t,t} + \sum_{u \in N(t)\backslash s} \Delta J_{u\to t}^{(n-1)} \quad &;& \quad h_{t\backslash s}^{(n)} = h_t + \sum_{u \in N(t)\backslash s} \Delta h_{u\to t}^{(n-1)} \\
\Delta J_{t\to s}^{(n)} &= -J_{s,t} {J_{t\backslash s}^{(n)}}^{-1} J_{t,s} \quad &;& \quad \Delta h_{t\to s}^{(n)} = -J_{s,t} {J_{t\backslash s}^{(n)}}^{-1} h_{t\backslash s}^{(n)} \\
J_{s,s}^{(n)} &= J_{s,s} + \sum_{t \in N(s)} \Delta J_{t\to s}^{(n)} \quad &;& \quad h_{s}^{(n)} = h_s + \sum_{t \in N(s)} \Delta h_{t\to s}^{(n)}.
\end{aligned}$$

At iteration n, the estimates for the mean and variance at node s are computed as $\mu_s^{(n)} = {J_{s,s}^{(n)}}^{-1} h_s^{(n)}$ and $P_{s,s}^{(n)} = {J_{s,s}^{(n)}}^{-1}$.

Estimation in graphs containing cycles For graphs with loops, a natural strategy is to cluster subsets of nodes in the graph to create a cycle-free graph with "super-nodes". This can be accomplished in a consistent manner by constructing a junction tree of the chordal supergraph of the given graphical model before propagating BP-like updates throughout the junction tree to provide exact marginals on the cliques [46]. This junction-tree algorithm is effective only for graphs with thin junction trees, because the algorithm involves an exact inference step within each clique.

Another approach that provides approximate marginals and has been applied with some success is the loopy belief propagation algorithm [81]. This algorithm essentially uses the same local updates as in the BP algorithm for trees presented previously.

A variety of other methods for estimation in graphs containing cycles have been developed based on performing a sequence of computations on tractable subgraphs (i.e., subgraphs on which exact estimation is efficient) [68, 74, 79]. We discuss some of these methods and present a rich class of algorithms that exploit tractable subgraph computations in Chapter 4.

2.6 Walk-sum interpretation of Gaussian estimation

Let the conditional distribution of a Gaussian graphical model be parameterized in the information form as $x|y \sim \mathcal{N}^{-1}(h, J)$. We note that J is a symmetric positive-definite matrix. The posterior mean can be computed as the solution to the following linear system:

$$J\widehat{x} = h. \tag{2.17}$$

For the discussion in this section, we assume that the matrix J defined on $\mathcal{G} = (V, \mathcal{E})$ has been normalized to have unit diagonal entries. For example, if D is a diagonal matrix containing the diagonal entries of J, then the matrix $D^{-\frac{1}{2}}JD^{-\frac{1}{2}}$ contains re-scaled entries of J at off-diagonal locations and 1's along the diagonal. Such a re-scaling does not affect the convergence results of the algorithms in Chapter 4 (see the chapter for more details). However, re-scaled matrices are useful in order to provide simple characterizations of walk-sums.

2.6.1 Walk-summable Gaussian graphical models

We begin by interpreting Gaussian estimation as the computation of walk-sums in $\mathcal{G}$. Let $R = I - J$. The off-diagonal elements of R are precisely the partial correlation coefficients from (2.5), and have the same sparsity structure as that of J (and consequently the same structure as $\mathcal{G}$). Let these off-diagonal entries be the edge weights in $\mathcal{G}$, i.e. $R_{t,s}$ is the weight of the edge $\{t, s\}$. A *walk* in $\mathcal{G}$ is defined to be a sequence of vertices $w = \{w_i\}_{i=0}^{\ell}$ such that $\{w_i, w_{i+1}\} \in \mathcal{E}$ for each $i = 0, \ldots, \ell - 1$. Thus, there is no restriction on a walk crossing the same node or traversing the same edge multiple times. The *weight* of the walk $\phi(w)$ is defined:

$$\phi(w) \triangleq \prod_{i=0}^{\ell-1} R_{w_i, w_{i+1}}.$$

Note that the partial-correlation matrix R is essentially a matrix of edge weights. Interpreted differently, one can also view each element of R as the weight of the length-1 walk between two vertices. In general, $\left(R^{\ell}\right)_{t,s}$ is then the walk-sum $\phi(s \xrightarrow{\ell} t)$ over the (finite) set of all length-ℓ walks from s to t [56], where the *walk-sum* over a finite set is the sum of the weights of the walks in the set. Based on this point of view, we can interpret estimation in Gaussian models from equation (4.1) in terms of walk-sums:

$$P_{t,s} = \left((I - R)^{-1}\right)_{t,s} = \sum_{\ell=0}^{\infty} \left(R^{\ell}\right)_{t,s} = \sum_{\ell=0}^{\infty} \phi(s \xrightarrow{\ell} t). \tag{2.18}$$

Thus, the covariance between variables x_t and x_s is the length-ordered sum over all walks from s to t. This, however, is a very specific instance of an inference algorithm that converges if the spectral radius condition $\varrho(R) < 1$ is satisfied (so that the matrix geometric series converges). Other inference algorithms, however, may compute walks in *different orders*. In order to analyze the convergence of general inference algorithms that submit to a walk-sum interpretation, a stronger condition was developed in [56] as follows. Given a countable set of walks $\mathcal{W}$, the *walk-sum* over $\mathcal{W}$ is the unordered sum of the individual weights of the walks contained in $\mathcal{W}$:

$$\phi(\mathcal{W}) \triangleq \sum_{w \in \mathcal{W}} \phi(w).$$

In order for this sum to be well-defined, we consider the following class of Gaussian graphical models.

Definition 2.3 *A Gaussian graphical model defined on $\mathcal{G} = (V, \mathcal{E})$ is said to be* walk-summable *if the absolute walk-sums over the set of all walks between every pair of vertices in $\mathcal{G}$ are well-defined. That is, for every pair $s, t \in V$,*

$$\bar{\phi}(s \to t) \triangleq \sum_{w \in \mathcal{W}(s \to t)} |\phi(w)| < \infty.$$

Here, $\bar{\phi}$ denotes absolute walk-sums over a set of walks. $\mathcal{W}(s \to t)$ corresponds to the set of all walks[3] beginning at vertex s and ending at the vertex t in $\mathcal{G}$. Section 2.6.2 lists some easily tested equivalent and sufficient conditions for walk-summability. Based on the absolute convergence condition, walk-summability implies that walk-sums over a countable set of walks can be computed in *any order* and that the unordered walk-sum $\phi(s \to t)$ is well-defined [36, 66]. Therefore, in walk-summable models, the covariances and means can be interpreted as follows:

$$P_{t,s} = \phi(s \to t), \tag{2.19}$$

$$\mu_t = \sum_{s \in V} P_{t,s} h_s = \sum_{s \in V} h_s \phi(s \to t), \tag{2.20}$$

where (2.18) is used in the first equation, and (2.19) in the second. In words, the covariance between variables x_s and x_t is the walk-sum over the set of all walks from s to t, and the mean of variable x_t is the walk-sum over all walks ending at t with each walk being reweighted by the potential value at the starting node.

The goal in walk-sum analysis is to interpret an inference algorithm as the computation of walk-sums in $\mathcal{G}$. If the analysis shows that the walks being computed by an inference algorithm are the same as those required for the computation of the means and covariances above, then the correctness of the algorithm can be concluded directly for walk-summable models. This conclusion can be reached regardless of the order in which the algorithm computes the walks due to the fact that walk-sums can be computed in *any* order in walk-summable models. Thus, the walk-sum formalism allows for very strong yet intuitive statements about the convergence of inference algorithms that submit to a walk-sum interpretation.

2.6.2 Conditions for walk-summability and Walk-sum algebra

Very importantly, there are easily testable necessary and sufficient conditions for walk-summability. Let $\bar{R}$ denote the matrix of the absolute values of the elements of R. Then, walk-summability is equivalent to either [56]

- $\varrho(\bar{R}) < 1$, or
- $I - \bar{R} \succ 0$.

From the second condition, one can draw a connection to H-matrices in the linear algebra literature [41, 76]. Specifically, walk-summable information matrices are symmetric, positive-definite H-matrices.

Walk-summability of a model is sufficient but not necessary for the validity of the model (positive-definite information/covariance). Many classes of models are walk-summable [56]:

1. Diagonally-dominant models, i.e. for each $s \in V$, $\sum_{t \neq s} |J_{s,t}| < J_{s,s}$.

[3] We denote walk-sets by $\mathcal{W}$ but generally drop this notation when referring to the walk-sum over $\mathcal{W}$, i.e. the walk-sum of the set $\mathcal{W}(\sim)$ is denoted by $\phi(\sim)$.

2. Valid non-frustrated models, i.e. every cycle has an even number of negative edge weights and $I - R \succ 0$. Special cases include valid attractive models and tree-structured models.

3. *Pairwise normalizable* models, i.e. there exists a diagonal matrix $D \succ 0$ and a collection of matrices $\{J_e \succeq 0 | (J_e)_{s,t} = 0 \text{ if } (s,t) \neq e,\ e \in \mathcal{E}\}$ such that $J = D + \sum_{e \in \mathcal{E}} J_e$.

An example of a commonly encountered walk-summable model in statistical image processing is the thin-membrane prior [29, 83]. Further, linear systems involving sparse diagonally dominant matrices are also a common feature in finite element approximations of elliptical partial differential equations [72].

We now describe some operations that can be performed on walk-sets, and the corresponding walk-sum formulas. These relations are valid in walk-summable models [56]:

- Let $\{\mathcal{U}_n\}_{n=1}^{\infty}$ be a countable collection of mutually disjoint walk-sets. From the sum-partition theorem for absolutely summable series [36], we have that $\phi(\cup_{n=1}^{\infty}\mathcal{U}_n) = \sum_{n=1}^{\infty}\phi(\mathcal{U}_n)$. Further, let $\{\mathcal{U}_n\}_{n=1}^{\infty}$ be a countable collection of walk-sets where $\mathcal{U}_n \subseteq \mathcal{U}_{n+1}$. We have that $\phi(\cup_{n=1}^{\infty}\mathcal{U}_n) = \lim_{n\to\infty}\phi(\mathcal{U}_n)$.

- Let $u = u_0 u_1 \cdots u_{\text{end}}$ and $v = v_{\text{start}} v_1 \cdots v_{\ell(v)}$ be walks such that $u_{\text{end}} = v_{\text{start}}$. The concatenation of the walks is defined to be $u \cdot v \triangleq u_0 u_1 \cdots u_{\text{end}} v_1 \cdots v_{\ell(v)}$. Now consider a walk-set $\mathcal{U}$ with all walks ending at vertex u_{end} and a walk-set $\mathcal{V}$ with all walks starting at $v_{\text{start}} = u_{\text{end}}$. The concatenation of the walk-sets $\mathcal{U}, \mathcal{V}$ is defined:

$$\mathcal{U} \otimes \mathcal{V} \triangleq \{u \cdot v \mid u \in \mathcal{U}, v \in \mathcal{V}\}.$$

  If every walk $w \in \mathcal{U} \otimes \mathcal{V}$ can be decomposed uniquely into $u \in \mathcal{U}$ and $v \in \mathcal{V}$ so that $w = u \cdot v$, then $\mathcal{U} \otimes \mathcal{V}$ is said to be *uniquely decomposable* into the sets $\mathcal{U}, \mathcal{V}$. For such uniquely decomposable walk-sets, $\phi(\mathcal{U} \otimes \mathcal{V}) = \phi(\mathcal{U})\phi(\mathcal{V})$.

Finally, the following notational convention is employed in this thesis. We use wild-card symbols ($*$ and $\bullet$) to denote a union over all vertices in $\mathcal{G}$. For example, given a collection of walk-sets $\mathcal{W}(s)$, we interpret $\mathcal{W}(*)$ as $\bigcup_{s \in V} \mathcal{W}(s)$. Further, the walk-sum over the set $\mathcal{W}(*)$ is defined $\phi(\mathcal{W}(*)) \triangleq \sum_{s \in V} \phi(\mathcal{W}(s))$. In addition to edges being assigned weights, vertices can also be assigned weights (for example, the potential vector h). A re-weighted walk-sum of a walk $w = w_0 \cdots w_\ell$ with vertex weight vector h is then defined to be $\phi(h; w) \triangleq h_{w_0}\phi(w)$. Based on this notation, the mean of variable x_t from (2.20) can be re-written as

$$\mu_t = \phi(h; * \to t). \tag{2.21}$$

2.7 Algorithms for convex optimization

We conclude the background chapter by describing an abstract framework for convex optimization problems, and specifying a basic primal-dual interior point algorithm to solve these problems. This discussion is relevant to understanding and solving the maximum-entropy formulation for model selection presented in Chapter 3. As with the previous sections, our presentation is brief. For more details, we refer the reader to the vast and rich literature on convex optimization [8, 10, 65]. This section follows the style of presentation in [10].

2.7.1 General convex optimization problems, duality, and optimality conditions

Let f_i be convex functions for $i = 0, \ldots, m$, and let h_j be affine functions for $j = 1, \ldots, p$. Consider the following *convex optimization* problem:

$$\begin{array}{lll} (\mathrm{P}) \quad \arg\min\limits_{x} & f_0(x) \\ \quad\quad\quad\quad \text{s.t.} & f_i(x) \leq 0, \ i = 1, \ldots, m \\ & h_j(x) = 0, \ j = 1, \ldots, p. \end{array}$$

Our maximum-entropy formulation in Chapter 3 is expressed in terms of *maximizing* a *concave* function, subject to convex constraints. We will also refer to such problems as convex optimization problems. Convex optimization problems possess the interesting property that any locally optimal solution is also globally optimal. Thus, if a solution to the problem exists, it is unique. Given such an optimization problem, the *Lagrangian* is defined as follows:

$$L(x, \lambda, \nu) = f_0(x) + \sum_i \lambda_i f_i(x) + \sum_j \nu_j h_j(x).$$

The vectors λ and ν are known as the *dual variables*. The *Lagrange dual* function is then the solution to the following optimization problem:

$$g(\lambda, \nu) = \inf_x L(x, \lambda, \nu),$$

where the domain of minimization is the intersection of the domains of the functions f_i and h_j. The *dual problem* is defined as a maximization involving $g(\lambda, \nu)$:

$$\begin{array}{lll} (\mathrm{D}) \quad \arg\max\limits_{\lambda,\nu} & g(\lambda, \nu) \\ \quad\quad\quad\quad \text{s.t.} & \lambda_i \geq 0, \ i = 1, \ldots, m. \end{array}$$

We will refer to the constraints in this problem using the shorthand notation $\lambda \succeq 0$. The dual problem is also convex (in fact, even if the original problem (P) is not convex).

The problem (P) is also known as the primal problem. Let p^* be the optimal value of (P). One can check that $g(\lambda, \nu) \leq p^*$ for any $\lambda \succeq 0$ and any ν. Therefore, the dual problem (D) can be viewed as optimizing over all the lower bounds in order to find the tightest one. Let d^* be the optimal value of (D). We then have the following inequality, known as *weak duality*:

$$p^* \geq d^*.$$

The non-negative difference $p^* - d^*$ is the *duality gap*. *Strong duality* is said to hold when $p^* = d^*$, i.e. when there is no duality gap. Slater's constraint qualifications provide a simple condition for checking that there is no duality gap in a convex optimization problem:

$$\begin{aligned} \exists x' \text{ s.t.} \qquad & f_i(x') < 0, \text{ for } i = 1, \dots, m, \text{ and} \\ & h_j(x') = 0, \text{ for } j = 1, \dots, p. \end{aligned}$$

These conditions state that there exists an x' such that the equality constraints are satisfied, and the inequality constraints are satisfied *strictly*.

Optimality conditions Suppose that Slater's condition holds for a convex optimization problem. The values x^* and (λ^*, ν^*) are the optimal values for the primal and dual variables respectively, if and only if they satisfy the following Karush-Kuhn-Tucker (KKT) conditions [50, 52]:

$$\begin{aligned} f_i(x^*) &\leq 0, \; i = 1, \dots, m \\ h_j(x^*) &= 0, \; j = 1, \dots, p \\ \lambda_i^* &\geq 0, \; i = 1, \dots, m \\ \lambda_i^* f_i(x^*) &= 0, \; i = 1, \dots, m \qquad (2.22) \\ \nabla f_0(x^*) + \sum_i \lambda_i^* \nabla f_i(x^*) + \sum_j \nu_j^* \nabla h_j(x^*) &= 0. \qquad (2.23) \end{aligned}$$

The fourth condition (2.22) is known as *complementary slackness*. The set $\{i : \lambda_i^* = 0\}$ consists of the indices corresponding to *inactive* constraints.

2.7.2 Primal-dual interior point methods

Interior-point methods form an important class of algorithms to solve optimization problems. They are distinguished by the fact that they arrive at the solution by traversing through the interior of the feasible set (the set of points that satisfy the constraints), rather than following the boundary of the feasible set (see for example the simplex method [23]). Note that Slater's constraint qualifications must hold for interior-point methods to be applicable. Thus, we assume for the rest of this section that Slater's condition is satisfied.

Primal-dual algorithms form an important class of interior-point methods, and can often achieve super-linear convergence performance. Here, we describe a basic version of the algorithm, the details of which can be found in [10]. For a more advanced treatment of these algorithms, we refer the reader to [85]. We will focus here on convex optimization problems consisting only of convex, inequality constraints, because the maximum-entropy framework in Chapter 3 does not contain any equality constraints.

A primal-dual algorithm is so called because it jointly computes both the primal optimal variables x^* and the dual optimal variables λ^*. We describe some notation before proceeding with the specification of the algorithm. Let $f(x) = [f_1(x), \dots, f_m(x)]^T$ be a vector of the constraint functions evaluated at a point x, and let the Jacobian of $f(x)$ be defined:

$$Df(x) = \left[\nabla f_1(x)^T, \ldots, \nabla f_m(x)^T\right]^T.$$

The *dual residual* is defined as follows:

$$r_{dual} = \nabla f_0(x) + Df(x)^T \lambda,$$

and the *central residual* as:

$$(r_{cent})_i = -\lambda_i f_i(x) - \frac{1}{t},\ i = 1, \ldots, m,$$

where $t > 0$ is a parameter that controls how close the algorithm is to convergence through the ratio $\frac{m}{t}$ (this ratio is also called the *surrogate duality gap*). These residual parameters are used to couple the primal and dual variables, so that they can be jointly optimized.

The input to the primal-dual algorithm is a strictly feasible x, i.e. $f(x) \prec 0$, dual variables $\lambda \succ 0$, and some $\mu > 1$, $\epsilon_{feas} > 0$ and $\epsilon > 0$ (see details to follow for choosing these parameters). The following are the main steps in the primal-dual algorithm:

1. Set $t = \frac{\mu m}{-f(x)^T \lambda}$.

2. Compute the residuals r_{dual} and r_{cent}. Compute a *search direction* $(\Delta x_{pd}, \Delta\lambda_{pd})$. The primal search direction is the solution to the linear system:

$$M_{pd} \cdot \Delta x_{pd} = -\left(\nabla f_0(x) + \frac{1}{t}\sum_i \frac{\nabla f_i(x)}{-f_i(x)}\right),$$

where the matrix M_{pd} is specified as

$$M_{pd} = \nabla^2 f_0(x) + \sum_i \lambda_i \nabla^2 f_i(x) + \sum_i \frac{\lambda_i}{-f_i(x)} \nabla f_i(x) \nabla f_i(x)^T.$$

The dual search direction is then computed as:

$$\Delta\lambda_{pd} = -\mathbf{diag}(f(x))^{-1}\mathbf{diag}(\lambda) Df(x) \Delta x_{pd} + \mathbf{diag}(f(x)) r_{cent},$$

where **diag**(v) denotes a diagonal matrix with the entries of the vector v along the diagonal.

3. Given the search directions, the next task is to determine the *length* of the step to be taken in these directions. This one-dimensional search procedure is known as *line-search*. Let $r_{joint}(x, \lambda) = [r_{cent}(x, \lambda)\ r_{dual}(x, \lambda)]^T$ denote the joint residual vector. Let $x_{new} = x + s\Delta x_{pd}$ and $\lambda_{new} = \lambda + s\Delta\lambda_{pd}$ denote the updated values of x and λ if a step of length s were to be taken along the search directions. Given a parameter α (see details below), the largest $s < 1$ must be found such that

$$\|r_{joint}(x_{new}, \lambda_{new})\| \leq (1 - \alpha s) \|r_{joint}(x, \lambda)\|,$$

After such an s is found, we update x and λ as:

$$\begin{aligned} x &\leftarrow x + s\Delta x_{pd} \\ \lambda &\leftarrow \lambda + s\Delta\lambda_{pd}. \end{aligned}$$

4. If $\|r_{dual}\| \leq \epsilon_{feas}$ and $-f(x)^T\lambda \leq \epsilon$, then STOP. Otherwise, go back to step 1.

The parameter μ is chosen to be around 10, and the line-search parameter α is chosen to be in the range 0.01 to 0.1. The tolerances ϵ and ϵ_{feas} can be used to specify the accuracy to which a solution is desired, and can be chosen to be around 10^{-8}.

Step 2 in the primal-dual algorithm is the most computationally intensive among the four steps. The dimension of the matrix M_{pd} can be large in applications involving many variables. In Chapter 3, we exploit the sparsity structure of this matrix in order to solve the linear system efficiently.

Chapter 3

Modeling using Maximum Entropy Relaxation

In this chapter, we propose a novel framework for learning the Markov structure in a collection of variables given an empirical distribution. Our approach can also be viewed as a technique to construct tractable graphical model approximations to intractable distributions. The method is based on the maximum-entropy principle, which implicity favors sparse graphical models (see Section 2.4.3). Our formulation is based on a relaxation of the equality moment constraints in the classical maximum-entropy problem. We replace these linear equality constraints by a collection of non-linear convex constraints, each based on the marginal information divergence between a subset of variables.

Several methods have recently appeared [6, 54, 77] using ℓ_1-penalized information projections, where an ℓ_1-norm on model parameters is used to favor sparse graphs. It is known that these methods are dual to the maximum-entropy method using ℓ_∞ moment constraints [28], which is similar to our approach. However, the constraints in our formulation are expressed, perhaps more naturally, in terms of relative entropy. As a result, an interesting feature of our framework is that the optimal distribution is *invariant* to reparameterization of the exponential family. We emphasize this point later in this chapter when we formally discuss our method.

We begin by providing a detailed description of our framework, and discuss some of its salient features in Section 3.1. In Section 3.2, we develop a primal-dual interior point algorithm to solve this problem by exploiting tractable calculations on chordal graphs. We demonstrate the effectiveness of our approach in learning the Markov structure of simple models from data through simulation results in Section 3.3. We conclude with a brief summary in Section 4.5. Much of our discussion in Section 3.1 and Section 3.2 is for general exponential family distributions. Where relevant, we provide specific details pertaining to the case of Gaussian models. The simulation results in Section 3.3 focus exclusively on Gaussian model selection.

3.1 Maximum Entropy Relaxation

3.1.1 Exponential Family Formulation

Let $\mathfrak{F}$ be a marginalizable exponential family (see Section 2.3) with statistics ϕ and moment parameters $\eta \in \mathcal{M}$. Let p^* be a given probability distribution with corresponding moments $\eta^* \triangleq \mathbb{E}_{p^*}\{\phi(x)\}$. We would like to identify a lower-order Markov approximation of p^* defined on some sparse graph (to be determined) that still provides a reasonably faithful approximation of the given p^*.

We propose to address this problem by solution of the following *Maximum Entropy Relaxation* (MER) problem:

$$\text{(MER)} \quad \begin{array}{ll} \arg\max & H(\eta) \\ \text{s.t.} & \eta \in \mathcal{M} \\ & D_E(\eta_{\bar{E}} || \eta^*_{\bar{E}}) \leq \delta_E, \ \forall E \in \mathcal{E} \\ & D_v(\eta_v || \eta^*_v) \leq \delta_v, \ \forall v \in V \end{array}$$

where H is the entropy in the complete family, D_E and D_v are the marginal divergences on $E \in \mathcal{E}$ and $v \in V$ respectively, the edge set $\mathcal{E}$ serves to specify the constraint set, and $\delta = \{\delta_E, E \in \mathcal{E}\} \cup \{\delta_v, v \in V\}$ are a specified set of tolerances on marginal divergences. In this problem, $\eta_{\bar{E}}$ is the subset of all the moment parameters that have support inside edge E, i.e., $\eta_{\bar{\{i,j\}}} = \{\eta_i, \eta_j, \eta_{ij}\}$. Note that we restrict our attention to pairwise edges in the edge-set $\mathcal{E}$, which suffices for Gaussian models; however, this restriction can be removed to include higher-order interactions between subsets of variables (for example in discrete models).

The entropy $H(\eta)$ is a strictly concave function of η, because the statistics ϕ are minimal (see Section 2.3). $\mathcal{M}$ is a convex subset of $\mathcal{R}^d$, and each marginal divergence $D_E(\eta_{\bar{E}} || \eta^*_{\bar{E}})$ (or $D_v(\eta_v || \eta^*_v)$) is a convex function of $\eta_{\bar{E}}$ (or η_v) for any fixed valued of $\eta^*_{\bar{E}}$ (or η^*_v). Hence, this is a convex optimization problem. Thus, if the maximum entropy is obtained by some $\tilde{\eta} \in \mathcal{M}$, it is the unique solution of the MER problem. Further, for $\delta > 0$ the problem is strictly feasible in that the inequality constraints can be satisfied strictly. This implies that Slater's condition is satisfied (see Section 2.7). Therefore, we are free to apply the various analysis tools such as complementary slackness and the Karush-Kuhn-Tucker (KKT) conditions described in Section 2.7.1, and the primal-dual interior-point algorithm of Section 2.7.2.

Invariance with respect to reparameterization In the formulation of the MER problem, the subset of moment parameters $\eta_{\bar{E}}$ corresponding to edge E are exactly the moment parameters of the marginal distribution on edge E. This is because the underlying exponential family is marginalizable (see Section 2.3.2). Thus, the MER problem could alternatively be written in a more general form as follows:

$$\begin{array}{ll} \arg\max & H(p) \\ \text{s.t.} & p \in \mathfrak{F} \\ & D_E(p_E || p^*_E) \leq \delta_E, \ \forall E \in \mathcal{E} \\ & D_v(p_v || p^*_v) \leq \delta_v, \ \forall v \in V \end{array}$$

where the only constraint on p is that it must belong to the exponential family $\mathfrak{F}$. Note here that we assume that the target distribution p^* is also an element of the full exponential family. Hence, our MER formulation possesses an intrinsic quality of being invariant to reparameterization of the exponential family. The rest of this chapter, however, is focused on analyzing and developing tractable solutions to the MER problem in terms of the moment parameters η.

3.1.2 Markovianity of MER solution

We have not imposed any Markov constraints on the solution of the MER problem. The set of edges $\mathcal{E}$ serves to summarize the constraint set, and may very well correspond to the fully connected graph, i.e., $\mathcal{E} = \binom{V}{2}$. However, we have the following result concerning the Markovianity of the MER distribution $\tilde{\eta}$. The constraint on edge $E \in \mathcal{E}$ is said to be *active* if $D_E(\tilde{\eta}_{\bar{E}}||\eta^*_{\bar{E}}) = \delta_E$. Let $\mathcal{E}_{active}$ denote the collection of active edges, and let $\mathcal{G}_{active} = (V, \mathcal{E}_{active})$ denote the graph formed by the active edges.

Theorem 3.1 Model-Thinning Effect*: The solution of the MER problem (if it exists) is Markov with respect to the graph $\mathcal{G} = (V, \mathcal{E})$ that specify the constraints. Moreover, and very importantly, it is also Markov on the sub-graph $\mathcal{G}_{active} = (V, \mathcal{E}_{active})$ defined by just the active edge constraints.*

Proof: The KKT conditions assert that there exist Lagrange multipliers $\lambda \geq 0$ such that

$$\nabla H(\tilde{\eta}) - \sum_{E \in \mathcal{E}} \lambda_E \cdot \nabla_{\tilde{\eta}} D_E(\tilde{\eta}_{\bar{E}}||\eta^*_{\bar{E}}) - \sum_{v \in V} \lambda_v \cdot \nabla_{\tilde{\eta}} D_v(\tilde{\eta}_v||\eta^*_v) = 0 \tag{3.1}$$

Moreover, by complementary slackness, $\lambda_E = 0$ for the inactive edge constraints. Hence, using $\nabla H(\tilde{\eta}) = -\Lambda^{-1}(\tilde{\eta})$, $\nabla_{\tilde{\eta}} D_E(\tilde{\eta}_{\bar{E}}||\eta^*_{\bar{E}}) = \Lambda_E^{-1}(\tilde{\eta}_{\bar{E}}) - \Lambda_E^{-1}(\eta^*_{\bar{E}})$, and $\nabla_{\tilde{\eta}} D_v(\tilde{\eta}_v||\eta^*_v) = \Lambda_v^{-1}(\tilde{\eta}_v) - \Lambda_v^{-1}(\eta^*_v)$, we have

$$\Lambda^{-1}(\tilde{\eta}) + \sum_{E \in \mathcal{G}_{active}} \lambda_E \cdot (\Lambda_E^{-1}(\tilde{\eta}_{\bar{E}}) - \Lambda_E^{-1}(\eta^*_{\bar{E}})) + \sum_{v \in V} \lambda_v \cdot (\Lambda_v^{-1}(\tilde{\eta}_v) - \Lambda_v^{-1}(\eta^*_v)) = 0. \tag{3.2}$$

Note that strictly speaking, the transformations Λ_E^{-1} and Λ_v^{-1} operate on lower-dimensional vectors of moments than the full moment vector $\tilde{\eta}$; in order for the left-hand-side to be meaningful, the terms in the second and third sums are appropriately zero-padded, and we abuse notation by only referring to the lower-dimensional transformations Λ_E^{-1} and Λ_v^{-1} without explicitly mentioning the zero-padding.

Examining (3.2), $\tilde{\theta}_E \triangleq (\Lambda^{-1}(\tilde{\eta}))_E = 0$ if E is not an edge of $\mathcal{G}_{active}$, which implies that the corresponding MER probability distribution $\tilde{\eta}$ is Markov on $\mathcal{G}_{active}$. Since $\mathcal{G}_{active}$ is a subgraph of $\mathcal{G}$, we also have that $\tilde{\eta}$ is Markov on $\mathcal{G}$. $\square$

Fundamentally, this is the mechanism that allows us to learn graph structure by solving a convex optimization problem. When edge constraints are inactive in the final solution of MER, the model is automatically "thinned".

3.2 Algorithms for Solving MER

We discuss the key steps in developing a tractable algorithm to solve the MER problem. Section 3.2.1 describes the efficient computation of entropy, its gradient, and its Hessian in thin chordal graphs. These computations form the basis for finding the primal-dual search direction. As with the previous section, most of this section is relevant for general exponential families. However, we mention specific formulas and the associated computational complexity for Gaussian models.

3.2.1 Computations on Thin Chordal Graphs

From Section 2.2.1, we have that distributions that are Markov on a chordal graph $\mathcal{G}$ can be factored as

$$p(x) = \frac{\prod_{C \in \mathcal{C}} p(x_C)}{\prod_{S \in \mathcal{S}} p(x_S)}, \tag{3.3}$$

where $\mathcal{C}$ is the set of maximal cliques and $\mathcal{S}$ is the collection of edge-wise separators $C_i \cap C_j$ defined by the edges $\{C_i, C_j\}$ of any junction tree of the graph. We remind the reader that a chordal graph is said to be *thin* if it has small maximal cliques.

Using (3.3), entropy can be expressed in terms of marginal entropies on the cliques and separators of a junction tree of the graph. The marginalizability of the exponential family $\mathfrak{F}$ plays a key role here. In particular, let $\eta_{\mathcal{G}}$ correspond to the subset of moment parameters of the chordal graph $\mathcal{G} = (V, \mathcal{E})$, i.e., $\eta_{\mathcal{G}} = \{\eta_v : v \in V\} \cup \{\eta_E : E \in \mathcal{E}\}$. Let $C \subset V$ be some clique of the graph $\mathcal{G}$, so that we can represent the clique as the graph $(C, \binom{C}{2})$. Since $\mathfrak{F}$ is marginalizable, the moment parameters corresponding to the marginal distribution of variables C are precisely the *subset* of the moment parameters[1] of $\eta_{\mathcal{G}}$ corresponding to nodes and edges in $(C, \binom{C}{2})$, i.e., $\eta_{\bar{C}} = \eta_C \cup \eta_{\binom{C}{2}}$.

Based on the factorization (3.3) and using the notation introduced, we have that:

$$H_{\mathcal{G}}(\eta_{\mathcal{G}}) = \sum_{C \in \mathcal{C}} H_C(\eta_{\bar{C}}) - \sum_{S \in \mathcal{S}} H_S(\eta_{\bar{S}}). \tag{3.4}$$

We clarify here that when the specified set of moment parameters $\eta_{\bar{C}}$ correspond to the fully connected graph of the variables at nodes in the *set* C, the resulting entropy is simply denoted H_C; however, if the moment parameters $\eta_{\mathcal{G}}$ correspond to any *graph* $\mathcal{G}$, then the associated entropy is denoted by $H_{\mathcal{G}}$. We use similar notation for other functions throughout the rest of this chapter. The gradient of negative-entropy with respect to the moment parameters are the corresponding exponential parameters (see Section 2.3.2). Differentiating both sides of (3.4) with respect to moment parameters, we have that

$$\Lambda_{\mathcal{G}}^{-1}(\eta_{\mathcal{G}}) = \sum_{C \in \mathcal{C}} \Lambda_C^{-1}(\eta_{\bar{C}}) - \sum_{S \in \mathcal{S}} \Lambda_S^{-1}(\eta_{\bar{S}}), \tag{3.5}$$

where $\Lambda^{-1}(\eta) = \theta(\eta)$ is the mapping from moment parameters to the associated expo-

[1]These relations hold even for non-chordal graphs, but our focus here is on computations with respect to cliques in chordal graphs because we want to exploit the structure of the factorization (3.3).

nential parameters. In this equation, the mappings in each sum on the right-hand-side are between variables that have lower dimension than $\eta_{\mathcal{G}}$. Hence, each term in each sum on the right-hand-side must be appropriately zero-padded (zeroes at all locations not corresponding to the subgraph $(C, \binom{C}{2})$ or $(S, \binom{S}{2})$) so that the equation is consistent. Next, we have from the discussion in Section 2.3.2 that the Hessian of negative-entropy is the Fisher information with respect to the moment parameterization. Thus, differentiating both sides of (3.5) to compute the Jacobian $D\Lambda^{-1}(\eta)$ (or the Hessian $-\nabla^2 H(\eta)$), we have

$$G^*_{\mathcal{G}}(\eta_{\mathcal{G}}) = \sum_{C \in \mathcal{C}} G^*_C(\eta_{\bar{C}}) - \sum_{S \in \mathcal{S}} G^*_S(\eta_{\bar{S}}). \tag{3.6}$$

Implicit again in (3.6) is the padding of the terms on the right with zeroes at appropriate locations. A key point here is that the Fisher information is *sparse* for thin chordal graphs, because each term in each sum of the right-hand-side of (3.6) has small support (as the maximal cliques are small). The sparsity of the Fisher information matrix is important later when we use sparse matrix computations to efficiently compute search directions in each step of the primal-dual interior-point method.

In order to perform these calculations, we need to be able to compute H_C, Λ_C^{-1} and G^*_C for fully-connected subsets of nodes $C \subset V$. We provide explicit formulas for these computations in Gaussian models. These calculations are tractable for small subsets, thus enabling efficient computation of (3.4), (3.5), and (3.6) for thin chordal graphs.

Computations in Gaussian models In Gaussian models[2], letting $P(\eta)$ denote the covariance matrix of a set of moment parameters η (see Section 2.3), one obtains [3, 20]:

$$H(\eta) = \tfrac{1}{2}(\log\det P(\eta) + |V| \cdot \log 2\pi e).$$

In the complete Gaussian model, the entries of $G^*(\eta)$ are given by

$$\begin{aligned} G^*_{st,uv}(\eta) &= J_{s,u}J_{t,v} + J_{s,v}J_{t,u}, \\ G^*_{st,u}(\eta) &= J_{s,u}J_{t,u}, \\ G^*_{s,t}(\eta) &= \tfrac{1}{2}J^2_{s,t}, \end{aligned}$$

with $J = P(\eta)^{-1}$ (see Appendix B for derivations). We use these formulas to compute each term in the sums on the right-hand-sides of equations (3.4-3.6). The computation of (3.4-3.5) is $\mathcal{O}(|V|w^3)$ in a Gaussian model, where w is the maximum clique size. Computing the sparse matrix (3.6) is $\mathcal{O}(|V|w^4)$.

Difficulty in computations for non-chordal graphs The entropy function can only be computed explicitly for a set of moment parameters η corresponding to the fully connected graph, i.e. one including parameters corresponding to all vertices and all pairs of vertices. Computation of entropy given a subset of the moment parameters $\eta_{\mathcal{G}}$ corresponding to a

[2]Since the mean vector does not play a critical role in the model identification problem, we assume throughout this chapter that the models are zero-mean.

chordal graph $\mathcal{G}$ is then possible by decomposing $H_{\mathcal{G}}(\eta_{\mathcal{G}})$ in terms of local entropy computations on cliques and separators (3.4), each of which is an explicit entropy computation on a set of moment parameters corresponding to a fully connected subgraph. However, computing $H_{\mathcal{G}'}(\eta_{\mathcal{G}'})$ for a non-chordal graph $\mathcal{G}'$ involves solving a variational problem to compute the *maximum-entropy completion* to a chordal supergraph. More precisely, let $\mathcal{G}$ be a chordal supergraph of the non-chordal graph $\mathcal{G}'$, and let $\eta_{\mathcal{G}\setminus\mathcal{G}'}$ be the set of moment parameters associated with the edges that are present in $\mathcal{G}$ but not in $\mathcal{G}'$. Then, $H_{\mathcal{G}'}(\eta_{\mathcal{G}'})$ is the optimal value of the following maximum-entropy problem:

$$\begin{aligned} \arg\max_{\hat{\eta}_{\mathcal{G}}} \quad & H_{\mathcal{G}}(\hat{\eta}_{\mathcal{G}}) \\ \text{s.t.} \quad & \hat{\eta}_{\mathcal{G}'} = \eta_{\mathcal{G}'}, \end{aligned}$$

where $\hat{\eta}_{\mathcal{G}} = [\hat{\eta}_{\mathcal{G}'}\ \hat{\eta}_{\mathcal{G}\setminus\mathcal{G}'}]$. Evaluating the objective function in this optimization problem is based on the formula (3.4) for computing entropy in a chordal graph. From the duality of the maximum-entropy principle to I-projections onto e-flat manifolds (see Section 2.4.3), one can check that the exponential parameters underlying the solution to this maximum-entropy completion problem will be zero for the edges in $\mathcal{G}$ but not in $\mathcal{G}'$ (i.e. the extra fill edges used to obtain a chordal supergraph).

3.2.2 Incremental Approach

We describe an algorithm to solve the MER problem. As with the last section, much of our discussion is relevant for MER problems involving any marginalizable exponential family. The MER problem is formulated with respect to the complete exponential family (not assuming any Markov structure in advance). For problems of even moderate size, direct solution of MER in the complete model can become intractable due to the high dimension of the parameter vector η. However, based on the model-thinning property, we conclude that if the solution is actually sparse, it should not be necessary to solve MER in the complete parameterization. Hence, we propose the following algorithm to adaptively identify the subset of active constraints and a corresponding lower-order Markov family containing the MER solution:

1. Set $k = 0$. Start with the disconnected graph $\mathcal{G}^{(0)} = (V, \emptyset)$ including only node constraints.

2. Find a chordal supergraph $\mathcal{G}_c^{(k)}$ of $\mathcal{G}^{(k)}$ [7, 13, 70]. Solve the reduced MER subproblem with respect to the moments corresponding to the chordal graph $\mathcal{G}_c^{(k)}$, i.e. $\eta_{\mathcal{G}_c^{(k)}}$. Only include the node and edge constraints based on $\mathcal{G}^{(k)} = (V, \mathcal{E}^{(k)})$. The reduced problem can be formulated as follows:

$$\begin{aligned} \arg\max \quad & H_{\mathcal{G}_C^{(k)}}(\eta_{\mathcal{G}_c^{(k)}}) \\ \text{s.t.} \quad & \eta_{\mathcal{G}_c^{(k)}} \in \mathcal{M}_{\mathcal{G}_c^{(k)}} \\ & D_E(\eta_{\bar{E}} \| \eta^*_{\bar{E}}) \le \delta_E,\ \forall E \in \mathcal{E}^{(k)} \\ & D_v(\eta_v \| \eta^*_v) \le \delta_v,\ \forall v \in V, \end{aligned}$$

where $\mathcal{M}_{\mathcal{G}_c^{(k)}}$ corresponds to the set of realizable moment parameters based on the e-flat manifold of distributions that are Markov on $\mathcal{G}_c^{(k)}$. This reduced problem can be solved using the primal-dual method described in the following section.

3. The solution in the previous step only provides the moment parameters corresponding to the chordal graph $\mathcal{G}_c^{(k)}$. One could then compute the maximum-entropy completion to obtain the full set of moment parameters. This can be achieved efficiently in the Gaussian case using $\mathcal{O}(|V|^2 w^3)$ computations [31], where w is the treewidth of $\mathcal{G}_c^{(k)}$. Denote the resulting solution of the full set of moment parameters by $\tilde{\eta}_{\bar{V}}^{(k)}$. Evaluate the constraint violations $g_E = D_E((\tilde{\eta}_{\bar{V}}^{(k)})_{\bar{E}} || \eta_{\bar{E}}^*) - \delta_E$ for all $E \in \mathcal{E} \setminus \mathcal{E}^{(k)}$.

4. If $g_E < 0$ for all $E \in \mathcal{E} \setminus \mathcal{E}^{(k)}$, STOP. Then, $\tilde{\eta} = \tilde{\eta}_{\bar{V}}^{(k)}$ is the MER solution.

5. Otherwise, build $\mathcal{G}^{(k+1)}$ by adding edges to $\mathcal{G}^{(k)}$ corresponding to the K largest, positive constraint violations (if there are less than K such edges, add just the edges corresponding to violated constraints). Set $k \leftarrow k + 1$ and go back to step 2.

We emphasize two important features of this incremental approach:

- Provided we continue adding violated constraints until all the remaining constraints are satisfied, the final graph $\mathcal{G}^{(k)}$ contains $\tilde{\mathcal{G}}$ (the graph with respect to which the global MER solution is Markov), and $\tilde{\eta}_{\bar{V}}^{(k)}$ is therefore the *optimal* solution of the original MER problem in the complete family. Unlike greedy methods used in combinatorial approaches, our method is distinguished by the fact that the solution obtained is optimal with respect to our global MER criterion.

- Although we embed the problem in a chordal graph, we still only impose constraints over $\mathcal{G}^{(k)}$, and hence, by the model thinning property, this embedding does not alter the MER solution with respect to $\mathcal{G}^{(k)}$. In other words, adding fill edges to obtain a chordal super-graph $\mathcal{G}_c^{(k)}$ does not spoil the Markov structure of the MER solution with respect to $\mathcal{G}^{(k)}$. Therefore, the MER solution could, in general, be Markov with respect to a non-chordal graph.

Suppose that the solution to the global MER problem has low treewidth. It is likely then that each of the graphs $\mathcal{G}^{(k)}$ also have low treewidth (i.e. their chordal supergraphs $\mathcal{G}_c^{(k)}$ have low treewidth), because only those features that are most strongly violated are added at each step. Experimental results in Section 3.3 confirm that such behavior is typically the case. It is this mechanism that makes our approach tractable and scalable, because the computations in steps 2 and 3 can be performed efficiently when the treewidth of $\mathcal{G}^{(k)}$ is small. In Chapter 5, we propose an extension of our approach to handle cases where the MER solution has an intractable graph (i.e. high treewidth).

3.2.3 Primal-Dual Method on Thin Chordal Graphs

We turn our focus to efficiently solving the reduced MER sub-problem in step 2 of the incremental approach described in the previous section, provided the constraint graph $\mathcal{G}^{(k)}$

is sufficiently sparse. By embedding the optimization problem in a Markov family based on a chordal graph, we are able to compute entropy and its derivatives in an efficient manner using the formulas in Section 3.2.1. As discussed in that section, computing $h(\eta_{\mathcal{G}})$ for a *non-chordal* graph $\mathcal{G}$ is difficult. Hence, the chordal graph embedding method is a critical element in our approach to maximum entropy modeling.

For notational simplicity in this section only, we denote $\eta_{\mathcal{G}_c^{(k)}}$, the variable in the optimization problem at step k, by $\eta^{(k)}$ with the understanding that $\eta^{(k)}$ only contains the parameters corresponding to the nodes and edges of the chordal graph $\mathcal{G}_c^{(k)}$. As described in Section 2.7.2, the key step in the primal-dual algorithm is the computation of the primal-dual search direction by solving a linear system (step 2 of the algorithm). This linear system specialized to the specific MER problem at hand can be restated as follows:

$$M_{pd} \cdot \Delta\eta^{(k)} = r, \tag{3.7}$$

where

$$M_{pd} = G^*_{\mathcal{G}_c^{(k)}}(\eta^{(k)}) + \sum_{v \in V} \lambda_v \left(G_v^*(\eta_v^{(k)}) + \tfrac{1}{a_v} b_v b_v^T \right) + \sum_{E \in \mathcal{E}^{(k)}} \lambda_E \left(G_E^*(\eta_{\bar{E}}^{(k)}) + \tfrac{1}{a_E} b_E b_E^T \right), \tag{3.8}$$

and

$$r = -\Lambda^{-1}_{\mathcal{G}_c^{(k)}}(\eta^{(k)}) - \frac{1}{t} \left(\sum_{v \in V} \frac{1}{a_v} b_v + \sum_{E \in \mathcal{E}^{(k)}} \frac{1}{a_E} b_E \right), \tag{3.9}$$

with $a_v = \delta_v - D_v(\eta_v^{(k)} || \eta_v^*)$, $a_E = \delta_E - D_E(\eta_{\bar{E}}^{(k)} || \eta_{\bar{E}}^*)$, $b_v = \Lambda_v^{-1}(\eta_v^{(k)}) - \Lambda_v^{-1}(\eta_v^*)$, and $b_E = \Lambda_E^{-1}(\eta_{\bar{E}}^{(k)}) - \Lambda_E^{-1}(\eta_{\bar{E}}^*)$. The vectors b_v and b_E, and the matrices G_v^*, G_E^*, $b_v b_v^T$ and $b_E b_E^T$ are low-dimensional; as usual, each term in each of the sums on the right-hand-sides of (3.8) and (3.9) is appropriately zero-padded so that the equations are consistent.

The matrix $G^*_{\mathcal{G}_c^{(k)}}$ in (3.8) is sparse, inheriting the sparse structure of the chordal graph $\mathcal{G}_c^{(k)}$ through (3.6). Furthermore, each additional term in (3.8) also has support nested inside the support of $G^*_{\mathcal{G}_c^{(k)}}$. This is because each element of the vertex set V and of the edge set $\mathcal{E}^{(k)}$ belongs to at least one of the maximal cliques of the chordal graph $\mathcal{G}_c^{(k)}$ (the decomposition in (3.6) includes a sum of local Fisher information matrices over the maximal cliques). Hence, the fill-pattern of M_{pd} is the same as for $G^*_{\mathcal{G}_c^{(k)}}$.

Using the sparse structure of M_{pd}, we can compute $\Delta\eta^{(k)} = M_{pd}^{-1} r$ efficiently. One approach is based on sparse Cholesky factorization [38, 76], where the matrix M_{pd} is factorized as $M_{pd} = AA^T$ with A being lower-triangular. This factorization can be performed tractably by exploiting the sparsity of M_{pd}. Solving the linear system $AA^T \Delta\eta^{(k)} = r$ is then extremely efficient and can be done in two stages using back-substitution [76]; first solve the system $Ax = r$ and then the system $A^T \Delta\eta^{(k)} = x$. This approach requires $\mathcal{O}(|V| \cdot w^6)$ in the Gaussian model. The primal-dual method also requires computation of $\Lambda^{-1}_{\mathcal{G}_c^{(k)}}(\eta^{(k)})$ in equation (3.9), which is given by a tractable computation (3.5). Although we solve only for the subset of moment parameters $\tilde{\eta}^{(k)}$, it is straight-forward to obtain $\tilde{\theta}_{\mathcal{G}_c^{(k)}} \triangleq \Lambda^{-1}_{\mathcal{G}_c^{(k)}}(\tilde{\eta}^{(k)})$, again by (3.5).

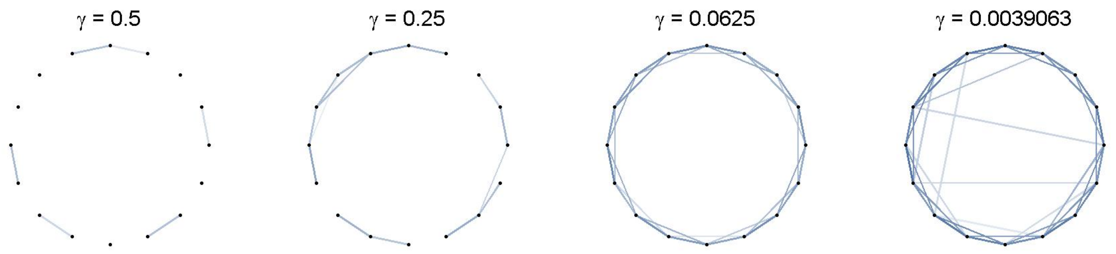


Figure 3-1: Graphs of the MER solution for various values of γ in the Gaussian model.

3.3 Simulation Results

In this section, we describe the results of simulations that demonstrate the effectiveness of the MER framework in learning the Markov structure of Gaussian models from sample data. The tolerance parameters used in the MER problem are set in proportion to the number of parameters needed to specify the marginal distribution as follows:

$$\delta_E = \gamma \cdot \left(|E| + \binom{|E|}{2} \right) = 3\gamma \tag{3.10}$$

$$\delta_v = \gamma. \tag{3.11}$$

Here, $\gamma > 0$ is an overall regularization parameter which controls the trade-off between complexity and accuracy in the resulting MER solution. Our motivation for setting δ_v and δ_E is based on the parametric complexity or, in other words, the number of free parameters in v and E respectively. Similar principles have been developed in the literature on model-order selection (see for example [2]).

In addition, we note that for large sample size the expectation of $D(\eta||\eta^*)$, where η are the actual moments and η^* are empirical, is approximately equal to the number of parameters divided by the number of samples N_s [2]. This suggests that a natural choice for γ in (3.10-3.11) may be $\gamma \sim \frac{1}{N_s}$, or a function that decays slower such as $\gamma \sim \frac{\log(N_s)}{N_s}$. In the following examples, we explore the effect of varying γ.

We describe two sets of experiments. Both simulations are based on 400 samples of test models. In the first experiment, we generate samples from a 16-node cyclic Gaussian model, where the nodes are arranged in a circle and each node is connected to nodes that are one or two steps away on the circle. The node weights are $J_{v,v} = -2\theta_v = 1.0$ for every node v, and the edge weights are $J_{u,v} = -\theta_{uv} = -0.1875$ for each edge $\{u, v\}$. Fig. 3-1 shows the MER solution graphs for various values of γ. Notice that as the value of γ decreases, the effect of the relaxation is smaller and more edges are included in the MER solution. For $\gamma = 0.0625$ the correct underlying graph structure is recovered.

The second experiment involves a 10×10 nearest-neighbor grid-structured model with the node weights being $J_{v,v} = -2\theta_v = 1.0$ for every node v, and the edge weights being $J_{u,v} = -\theta_{uv} = -0.24$ for each edge $\{u, v\}$. Again, 400 samples were generated based on this model and the MER problem is solved for a fixed value of $\gamma = 0.08$. The initial MER problem is solved on the completely disconnected graph with only the 100 node

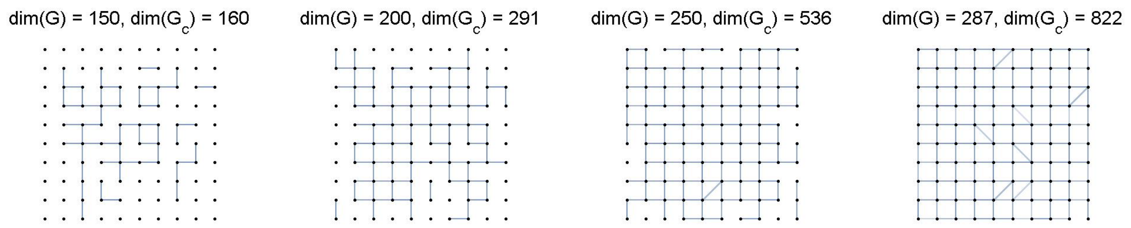

dim(G) = 150, dim(G_c) = 160
dim(G) = 200, dim(G_c) = 291
dim(G) = 250, dim(G_c) = 536
dim(G) = 287, dim(G_c) = 822

Figure 3-2: Illustration of the incremental approach for identifying the set of active constraints in the MER problem by solving a sequence of sub-problems defined on sub-graphs of the final solution (far right). Here, $\dim(\mathcal{G})$ and $\dim(\mathcal{G}_c)$ are the number of nodes plus the number of edges of the constraint graph $\mathcal{G}$ and its chordal super-graph $\mathcal{G}_c$. The number of constraints is equal to $\dim(\mathcal{G})$, and the dimension of $\eta_{\mathcal{G}_c}$ in each MER sub-problem is $\dim(\mathcal{G}_c)$.

constraints. At each successive step (of the incremental procedure of Section 3.2.2), the 50 most violated edge constraints (that are not previously included) are added. Fig. 3-2 demonstrates this incremental approach. The graphs shown correspond to the graphs of the MER solutions of each step of the incremental method, except the first step in which case the graph would be completely disconnected (because only node constraints are imposed in the first step). The dimension (number of node plus edge parameters) of the constraint graphs $\mathcal{G}^{(k)}$, and the associated chordal supergraphs $\mathcal{G}_c^{(k)}$ are shown. After just 5 steps of the incremental approach, the algorithm terminates (all remaining edge constraints that are not included are satisfied) and the underlying graph structure is recovered with only a few spurious or missing edges. We also note that solving the MER problem directly in the complete family (corresponding to the complete graph) without using the incremental approach would be computationally prohibitive because the resulting Fisher information is a $10,000 \times 10,000$ full matrix (equivalently, the matrix M_{pd} in (3.8) is also a $10,000 \times 10,000$ full matrix). Yet, our incremental approach, using a sequence of thin graphs, solves the MER problem exactly in a few minutes by adaptively identifying the worst violated constraints at each step and including very few "extra" constraints.

3.4 Discussion

We have presented a convex optimization approach for learning the graph structure of a collection of random variables from sample data. The formulation is also useful for constructing a tractable graphical model approximation to a specified intractable probability distribution. This method differs from previous approaches that addressed this problem primarily from the point of view of solving a combinatorial optimization problem. Our framework is based on the sparsity-favoring characteristic of entropy (i.e., models with sparse exponential parameters are favored) that is implicit in the maximum entropy principle. We also exploit sparse, tractable computations of the entropy function and its derivatives on thin chordal graphs in order to solve the MER problem using a scalable primal-dual interior point method. We have demonstrated the effectiveness of our approach with simulation results for the Gaussian model. Our method is also suitable for the graphical model selection problem involving discrete variables, by virtue of the abstract formulation with respect to marginalizable exponential families. In Chapter 5, we discuss possible future research directions resulting from the ideas developed in this chapter.

Chapter 4

Efficient Estimation using Tractable Subgraphs: A Walk-Sum Analysis

Letting the conditional distribution of a Gaussian graphical model be parameterized in the information form as $x|y \sim \mathcal{N}^{-1}(h, J)$, the posterior mean can be computed as the solution to the following linear system:

$$J\widehat{x} = h. \tag{4.1}$$

In this chapter, we describe a general class of algorithms to solve this linear system. All these algorithms have the common feature that they solve a sequence of estimation problems on trees or, more generally, tractable subgraphs. We refer to the trees and subgraphs on which inference is performed at each iteration as *preconditioners*, following the terminology used in the linear algebra literature.

We analyze these algorithms based on the walk-sum interpretation of Gaussian inference discussed in Section 2.6. The overall template for analyzing our inference algorithms is simple. First, we show that the algorithms submit to a walk-sum interpretation. Next, we show that the walk-sets computed by these algorithms are nested, i.e. $\mathcal{W}_n \subseteq \mathcal{W}_{n+1}$, where $\mathcal{W}_n$ is the set of walks computed at iteration n. Finally, we show that every walk required for the computation of the mean (2.20) is contained in $\mathcal{W}_n$ for some n. While each step in this procedure is non-trivial, combined together they allow us to conclude that the algorithms converge in walk-summable models.

One of the conditions for walk-summability in Section 2.6 shows that walk-summable models are equivalent to models for which the information matrix is an H-matrix [41, 76]. Several methods for finding good preconditioners for such matrices have been explored in the linear algebra literature, but these have been restricted to either cycling through a fixed set of preconditioners [12] or to so-called "multi-splitting" algorithms [32, 39]. These results do not address the problem of convergence of non-stationary iterations using arbitrary (non-cyclic) sequences of subgraphs. The analysis of such algorithms along with the development of methods to pick a good sequence of preconditioners are the main novel contributions of this chapter, and the recently developed concept of walk-sums is critical to our analysis.

Section 4.1 describes all the algorithms that we analyze in this chapter, while Section 4.2 contains the analysis and walk-sum diagrams that provide interpretations of the algorithms in terms of walk-sum computations. In Section 4.3, we use the walk-sum interpretation of Section 4.2 to show that these algorithms converge in walk-summable models. Section 4.4 presents techniques for choosing tree-based preconditioners and subsets of variables adaptively for the Embedded Trees (ET) and block Gauss-Seidel (GS) iterations respectively, and demonstrates the effectiveness of these methods through simulation. We conclude with a brief summary in Section 4.5. Appendix A provides additional details and proofs. All our algorithms and analysis in this chapter are specified for normalized models (see Section 2.6 for details), although these can be easily extended to the un-normalized case. Appendix A also contains a section that discusses these generalizations.

4.1 Non-Stationary Embedded Subgraph Algorithms

In this section, we describe a framework for the computation of the conditional mean estimates in order to solve the Gaussian estimation problem. We present three algorithms that become successively more complex in nature. We begin with the parallel ET algorithm originally presented in [73, 74]. Next, we describe a serial update scheme that involves processing only a subset of the variables at each iteration. Finally, we discuss a generalization to these non-stationary algorithms that is tolerant to temporary communication failure by using local memory at each node to remember past messages from neighboring nodes. A similar memory-based approach was used in [24] for the special case of stationary iterations. The key theme underlying all these algorithms is that they are based on solving a sequence of inference problems on tractable subgraphs involving all or a subset of the variables. Convergent iterations that compute means can also be used to compute exact error variances [74]. Hence, we restrict ourselves to analyzing iterations that compute the conditional mean.

4.1.1 Non-Stationary Parallel Updates: Embedded Trees Algorithm

Let $\mathcal{S}$ be some subgraph of the graph $\mathcal{G}$. The stationary ET algorithm is derived by splitting the matrix $J = J_{\mathcal{S}} - K_{\mathcal{S}}$, where $J_{\mathcal{S}}$ is known as the *preconditioner* and $K_{\mathcal{S}}$ is known as the *cutting matrix*. Each edge in $\mathcal{G}$ is either an element of $\mathcal{S}$ or $\mathcal{E}\backslash\mathcal{S}$. Accordingly, every non-zero off-diagonal entry of J is either an element of $J_{\mathcal{S}}$ or of $-K_{\mathcal{S}}$. The diagonal entries of J are part of $J_{\mathcal{S}}$. Hence, the matrix $K_{\mathcal{S}}$ is symmetric, zero along the diagonal, and contains non-zero entries only in those locations that correspond to edges not included in the subgraph generated by the splitting. Cutting matrices may have non-zero diagonal entries in general, but we only consider zero-diagonal cutting matrices in this chapter. The splitting of J according to $\mathcal{S}$ transforms (4.1) to $J_{\mathcal{S}}\widehat{x} = K_{\mathcal{S}}\widehat{x}+h$, which suggests a recursive method for solving the original linear system:

$$J_{\mathcal{S}}\widehat{x}^{(n)} = K_{\mathcal{S}}\widehat{x}^{(n-1)} + h. \tag{4.2}$$

If $J_{\mathcal{S}}^{-1}$ exists then a necessary and sufficient condition for the iterates $\{\widehat{x}^{(n)}\}_{n=0}^{\infty}$ to converge to $J^{-1}h$ for any initial guess $\widehat{x}^{(0)}$ is that $\varrho(J_{\mathcal{S}}^{-1}K_{\mathcal{S}}) < 1$ [74]. ET iterations can be very effective if applying $J_{\mathcal{S}}^{-1}$ to a vector is efficient, e.g. if $\mathcal{S}$ corresponds to a tree or, in general, any tractable subgraph.

A non-stationary ET iteration is obtained by letting $J = J_{\mathcal{S}_n} - K_{\mathcal{S}_n}$, where the matrices $J_{\mathcal{S}_n}$ correspond to some embedded tree or subgraph $\mathcal{S}_n$ in $\mathcal{G}$ and can vary in an arbitrary manner with n. This leads to the following ET iteration:

$$J_{\mathcal{S}_n} \widehat{x}^{(n)} = K_{\mathcal{S}_n} \widehat{x}^{(n-1)} + h. \tag{4.3}$$

Our walk-sum analysis proves the convergence of non-stationary ET iterations based on any sequence of subgraphs $\{\mathcal{S}_n\}_{n=1}^{\infty}$ in walk-summable models. Every step of the above algorithm is tractable if applying $J_{\mathcal{S}_n}^{-1}$ to a vector can be performed efficiently. Indeed, an important degree of freedom in the above algorithm is the choice of $\mathcal{S}_n$ at each stage so as to speed up convergence, while keeping the computation at every iteration tractable. We discuss some approaches to addressing this issue in Section 4.4.

4.1.2 Non-Stationary Serial Updates of Subsets of Variables

We begin by describing the block GS iteration [38, 76]. For each $n = 1, 2, \ldots$, let $V_n \subseteq V$ be some subset of V. The variables $x_{V_n} = \{x_s : s \in V_n\}$ are updated at iteration n. The remaining variables do not change from iteration $n-1$ to n. Let $J^{(n)} = [J]_{V_n}$ be the $|V_n| \times |V_n|$-dimensional principal sub-matrix corresponding to the variables V_n. The block GS update at iteration n is as follows:

$$\widehat{x}_{V_n}^{(n)} = {J^{(n)}}^{-1} \left(R_{V_n, V_n^c}\, \widehat{x}_{V_n^c}^{(n-1)} + h_{V_n} \right), \tag{4.4}$$

$$\widehat{x}_{V_n^c}^{(n)} = \widehat{x}_{V_n^c}^{(n-1)}. \tag{4.5}$$

Here, V_n^c refers to the complement of the vertex set V_n. In equation (4.4), R_{V_n, V_n^c} refers to the sub-matrix of edge weights of edges from the vertices V_n^c to V_n. Every step of the above algorithm is tractable as long as applying ${J^{(n)}}^{-1}$ to a vector can be performed efficiently.

We now present a general serial iteration that incorporates an element of the ET algorithm of Section 4.1.1. This update scheme involves a single ET iteration within the induced subgraph of the update variables V_n. We split the edges $\mathcal{E}(V_n)$ in the induced subgraph of V_n into a tractable set $\mathcal{E}_n$ and a set of cut edges $\mathcal{E}(V_n) \backslash \mathcal{E}_n$. Such a splitting leads to a tractable subgraph $\mathcal{S}_n = (V_n, \mathcal{E}_n)$ of the induced subgraph of V_n. That is, the matrix $J^{(n)}$ is split as $J^{(n)} = J_{\mathcal{S}_n} - K_{\mathcal{S}_n}$. This matrix splitting is defined analogous to the splitting in Section 4.1.1. The modified conditional mean update at iteration n is as follows:

$$\widehat{x}_{V_n}^{(n)} = J_{\mathcal{S}_n}^{-1} \left(K_{\mathcal{S}_n}\, \widehat{x}_{V_n}^{(n-1)} + R_{V_n, V_n^c}\, \widehat{x}_{V_n^c}^{(n-1)} + h_{V_n} \right), \tag{4.6}$$

$$\widehat{x}_{V_n^c}^{(n)} = \widehat{x}_{V_n^c}^{(n-1)}. \tag{4.7}$$

Every step of this algorithm is tractable as long as applying $J_{\mathcal{S}_n}^{-1}$ to a vector can be performed efficiently.

The preceding algorithm is a generalization of both the block GS update (4.4)−(4.5) and the non-stationary ET algorithm (4.3), thus allowing for a unified analysis framework. Specifically, by letting $\mathcal{E}_n = \mathcal{E}(V_n)$ for all n above, we obtain the block GS algorithm. On the other hand, by letting $V_n = V$ for all n, we recover the ET algorithm. This hybrid approach also offers a tractable and flexible method for inference in large-scale estimation problems, because it possesses all the benefits of the ET and block GS iterations.

We note that in general application there is one potential complication with both the serial and the parallel iterations presented so far. Specifically, for an arbitrary graphical model with positive-definite information matrix J, the corresponding information sub-matrix $J_{\mathcal{S}_n}$ for some choices of subgraphs $\mathcal{S}_n$ may not be valid, i.e. may have negative eigenvalues[1]. Importantly, this problem *never* arises for walk-summable models, and thus we are free to use any sequence of embedded subgraphs for our iterations and be guaranteed that the computations make sense probabilistically.

Lemma 4.1 *Let J be a walk-summable model, let $\widetilde{V} \subseteq V$, and let $J_{\mathcal{S}}$ be the $|\widetilde{V}| \times |\widetilde{V}|$-dimensional information matrix corresponding to the distribution over some subgraph $\mathcal{S}$ of the induced subgraph $\mathcal{E}(\widetilde{V})$. Then, $J_{\mathcal{S}}$ is walk-summable, and $J_{\mathcal{S}} \succ 0$.*

Proof: For every pair of vertices $s, t \in \widetilde{V}$, it is clear that the walks between s and t in $\mathcal{S}$ are a subset of the walks between these vertices in $\mathcal{G}$, i.e. $\mathcal{W}(s \xrightarrow{\mathcal{S}} t) \subseteq \mathcal{W}(s \rightarrow t)$. Hence, $\bar{\phi}(s \xrightarrow{\mathcal{S}} t) \leq \bar{\phi}(s \rightarrow t) < \infty$, because J is walk-summable. Thus, the model specified by $J_{\mathcal{S}}$ is walk-summable. This allows us to conclude that $J_{\mathcal{S}} \succ 0$ because walk-summability implies validity of a model. $\square$

4.1.3 Distributed Interpretation of (4.6)$-$(4.7) and Communication Failure

We first re-interpret the equations (4.6)$-$(4.7) as local message-passing steps between nodes followed by inference within the subgraph $\mathcal{S}_n$. At iteration n, let κ_n denote the set of *directed* edges in $\mathcal{E}(V_n) \backslash \mathcal{E}_n$ and from V_n^c to V_n:

$$\kappa_n \triangleq \{(s,t) \mid \{s,t\} \in \mathcal{E}(V_n) \backslash \mathcal{E}_n \text{ or } s \in V_n^c, t \in V_n\}. \tag{4.8}$$

The edge set κ_n corresponds to the non-zero elements of the matrices $K_{\mathcal{S}_n}$ and R_{V_n, V_n^c} in equation (4.6). Edges in κ_n are used to communicate information about the values at iteration $n-1$ to neighboring nodes for processing at iteration n.

For each $t \in V_n$, the message $M(s \rightarrow t) = R_{t,s}\, \widehat{x}_s^{(n-1)}$ is sent at iteration n from s to t using the links in κ_n. Let $M_n(t)$ denote the summary of all the messages received at node t at iteration n:

$$M_n(t) = \sum_{\{s \mid (s,t) \in \kappa_n\}} M(s \rightarrow t) = \sum_{\{s \mid (s,t) \in \kappa_n\}} R_{t,s}\, \widehat{x}_s^{(n-1)}. \tag{4.9}$$

Thus, each $t \in V_n$ *fuses* all the information received about the previous iteration and combines this with its local potential value h_t to form a modified potential vector that is then

[1]For example, consider a 5-cycle with each edge having a partial correlation of -0.6. This model is valid (but not walk-summable) with the corresponding J having a minimum eigenvalue of 0.0292. A spanning tree model $J_{\mathcal{S}}$ obtained by removing one of the edges in the cycle, however, is invalid with a minimum eigenvalue of -0.0392.

used for inference within the subgraph $\mathcal{S}_n$:

$$\widehat{x}_{V_n}^{(n)} = J_{\mathcal{S}_n}^{-1} \left(M_n(V_n) + h_{V_n}\right), \tag{4.10}$$

where $M_n(V_n)$ denotes the entire vector of fused messages $M_n(t)$ for $t \in V_n$. An interesting aspect of these message-passing operations is that they are *local* and only nodes that are neighbors in $\mathcal{G}$ may participate in any communication. If the subgraph $\mathcal{S}_n$ is tree-structured, the inference step (4.10) can also be performed efficiently in a distributed manner using only local BP messages [62].

We now present an algorithm that is tolerant to temporary link failure by using local memory at each node t to store the most recent message $M(s \to t)$ received at t from s. If the link (s,t) fails at some future iteration the stored message can be used in place of the new expected message. In order for the overall memory-based protocol to be consistent, we also introduce an additional post-inference message-passing step at each iteration. To make the above points precise, we specify a memory protocol that the network must follow; we assume that each node in the network has sufficient memory to store the most-recent messages received from its neighbors. First, $\mathcal{S}_n$ must not contain any failed links; every link $\{s,t\} \in \mathcal{E}(V_n)$ that fails at iteration n must be a part of the cut-set[2]: $(s,t), (t,s) \in \kappa_n$. Therefore, the links $\mathcal{E}_n$ that are used for the inference step (4.10) must be active at iteration n. Second, in order for nodes to synchronize after each iteration, they must perform a post-inference message-passing step. *After* the inference step (4.10) at iteration n, the variables in V_n must update their neighbors *in the subgraph* $\mathcal{S}_n$. That is, for each $t \in V_n$, a message must be received post-inference from every s such that $\{s,t\} \in \mathcal{E}_n$:

$$M(s \to t) = R_{t,s}\, \widehat{x}_s^{(n)}. \tag{4.11}$$

This operation is possible since the edge $\{s,t\}$ is assumed to active. Apart from these two rules, all other aspects of the algorithm presented previously remain the same. Note that every new message received overwrites the existing stored message, and only the most recent message received is stored in memory.

Thus, link failure affects only equation (4.9) in our iterative procedure. Suppose that a message to be received at $t \in V_n$ from node s is unavailable due to communication failure. The message $M(s \to t)$ from memory can be used instead in the fusion formula (4.9). Let $r_n(s \to t)$ denote the iteration count of the most recent information at node t about variable s at the information fusion step (4.9) at iteration n. In general, $r_n(s \to t) \leq n-1$, with equality if $t \in V_n$ and $(s,t) \in \kappa_n$ is active. With this notation, we can re-write the fusion equation (4.9):

$$M_n(t) = \sum_{\{s|(s,t)\in\kappa_n\}} M(s \to t) = \sum_{\{s|(s,t)\in\kappa_n\}} R_{t,s}\, \widehat{x}_s^{r_n(s\to t)}. \tag{4.12}$$

[2]One way to ensure this is to select $\mathcal{S}_n$ to explicitly avoid the failed links. See Section 4.4.2 for more details.

4.2 Walk-sum interpretation and Walk-sum diagrams

In this section, we analyze each iteration of the algorithms of Section 4.1 as the computation of walk-sums in $\mathcal{G}$. Our analysis is presented for the most general algorithm involving failing links, since the parallel and serial non-stationary updates without failing links are special cases. For each of these algorithms, we then present walk-sum diagrams that provide intuitive, graphical interpretations of the walks being computed. Examples that we discuss include classic methods such as Gauss-Jacobi (GJ) and GS, and iterations involving general subgraphs. Throughout this section, we assume that the initial guess $\widehat{x}^{(0)} = 0$, and we initialize $M(s \to t) = 0$ and $r_1(s \to t) = 0$ for each directed edge $(s,t) \in \mathcal{E}$. In Section 4.3, we prove the convergence of our algorithms for any initial guess $\widehat{x}^{(0)}$.

4.2.1 Walk-sum interpretation

For every pair of vertices $s, t \in V$, we define a recursive sequence of walk-sets. We then show that these walk-sets are exactly the walks being computed by the iterative procedure in Section 4.1.3:

$$\begin{aligned} \mathcal{W}_n(s \to t) &= \mathcal{W}_{r_n(*\to\bullet)}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t) \bigcup \mathcal{W}(s \xrightarrow{\mathcal{S}_n} t), \\ &\qquad s \in V, t \in V_n, \quad (4.13) \\ \mathcal{W}_n(s \to t) &= \mathcal{W}_{n-1}(s \to t),\ s \in V, t \in V_n^c, \quad (4.14) \end{aligned}$$

with

$$\mathcal{W}_0(s \to t) = \emptyset,\ s, t \in V. \tag{4.15}$$

The notation in these equations is defined in Section 2.6.2. $\mathcal{W}_{r_n(*\to\bullet)}(s \to *)$ denotes the walks computed up to iteration $r_n(* \to \bullet)$. $\mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet)$ corresponds to a length-1 walk (called a *hop*) across a directed edge in κ_n. Finally, $\mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t)$ denotes walks within $\mathcal{S}_n$ that end at t. Thus, the first RHS term in (4.13) is the set of previously computed walks that hop across an edge in κ_n, and then propagate within $\mathcal{S}_n$. $\mathcal{W}(s \xrightarrow{\mathcal{S}_n} t)$ is the set of walks that live entirely within $\mathcal{S}_n$. To simplify notation, we define $\phi_n(s \to t) \triangleq \phi(\mathcal{W}_n(s \to t))$. We now relate the walk-sets $\mathcal{W}_n(s \to t)$ to the estimate $\widehat{x}_t^{(n)}$ at iteration n.

Proposition 4.1 *At iteration $n = 0, 1, \ldots$, with $\widehat{x}^{(0)} = 0$, the estimate for node $t \in V$ is given by:*

$$\widehat{x}_t^{(n)} = \sum_{s \in V} h_s \phi_n(s \to t) = \phi_n(h; * \to t), \tag{4.16}$$

where the walk-sum is over the walk-sets defined by (4.13–4.15), and $\widehat{x}_t^{(n)}$ is computed using (4.10,4.12).

This proposition, proven in Appendix A, states that each of our algorithms has a precise walk-sum interpretation. A consequence of this statement is that no walk is over-counted, i.e., each walk in $\mathcal{W}_n$ submits to a unique decomposition with respect to the construction process (4.13–4.15) (see proof for details), and appears exactly once in the sum at each iteration. As discussed in Section 4.3 (Propositions 4.3 and 4.4), the iterative process does even more; the walk-sets at successive iterations are nested and, under an appropriate condition, are "complete" so that convergence is guaranteed for walk-summable models. Showing and understanding all these properties are greatly facilitated by the introduction of a visual representation of how each of our algorithms computes walks, and that is the subject of the next subsection.

4.2.2 Walk-sum diagrams

In the rest of this section, we present a graphical interpretation of our algorithms, and of the walk-sets $\mathcal{W}_n$ (4.13−4.15) that are central to Proposition 4.1 (which in turn is the key to our convergence analysis in Section 4.3). This interpretation provides a clearer picture of memory usage and information flow at each iteration. Specifically, for each algorithm we construct a sequence of graphs $\mathcal{G}^{(n)}$ such that a particular set of walks in these graphs corresponds *exactly* to the sets $\mathcal{W}_n$ (4.13−4.15) computed by the sequence of iterates $\widehat{x}^{(n)}$. The graphs $\mathcal{G}^{(n)}$ are called *walk-sum diagrams*. Recall that $\mathcal{S}_n$ corresponds to the subgraph used at iteration n, generally using some of the values computed from a preceding iteration. The graph $\mathcal{G}^{(n)}$ captures all of these preceding computations leading up to and including the computations at iteration n.

As a result, $\mathcal{G}^{(n)}$ has very specific structure for each algorithm. It consists of a number of *levels* — within each level we capture the subgraph used at the corresponding iteration, and the final level n corresponds to the results at the end of iteration n. Although some variables may not be updated at each iteration, the values of those variables are preserved for use in subsequent iterations; thus, each level of $\mathcal{G}^{(n)}$ includes all the nodes in V. The update variables at any iteration (i.e., the nodes in $\mathcal{S}_n$) are represented as solid circles, and the non-update ones as open circles. All edges in each $\mathcal{S}_n$ — edges of $\mathcal{G}$ included in this subgraph — are included in that level of the diagram. As in $\mathcal{G}$, these are undirected edges, as our algorithms perform inference on this subgraph. However, this inference update uses some values from preceding iterations (4.10,4.12); hence, we use directed edges (corresponding to κ_n) from nodes at preceding levels. The directed nature of these edges is critical as they capture the one-directional flow of computations from iteration to iteration, while the undirected edges within each level capture the inference computation (4.10) at each iteration. At the end of iteration n, only the values at level n are of interest. Therefore, the set of walks (re-weighted by h) in $\mathcal{G}^{(n)}$ that begin at any solid node at any level, and end at any node at the last level are of importance, where walks can only move in the direction of directed edges between levels, but in any direction along the undirected edges within each level.

Later in this section we provide a general procedure for constructing walk-sum diagrams for our most general algorithms, but we begin by illustrating these diagrams and the points made in the preceding paragraph using a simple 3-node, fully connected graph (with variables denoted x_1, x_2, x_3). We look at two of the simplest iterative algorithms in the classes we have described, namely the classic GJ and GS iterations [38, 76]. Figure 4-1 shows the walk-sum diagrams for these algorithms.

In the GJ algorithm each variable is updated at each iteration using the values from the preceding iteration of every other variable (this corresponds to a stationary ET algorithm

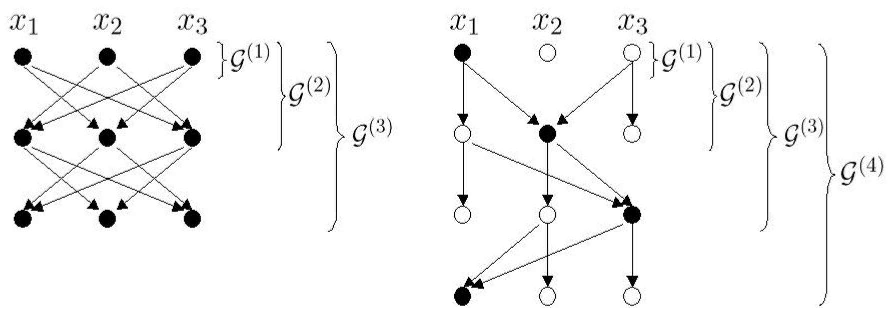


Figure 4-1: (Left) Gauss-Jacobi walk-sum diagrams $\mathcal{G}^{(n)}$ for $n = 1, 2, 3$. (Right) Gauss-Seidel walk-sum diagrams $\mathcal{G}^{(n)}$ for $n = 1, 2, 3, 4$.

(7) with the subgraph $\mathcal{S}_n$ being the fully disconnected graph of all the nodes V). Thus each level on the left in Figure 4-1 is fully disconnected, with solid nodes for all variables and directed edges from each node at the preceding level to every other node at the next level. This provides a simple way of seeing both how walks are extended from one level to the next and, more subtly, how walks captured at one iteration are also captured at subsequent iterations. For example, the walk 12 in $\mathcal{G}^{(2)}$ is captured by the directed edge that begins at node 1 at level 1 and proceeds to node 2 at level 2 (the final level of $\mathcal{G}^{(2)}$). However, this walk in $\mathcal{G}^{(3)}$ is captured by the walk that begins at node 1 at level 2 and proceeds to node 2 at level 3 in $\mathcal{G}^{(3)}$.

The GS algorithm is a serial iteration that updates one variable at a time, cyclically, so that after $|V|$ iterations each variable is updated exactly once. On the right-hand side of Figure 4-1, only one node at each level is solid, using values of the other nodes from the preceding level. For non-update variables at any iteration, a weight-1 directed edge is included from the same node at the preceding level. For example, since x_2 is updated at level 2, we have open circles for nodes 1 and 3 at that level and weight-1 directed edges from their copies at level 1. Weight-1 edges do not affect the weight of any walk. Hence, at level 4 we still capture the walk 12 from level 2 (from node 1 at level 1 to node 2 at level 2); the walk is extended to node 2 at levels 3 and 4 with weight-1 directed edges.

For general graphs, the walk-sum diagram $\mathcal{G}^{(n)}$ of one of our algorithms is constructed as follows:

1. For $n = 1$, create a new copy of each $t \in V$ using solid circles for update variables and open circles for non-update variables; label these $t^{(1)}$. Draw the subgraph $\mathcal{S}_1$ using the solid nodes and undirected edges weighted by the partial correlation coefficient of each edge. $\mathcal{G}^{(1)}$ is the same as $\mathcal{S}_1$ with the exception that $\mathcal{G}^{(1)}$ also contains non-update variables denoted by open circles.

2. Given $\mathcal{G}^{(n-1)}$, create a new copy of each $t \in V$ using solid circles for update variables and open circles otherwise; label these $t^{(n)}$. Draw $\mathcal{S}_n$ using the update variables with undirected edges. Draw a *directed* edge from the variable $u^{r_n(u \to v)}$ in $\mathcal{G}^{(n-1)}$ (since $r_n(u \to v) \leq n - 1$) to $v^{(n)}$ for each $(u, v) \in \kappa_n$. If there are no failed links, $r_n(u \to v) = n - 1$. Both these undirected and directed edges are weighted by their respective partial correlation coefficients. Draw a directed edge to each non-update variable $t^{(n)}$ from the corresponding $t^{(n-1)}$ with unit edge weight.

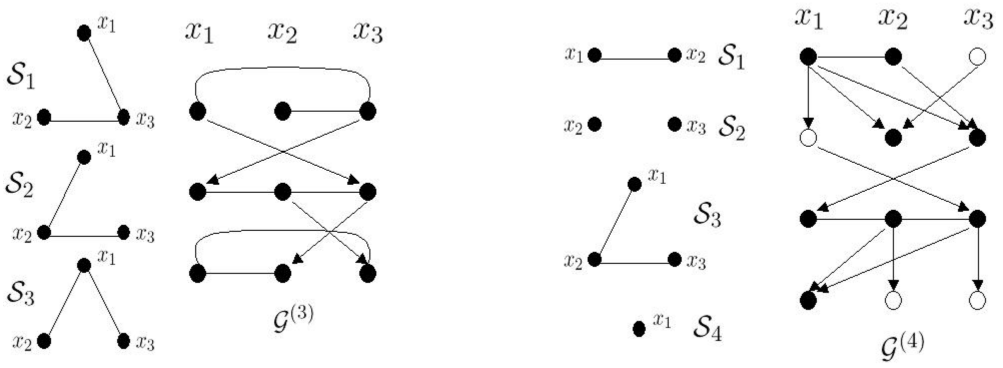


Figure 4-2: (Left) Non-stationary ET: subgraphs and walk-sum diagram. (Right) Hybrid serial updates: subgraphs and walk-sum diagram.

A *level* k in a walk-sum diagram refers to the k'th replica of the variables.

Rules for walks in $\mathcal{G}^{(n)}$: Walks must respect the orientation of each edge, i.e., walks can cross an undirected edge in either direction, but can only cross directed edges in one direction. In addition, walks can only start at the update variables V_k for each level $k \leq n$. Interpreted in this manner, walks in $\mathcal{G}^{(n)}$ re-weighted by h and ending at one of the variables $t^{(k)}$ are exactly the walks computed in $\widehat{x}_t^{(k)}$.

Proposition 4.2 *Let $\mathcal{G}^{(n)}$ be a walk-sum diagram constructed and interpreted according to the preceding rules. For any $t \in V$ and $k \leq n$,*

$$\widehat{x}_t^{(k)} = \phi(h; * \xrightarrow{\mathcal{G}^{(n)}} t^{(k)}). \tag{4.17}$$

Proof: Based on the preceding discussion, one can check the equivalence of the walks computed by the walk-sum diagrams with the walk-sets (4.13$-$4.15). Proposition 4.1 then yields (4.17). □

The following sections describe walk-sum diagrams for the various algorithms presented in Section 4.1.

4.2.3 Non-Stationary Parallel Updates

We describe walk-sum diagrams for the parallel ET algorithm of Section 4.1.1. Here, $V_n = V$ for all n. Since there is no link failure $r_n(* \to \bullet) = n - 1$. Hence, the walk-sum formulas (4.13$-$4.14) reduce to

$$\mathcal{W}_n(s \to t) = \mathcal{W}_{n-1}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t) \bigcup \mathcal{W}(s \xrightarrow{\mathcal{S}_n} t), \; s, t \in V. \tag{4.18}$$

The stationary GJ iteration discussed previously falls in this class. The left-hand side of Figure 4-2 shows the trees $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$, and the corresponding first three levels of the walk-sum diagrams for a more general non-stationary ET iteration. This example illustrates how walks are "collected" in walk-sum diagrams at each iteration. First, walks can proceed along undirected edges within each level, and from one level to the next along directed edges (capturing cut edges). Second, the walks relevant at each iteration must end at that

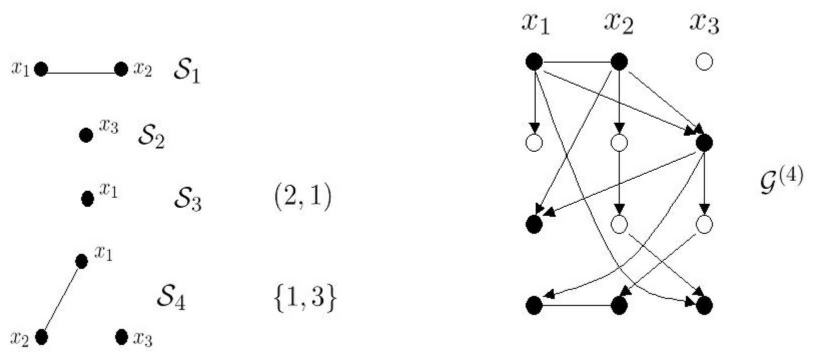


Figure 4-3: Non-stationary updates with failing links: Subgraphs used along with failed edges at each iteration (left) and walk-sum diagram $\mathcal{G}^{(4)}$ (right).

level. For example, the walk 13231 is captured at iteration 1 as it is present in the undirected edges at level 1. At iteration 2, however, we are interested in walks ending at level 2. The walk 13231 is still captured, but in a different manner — through the walk 1323 at level 1, followed by the hop 31 along the directed edge from node 3 at level 1 to node 1 at level 2. At iteration 3, this walk is captured first by the hop from node 1 at level 1 to node 3 at level 2, then by the hop 32 at level 2, followed by the hop from node 2 at level 2 to node 3 at level 3, and finally by the hop 31 at level 3.

4.2.4 Non-Stationary Serial Updates

We describe similar walk-sum diagrams for the serial update scheme of Section 4.1.2. Since there is no link failure, $r_n(* \to \bullet) = n - 1$. The recursive walk-set update (4.13) can be specialized as follows:

$$\mathcal{W}_n(s \to t) = \mathcal{W}_{n-1}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t) \bigcup \mathcal{W}(s \xrightarrow{\mathcal{S}_n} t), \; s \in V, t \in V_n. \tag{4.19}$$

While (4.18) is a specialization to iterations with parallel updates, (4.19) is relevant for serial updates. The GS iteration discussed in Section 4.2.2 falls in this class, as do more general serial updates described in Section 4.1.2 in which we update a subset of variables V_n based on a subgraph of the induced graph of V_n. The right-hand side of Figure 4-2 illustrates an example for our 3-node model. We show the subgraphs $\mathcal{S}_n$ used in the first four stages of the algorithm and the corresponding 4-level walk-sum diagram. Note that at iteration 2 we update variables x_2 and x_3 without taking into account the edge connecting them. Indeed, the updates at the first four iterations of this example include block GS, a hybrid of ET and block GS, parallel ET, and GS, respectively.

4.2.5 Failing links

We now discuss the general non-stationary update scheme of Section 4.1.3 involving failing links. The recursive walk-set computation equations for this iteration are given by (4.13−4.15). Figure 4-3 shows the subgraph and the edges in κ_n that fail at each iteration, and the corresponding 4-level walk-sum diagram. We elaborate on the computation and propagation of information at each iteration. At iteration 1, inference is performed using subgraph $\mathcal{S}_1$, followed by nodes 1 and 2 passing a message to each other according to the post-inference message-passing rule (4.11). At iteration 2 only x_3 is updated. As no links fail, node 3 gets information from nodes 1 and 2 at level 1. At iteration 3, the link $(2, 1)$ fails. But node 1 has information about x_2 at level 1 (due to the post-inference message passing step from iteration 1). This information is used from the local memory at node 1 in (4.12), and is represented by the arrow from node 2 at level 1 to node 1 at level 3. At iteration 4, the links $(1, 3)$ and $(3, 1)$ fail. Similar reasoning as in iteration 3 applies to the arrows drawn across multiple levels from node 1 to node 3, and from node 3 to node 1. Further, post-inference message-passing at this iteration only takes place between nodes 1 and 2 because the only edge in $\mathcal{S}_4$ is $\{1, 2\}$.

4.3 Convergence Analysis

We now show that all the algorithms of Section 4.1 converge in walk-summable models. As in Section 4.2.1, we focus on the most general non-stationary algorithm with failing links of Section 4.1.3. We begin by showing that $\widehat{x}^{(n)}$ converges to the correct means when $\widehat{x}^{(0)} = 0$. Next, we use this result to show that we can achieve convergence to the correct means for any initial guess $\widehat{x}^{(0)}$.

The proof that $\phi_n(h; * \to t) \to (J^{-1}h)_t$ as $n \to \infty$ relies on the fact that $\mathcal{W}_n(s \to t)$ eventually contains every element of the set $\mathcal{W}(s \to t)$ of all the walks in $\mathcal{G}$ from s to t, a condition we refer to as *completeness*. Showing this begins with the following proposition proved in Appendix A.

Proposition 4.3 *(*Nesting*) The walk-sets defined in equations (4.13–4.15) are nested, i.e. for every pair of vertices $s, t \in V$, $\mathcal{W}_{n-1}(s \to t) \subseteq \mathcal{W}_n(s \to t)$ for each n.*

This statement is easily seen for a stationary ET algorithm because the walk-sum diagram $\mathcal{G}^{(n)}$ from levels 2 to n is a replica of $\mathcal{G}^{(n-1)}$ (for example, the GJ diagram in Figure 4-1). However, the proposition is less clear for non-stationary iterations. The discussion in Section 4.2.3 illustrates this point; the paths that a walk traverses change drastically depending on the level in the walk-sum diagram at which the walk ends. Nonetheless, as shown in Appendix A, the structure of the estimation algorithms that we consider ensures that whenever a walk is not explicitly captured in the same form it appeared in the preceding iteration, it is recovered through a different path in the subsequent walk-sum diagram (no walks are lost).

Completeness relies on both nesting and the following additional condition.

Definition 4.1 *Let (u, v) be any directed edge in $\mathcal{G}$. For each n, let $\kappa_n^{active} \subseteq \kappa_n$ denote the set of directed active edges (links that do not fail) in κ_n at iteration n. The edge (u, v) is said to be* updated infinitely often[3] *if for every $N \geq 0$, there exists an $m > N$ such that $(u, v) \in \mathcal{E}_m \cup \kappa_m^{active}$.*

[3]If $\mathcal{G}$ contains a singleton node, then this node must be updated at least once.

If there is no link failure, this definition reduces to including each vertex in V in the update set V_n infinitely often. For parallel non-stationary ET iterations (Section 4.1.1), this property is satisfied for *any sequence of subgraphs*. Note that there are cases in which inference algorithms may not have to traverse each edge infinitely often. For instance, suppose that $\mathcal{G}$ can be decomposed into subgraphs $\mathcal{G}_1$ and $\mathcal{G}_2$ that are connected by a single edge, with $\mathcal{G}_2$ having small size so that we can perform exact computations. For example, $\mathcal{G}_2$ could be a leaf node (i.e., have degree one). We can eliminate the variables in $\mathcal{G}_2$, propagate information "into" $\mathcal{G}_1$ along the single connecting edge, perform inference within $\mathcal{G}_1$, and then back-substitute. Hence, the single connecting edge is traversed only finitely often. In this case the hard part of the overall inference procedure is on the reduced graph with leaves and small, dangling subgraphs eliminated, and we focus on inference problems on such graphs. Thus, we assume that each vertex in $\mathcal{G}$ has degree at least two and study algorithms that traverse each edge infinitely often.

Proposition 4.4 *(*Completeness*) Let $w = s \cdots t$ be an arbitrary walk from s to t in $\mathcal{G}$. If every edge in $\mathcal{G}$ is updated infinitely often (in both directions), then there exists an N such that $w \in \mathcal{W}_n(s \to t)$ for all $n \geq N$, where the walk-set $\mathcal{W}_n(s \to t)$ is defined in (4.13−4.15).*

The proof of this proposition appears in Appendix A. We can now state and prove the following.

Theorem 4.1 *If every edge in the graph $\mathcal{G}$ is updated infinitely often (in both directions), then $\phi_n(h; * \to t) \to (J^{-1}h)_t$ as $n \to \infty$ in walk-summable models, with $\phi_n(s \to t)$ as defined in Section 4.2.1.*

Proof: One can check that $\mathcal{W}_n(s \to t) \subseteq \mathcal{W}(s \to t), \forall n$. This is because equations (4.13−4.15) only use edges from the original graph $\mathcal{G}$. We have from Proposition 4.4 that every walk from s to t in $\mathcal{G}$ is eventually contained in $\mathcal{W}_n(s \to t)$. Thus, $\cup_{n=0}^{\infty} \mathcal{W}_n(s \to t) = \mathcal{W}(s \to t)$. Given these arguments and the nesting of the walk-sets $\mathcal{W}_n(s \to t)$ from Proposition 4.3, we can appeal to the results in Section 2.6.2 to conclude that $\phi_n(h; * \to t) \to (J^{-1}h)_t$ as $n \to \infty$. $\Box$

Theorem 4.1 shows that $\widehat{x}_t^{(n)} \to (J^{-1}h)_t$ for $\widehat{x}^{(0)} = 0$. The following result, proven in Appendix A, shows that in walk-summable models convergence is achieved for any choice of initial condition[4].

Theorem 4.2 *If every edge is updated infinitely often, then $\widehat{x}^{(n)}$ computed according to (4.10,4.12) converges to the correct means in walk-summable models for any initial guess $\widehat{x}^{(0)}$.*

This result shows that walk-summability is a *sufficient* condition for all our algorithms — non-stationary ET, serial updates, memory-based updates — to converge for a very large and flexible set of sequences of tractable subgraphs or subsets of variables (ones that update

[4]Note that in this case the messages must be initialized as $M(s \to t) = R_{t,s}\, \widehat{x}_s^{(0)}$ for each directed edge $(s,t) \in \mathcal{E}$.

each edge infinitely often) on which to perform successive updates. The following result, proven in Appendix A, shows that walk-summability is also *necessary* for this complete flexibility. Thus, while any of our algorithms *may* converge for some sequence of subsets of variables and tractable subgraphs, for a non-walk-summable model there is at least one sequence of updates for which the algorithms diverge.

Theorem 4.3 *For any non-walk-summable model, there exists at least one sequence of iterative steps that is ill-posed, or for which $\widehat{x}^{(n)}$, computed according to (4.10,4.12), diverges.*

4.4 Adaptive Iterations and Experimental Results

In this section we address two topics. The first is taking advantage of the great flexibility in choosing successive iterative steps by developing techniques that adaptively optimize the on-line choice of the next tree or subset of variables to use in order to reduce the error as quickly as possible. The second is providing experimental results that demonstrate the convergence behavior of these adaptive algorithms.

4.4.1 Choosing trees and subsets of variables adaptively

At iteration n, let the *error* be $e^{(n)} = \widehat{x} - \widehat{x}^{(n)}$ and the *residual error* be $h^{(n)} = h - J\,\widehat{x}^{(n)}$. Note that it is tractable to compute the residual error at each iteration.

Trees

We describe an efficient algorithm to choose spanning trees adaptively to use as preconditioners in the ET algorithm of Section 4.1.1. We have the following relationship between the error at iteration n and the residual error at iteration $n-1$:

$$e^{(n)} = (J^{-1} - J_{\mathcal{S}_n}^{-1})\, h^{(n-1)}.$$

Based on this relationship, we have the walk-sum interpretation $e_s^{(n)} = \phi(h^{(n-1)}; * \xrightarrow{\mathcal{G}\backslash\mathcal{S}_n} s)$, and consequently the following bound on the ℓ_1 norm of $e^{(n)}$:

$$\begin{aligned} \|e^{(n)}\|_{\ell_1} &= \sum_{s\in V} \left|\phi(h^{(n-1)}; * \xrightarrow{\mathcal{G}\backslash\mathcal{S}_n} s)\right| \\ &\le \bar{\phi}(|h^{(n-1)}|; \mathcal{G}\backslash\mathcal{S}_n) \\ &= \bar{\phi}(|h^{(n-1)}|; \mathcal{G}) - \bar{\phi}(|h^{(n-1)}|; \mathcal{S}_n), \end{aligned} \tag{4.20}$$

where $\mathcal{G}\backslash\mathcal{S}_n$ denotes walks in $\mathcal{G}$ that must traverse edges not in $\mathcal{S}_n$, $|h^{(n-1)}|$ refers to the entry-wise absolute value vector of $h^{(n-1)}$, $\bar{\phi}(|h^{(n-1)}|; \mathcal{G})$ refers to the re-weighted absolute walk-sum over all walks in $\mathcal{G}$, and $\bar{\phi}(|h^{(n-1)}|; \mathcal{S}_n)$ refers to the re-weighted absolute walk-sum over all walks in $\mathcal{S}_n$. The above inequality becomes an equality for attractive models with a non-negative potential vector h. Minimizing the error $e^{(n)}$ reduces to choosing $\mathcal{S}_n$ to *maximize* $\bar{\phi}(|h^{(n-1)}|; \mathcal{S}_n)$. Hence, if we maximize among all trees, we have the following *maximum walk-sum tree* problem:

$$\text{arg max}_{\mathcal{S}_n \text{ a tree}} \quad \bar{\phi}(|h^{(n-1)}|; \mathcal{S}_n). \tag{4.21}$$

Rather than solving this combinatorially complex problem, we instead solve a problem that minimizes a looser upper bound than (4.20). Specifically, consider any edge $\{u, v\} \in \mathcal{E}$ and all of the walks $\mathcal{S}(u, v) = (uv, vu, uvu, vuv, uvuv, vuvu, \dots)$ that live solely on this single edge. It is not difficult to show that

$$\begin{aligned} w_{u,v} &\triangleq \bar{\phi}(|h^{(n-1)}|; \mathcal{S}(u, v)) \\ &= (|h_u^{(n-1)}| + |h_v^{(n-1)}|) \sum_{\ell=1}^{\infty} |R_{u,v}|^{\ell} \\ &= (|h_u^{(n-1)}| + |h_v^{(n-1)}|) \frac{|R_{u,v}|}{1 - |R_{u,v}|}. \end{aligned} \tag{4.22}$$

This weight provides a measure of the error-reduction capacity of edge $\{u, v\}$ by itself at iteration n. This leads directly to choosing the *maximum spanning tree* [18] by solving

$$\text{arg max}_{\mathcal{S}_n \text{ a tree}} \sum_{\{u,v\} \in \mathcal{S}_n} w_{u,v}. \tag{4.23}$$

For any tree $\mathcal{S}_n$ the set of walks captured in the sum in (4.23) is a subset of all the walks in $\mathcal{S}_n$, so that solving (4.23) provides a lower bound on (4.21) and thus a looser upper bound than (4.20). For sparse graphical models with $|\mathcal{E}| = O(|V|)$, each iteration using this technique requires $O(|V| \log |V|)$ computations [18].

Subsets of variables

We present an algorithm to choose the next best subset of k variables for the block GS algorithm of Section 4.1.2. The error at iteration n can be written as follows:

$$\begin{aligned} e_{V_n}^{(n)} &= \widehat{x}_{V_n} - \widehat{x}_{V_n}^{(n)} = {J^{(n)}}^{-1} R_{V_n, V_n^c} [J^{-1} \, h^{(n-1)}]_{V_n^c}, \\ e_{V_n^c}^{(n)} &= \widehat{x}_{V_n^c} - \widehat{x}_{V_n^c}^{(n)} = e_{V_n^c}^{(n-1)} = [J^{-1} \, h^{(n-1)}]_{V_n^c}. \end{aligned}$$

As with (4.20), we have the following upper bound that is tight for attractive models with non-negative h:

$$\begin{aligned} \|e^{(n)}\|_{\ell_1} &= \|e_{V_n}^{(n)}\|_{\ell_1} + \|e_{V_n^c}^{(n)}\|_{\ell_1} \\ &\leq \left[\bar{\phi}(|h^{(n-1)}|; * \xrightarrow{\mathcal{G}} V_n) - \bar{\phi}(|h^{(n-1)}|; V_n \xrightarrow{\mathcal{E}(V_n)} V_n) \right] + \bar{\phi}(|h^{(n-1)}|; * \xrightarrow{\mathcal{G}} V_n^c) \\ &= \bar{\phi}(|h^{(n-1)}|; \mathcal{G}) - \bar{\phi}(|h^{(n-1)}|; V_n \xrightarrow{\mathcal{E}(V_n)} V_n), \end{aligned} \tag{4.24}$$

where $\mathcal{E}(V_n)$ refers to the edges in the induced subgraph of V_n. Minimizing this upper bound reduces to solving the following *maximum walk-sum block* problem:

$$\text{arg max}_{|V_n|\leq k} \quad \bar{\phi}(|h^{(n-1)}|; V_n \xrightarrow{\mathcal{E}(V_n)} V_n). \tag{4.25}$$

As with the maximum walk-sum tree problem, finding the optimal such block directly is combinatorially complex. Therefore, we consider the following relaxed maximum walk-sum block problem based on single-edge walks:

$$\text{arg max}_{|V_n|\leq k} \quad \bar{\phi}(|h^{(n-1)}|; V_n \xrightarrow{1e} V_n), \tag{4.26}$$

where $\xrightarrow{1e}$ denotes the restriction that walks can traverse at most one edge. The walks in (4.26) are a subset of the walks in (4.25). Thus, solving (4.26) provides a lower bound on (4.25), hence minimizing a looser upper bound on the error than (4.24).

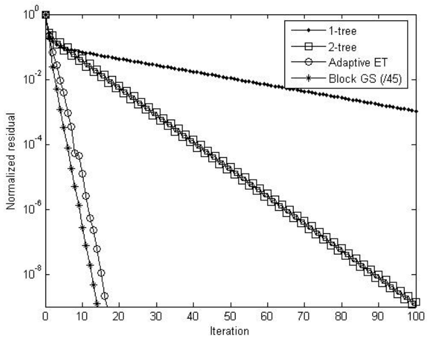
Solving (4.26) is also combinatorially complex; therefore, we use a greedy method for an approximate solution:

1. Set $V_n = \emptyset$. Assuming that the goal is to solve the problem for $k = 1$, compute node weights
$$w_u = |h_u^{(n-1)}|,$$
based on the walks captured by (4.26) if node u were to be included in V_n.

2. Find the maximum weight node u^* from $V \backslash V_n$, and set $V_n \leftarrow V_n \cup u^*$.

3. If $|V_n| = k$, stop. Otherwise, update each neighbor $v \in V \backslash V_n$ of u^* and go to step 2:
$$w_v \leftarrow w_v + \left(|h_{u^*}^{(n-1)}| + |h_v^{(n-1)}|\right) \frac{|R_{u^*,v}|}{1 - |R_{u^*,v}|}.$$
This update captures the extra walks in (4.26) if v were to be added to V_n.

Step 3 is the greedy aspect of the algorithm as it updates weights by computing the extra walks that would be captured in (4.26) if node v were added to V_n, with the assumption that the nodes already in V_n remain unchanged. Note that only the weights of the neighbors of u^* are updated in step 3; thus, there is a bias towards choosing a connected block. In choosing successive blocks in this way, we collect walks adaptively without explicit regard for the objective of updating each node infinitely often. However, our method is biased towards choosing variables that have not been updated for a few iterations as the residual error of such variables becomes larger relative to the other variables. Indeed, empirical evidence confirms this behavior with all the simulations leading to convergent iterations. For sparse graphical models with $|\mathcal{E}| = O(|V|)$ and k bounded, each iteration using this technique requires $O(\log |V|)$ computations using an efficient sort data structure.



Method	Avg. iterations
One-tree	143.07
Two-tree	102.70
Adaptive Max. Spanning Tree	44.04
Adaptive Block Gauss-Seidel (/45)	26.57

Figure 4-4: (Left) Convergence results for a randomly generated 15 x 15 nearest-neighbor grid-structured model. (Right) Average number of iterations required for the normalized residual to reduce by a factor of 10^{-10} over 100 randomly generated models.

Experimental Illustration

We test the preceding two adaptive algorithms on randomly generated 15 x 15 nearest-neighbor grid models with[5] $\varrho(\bar{R}) = 0.99$, and with $\widehat{x}^{(0)} = 0$. The blocks used in block GS were of size $k = 5$. We compare these adaptive methods to standard non-adaptive one-tree and two-tree ET iterations [73]. Figure 4-4 shows the performance of these algorithms. The plot shows the relative decrease in the normalized residual error $\frac{\|h^{(n)}\|_{\ell_2}}{\|h^{(0)}\|_{\ell_2}}$ versus the number of iterations. The table shows the average number of iterations required for these algorithms to reduce the normalized residual error below 10^{-10}. The average was computed based on the performance on 100 randomly generated models. All these models are poorly conditioned because they are barely walk-summable. The number of iterations for block GS is sub-sampled by a factor of $\frac{|V|}{k} = 45$ to provide a fair comparison of the algorithms. The one-tree ET method uses a spanning tree obtained by removing all the vertical edges except the middle column. The two-tree method alternates between this tree and its rotation (obtained by removing all the horizontal edges except the middle row). Figure 4-5 shows the trees computed by our adaptive algorithm in the first four iterations for the same randomly generated 15 x 15 example used to generate the plot in Figure 4-4.

Both the adaptive ET and block GS algorithms provide far faster convergence compared to the one-tree and two-tree iterations, thus providing a computationally attractive method for estimation in the broad class of walk-summable models.

4.4.2 Dealing with Communication Failure: Experimental Illustration

To illustrate our adaptive methods in the context of communication failure, we consider a simple model for a distributed sensor network in which links (edges) fail independently with failure probability α, and each failed link remains inactive for a certain number of iterations given by a geometric random variable with mean $\frac{1}{\beta}$. At each iteration, we find

[5]The grid edge weights are chosen uniformly at random from $[-1, 1]$. The matrix R is then scaled so that $\varrho(\bar{R}) = 0.99$. The potential vector h is chosen to be the all-ones vector.

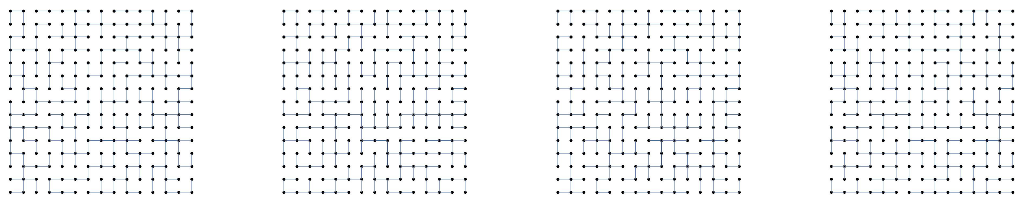


Figure 4-5: Trees chosen at the first four iterations for the same randomly generated 15 x 15 used in Figure 4-4.

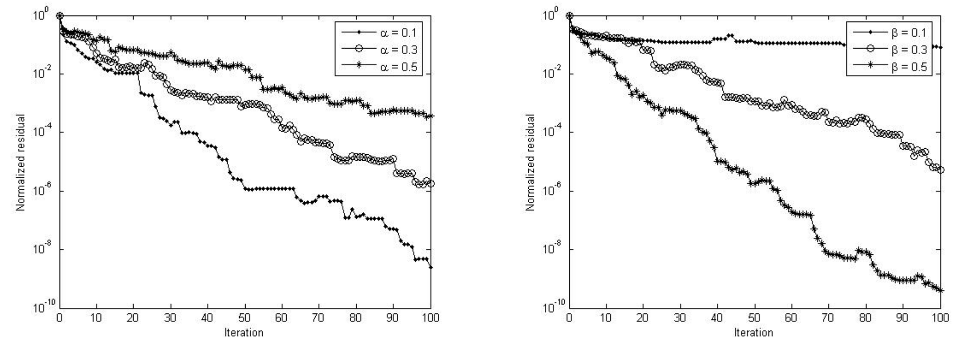


Figure 4-6: Convergence of memory-based algorithm on same randomly generated 15 x 15 used in Figure 4-4: Varying α with $\beta = 0.3$ (left) and varying β with $\alpha = 0.3$ (right).

the best spanning tree (or forest) among the active links using the approach described in Section 4.4.1. The maximum spanning tree problem can be solved in a distributed manner using the algorithms presented in [4, 33]. Figure 4-6 shows the convergence of our memory-based algorithm from Section 4.1.3 on the same randomly generated 15 x 15 grid model used to generate the plot in Figure 4-4 (again, with $\widehat{x}^{(0)} = 0$). The different curves are obtained by varying α and β. As expected, the first plot shows that our algorithm is slower to converge as the failure probability α increases, while the second plot shows that convergence is faster as β is increased (which decreases the average inactive time). These results show that our adaptive algorithms provide a scalable, flexible, and convergent method for the estimation problem in a distributed setting with communication failure.

4.5 Discussion

We have described and analyzed a rich set of algorithms for estimation in Gaussian graphical models with arbitrary structure. These algorithms are iterative in nature and involve a sequence of inference problems on tractable subgraphs over subsets of variables. Our framework includes parallel iterations such as ET, in which inference is performed on a tractable subgraph of the whole graph at each iteration, and serial iterations such as block GS, in which the induced subgraph of a small subset of variables is directly inverted at each iteration. We also describe hybrid versions of these algorithms that involve inference on a subgraph of a subset of variables. In addition, we discuss a method that uses local memory at each node to overcome temporary communication failures that may arise in distributed sensor networks. We analyze these algorithms based on the recently introduced walk-sum interpretation of Gaussian inference. A salient feature in our analysis is the development of walk-sum diagrams, which provide an intuitive graphical comparison between the various algorithms. This walk-sum analysis allows us to conclude that for the large class of walk-summable models, our algorithms converge for essentially any sequence of subgraphs and subsets of variables used. We then describe how this flexibility can be exploited by formulating efficient algorithms that choose spanning trees and subsets of variables adaptively at each iteration. These algorithms are used in the ET and block GS algorithms respectively to demonstrate that significantly faster convergence can be obtained using these methods over traditional one-tree and two-tree ET iterations.

Chapter 5

Conclusion

5.1 Contributions

In this thesis we have studied two central signal processing problems involving Gaussian graphical models, namely modeling and estimation. The modeling problem involves learning a sparse graphical model approximation to a specified distribution. The estimation problem in turn exploits this graph structure to solve high-dimensional estimation problems very efficiently.

Our approach to modeling is based on a convex optimization formulation that maximizes entropy within an exponential family subject to relaxed marginal divergence constraints on small subsets of variables. From the maximum entropy principle, the selection of a sparse graphical model structure arises naturally as a result of solving this problem. We develop a primal-dual interior-point algorithm to solve the optimization problem. A key ingredient that makes this algorithm efficient is the sparsity of the Fisher information matrix in models defined on chordal graphs. In problems involving many variables, we solve a sequence of tractable sub-problems by adaptively identifying and including only the most important constraints (those that are most violated) at each step. Simulation results demonstrate the effectiveness of our approach in learning the Markov structure of some simple models from data.

For the estimation problem, we make explicit use of the graph structure of a model to find the conditional means at each node. Computing these means is efficient if the underlying graph is tractable (such as a tree-structured model). We solve a sequence of estimation problems on such tractable subgraphs to compute the means in an intractable graph. We present a rich class of algorithms that includes the parallel Embedded Trees iteration, the serial block Gauss-Seidel iteration, and hybrid versions of these iterations. In addition, we also consider the case in which links between nodes may occasionally fail; such a framework provides a simple model for communication failure in distributed sensor networks. Our analysis is based on the recently introduced concept of walk-sum interpretation of inference in Gaussian graphical models. We describe the walks "computed" by the algorithms using walk-sum diagrams, and show that convergence can be achieved in walk-summable models for non-stationary iterations based on a very large and flexible set of sequences of subgraphs. Consequently, we are free to choose spanning trees and subsets of variables adaptively at each iteration. This leads to efficient methods for optimizing the next iteration step to achieve maximum reduction in error. Simulation results demonstrate that these non-stationary algorithms provide a significant speedup in convergence over traditional one-tree and two-tree iterations.

5.2 Open research questions

A variety of interesting questions arise from the results presented in this thesis. We discuss some of these here.

5.2.1 Asymptotic analysis of MER

As the number of samples provided increases, one would expect that the MER problem accurately identifies the underlying graph structure of the true model. In order to achieve this, the tolerances must go to zero at an appropriate rate as the number samples grows to infinity. Computing these tolerance decay rates based on results in the large-deviations literature [20, 25] is an important question in order to achieve asymptotic consistency in the MER problem.

The MER problem provides not just an estimate of structure, but also an estimate of the parameters of the (unknown) model. These parameters can be used as an estimate of the true *distribution* based on the given samples. Understanding the generalization error performance of MER in this context, and comparing it to the performance of naive maximum-likelihood estimation of the distribution could be useful in providing theoretical guarantees about the performance of MER.

5.2.2 MER with inconsistent observations

In some applications, observations may be provided over subsets of variables separately, rather than over the entire collection of variables jointly. For this reason, the empirical marginal statistics on overlapping subsets of variables may be inconsistent. Solving the MER problem in this context would require the right set of tolerance parameters for each marginal constraint. In addition to the above issue, one may not have direct access to measurements of variables, but to those of linear combinations of the variables. These linear functionals can correspond to local averages over subsets of variables (e.g. wavelet coefficients [57]). Therefore, the divergence constraints in the MER problem will be with respect to features (given by linear functionals) of the moment vectors rather than the moment vectors directly. Understanding what the appropriate choice of the tolerance parameters should be and studying other aspects of this problem may prove useful for multiscale modeling of regular grid-structured graphical models.

5.2.3 Inference and analysis in non-Gaussian models

The fundamental principle of solving a sequence of tractable inference problems on subgraphs has also been exploited for non-Gaussian inference problems (e.g. [79]). Extending our analysis techniques to better understand such methods is of clear interest. In recent work, the concept of self-avoiding walks has played a central role in providing new insight about inference in discrete models [82]. Algebraic techniques from graph theory [37] may also be useful in further extensions of our analysis to the non-Gaussian case.

5.2.4 Adaptive choice of subgraphs

The adaptive tree and block selection procedures presented in Chapter 4 are greedy in the sense that they only choose the “next-best” subgraph with a view to minimizing the error at the next iteration. Adaptively choosing the K next-best subgraphs *jointly* with the goal of achieving the greatest reduction in error *after* K iteration remains an interesting open problem. Finding such sets of subgraphs jointly may prove to be computationally prohibitive using brute force techniques, but methods from the theory of matroids and submodular functions may provide both practically feasible algorithms as well as theoretical performance guarantees [60].

Appendix A

Proofs for Chapter 4

A.1 Dealing with un-normalized models

Consider an information matrix $J = D - M$ (where D is the diagonal part of J) that is not normalized, i.e. $D \neq I$. The weight of a walk $w = \{w_i\}_{i=0}^{\ell}$ can be re-defined as follows:

$$\psi(w) = \frac{\prod_{i=0}^{\ell-1} M_{w_i,w_{i+1}}}{\prod_{i=0}^{\ell} D_{w_i,w_i}} = \frac{\prod_{i=0}^{\ell-1} \sqrt{D_{w_i,w_i}} R_{w_i,w_{i+1}} \sqrt{D_{w_{i+1},w_{i+1}}}}{\prod_{i=0}^{\ell} D_{w_i,w_i}} = \frac{\phi(w)}{\sqrt{D_{w_0,w_0} D_{w_\ell,w_\ell}}},$$

where $\psi(w)$ is the weight of w with respect to the un-normalized model, and $\phi(w)$ is the weight of w in the corresponding normalized model. We can then define walk-summability in terms of the absolute convergence of the un-normalized walk-sum $\bar{\psi}(s \to t)$ over all walks from s to t (for each pair of vertices $s, t \in V$). A necessary and sufficient condition for this un-normalized notion of walk-summability is $\varrho\left(\overline{D^{-\frac{1}{2}}\ M\ D^{-\frac{1}{2}}}\right) < 1$, which is equivalent to the original condition $\varrho(\bar{R}) < 1$ in the corresponding normalized model. Un-normalized versions of the algorithms in Section 4.1 can be constructed by replacing every occurrence of the partial correlation matrix R by the un-normalized off-diagonal matrix M. The rest of our analysis and convergence results remain unchanged because we deal with abstract walk-sets. (Note that in the proof of Proposition 4.1, every occurrence of R must be changed to M.) Alternatively, given an un-normalized model, one can first normalize the model ($J_{norm} \leftarrow D^{-\frac{1}{2}}\ J_{unnorm}\ D^{-\frac{1}{2}}$), then apply the algorithms of Section 4.1, and finally "de-normalize" the resulting estimate ($\widehat{x}_{unnorm}^{(n)} \leftarrow D^{\frac{1}{2}}\ \widehat{x}_{norm}^{(n)}$). Such a procedure would provide the same estimate as the direct application of the un-normalized versions of the algorithms in Section 4.1 as outlined above.

A.2 Proof of Proposition 4.1

Remarks: Before proceeding with the proof of the proposition, we make some observations about the walk-sets $\mathcal{W}_n(s \to t)$ that will prove useful for the other proofs as well. For $t \in V_n$, notice that since the set of edges contained in $\mathcal{E}_n$ (in subgraph $\mathcal{S}_n$) and κ_n are disjoint, the walk-sets $\mathcal{W}(s \xrightarrow{\mathcal{S}_n} t)$ and $\mathcal{W}_{r_n(* \to \bullet)}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t)$

are disjoint. Therefore, from Section 2.6.2,

$$\begin{aligned}\phi_n(s \to t) &= \phi(s \xrightarrow{\mathcal{S}_n} t) + \phi\left(\mathcal{W}_{r_n(*\to\bullet)}(s \to *) \otimes \mathcal{W}(* \xrightarrow{\kappa_n(1)} \bullet) \otimes \mathcal{W}(\bullet \xrightarrow{\mathcal{S}_n} t)\right) \\ &= \phi(s \xrightarrow{\mathcal{S}_n} t) \\ &\quad + \phi\left(\bigcup_{u,v\in V} \mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)\right). \quad \text{(A.1)}\end{aligned}$$

Every walk $w \in \mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)$ can be *decomposed uniquely* as $w = w_a \cdot w_b \cdot w_c$, where $w_a \in \mathcal{W}_{r_n(u\to v)}(s \to u)$, $w_b \in \mathcal{W}(u \xrightarrow{\kappa_n(1)} v)$, and $w_c \in \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)$. The unique decomposition property is a consequence of $\mathcal{E}_n$ and κ_n being disjoint, and the walk in κ_n being restricted to a length-1 hop. This property also implies that $\mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)$ and $\mathcal{W}_{r_n(u'\to v')}(s \to u') \otimes \mathcal{W}(u' \xrightarrow{\kappa_n(1)} v') \otimes \mathcal{W}(v' \xrightarrow{\mathcal{S}_n} t)$ are disjoint if $(u, v) \neq (u', v')$. Based on these two observations, we have from Section 2.6.2 that

$$\begin{aligned}&\phi\left(\bigcup_{u,v\in V} \mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)\right) \\ &= \sum_{u,v\in V} \phi\left(\mathcal{W}_{r_n(u\to v)}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} t)\right) \\ &= \sum_{u,v\in V} \phi_{r_n(u\to v)}(s \to u)\, \phi(u \xrightarrow{\kappa_n(1)} v)\, \phi(v \xrightarrow{\mathcal{S}_n} t). \quad \text{(A.2)}\end{aligned}$$

Proof of proposition: We provide an inductive proof. From (4.15), $\phi_0(s \to t) = 0$. Thus,

$$\phi_0(h; * \to t) = \sum_{s\in V} h_s\, \phi_0(s \to t) = 0 = \widehat{x}_t^{(0)},$$

which is consistent with the proposition because we assume that our initial guess is 0 at each node.

Assume that $\widehat{x}_t^{(n')} = \phi_{n'}(h; * \to t)$, for $0 \leq n' \leq n-1$, as the inductive hypothesis. For $t \in V_n^c$,

$$\widehat{x}_t^{(n)} = \widehat{x}_t^{(n-1)} = \phi_{n-1}(h; * \to t) = \phi_n(h; * \to t),$$

where the first equality is from (4.7), the second from the inductive hypothesis, and the third from (4.14). Hence, we can focus on nodes in V_n. For $t \in V_n$, (A.1–A.2) can be re-written as:

$$\phi_n(s \to t) = \phi(s \xrightarrow{\mathcal{S}_n} t) + \sum_{(u,v)\in\kappa_n} \phi_{r_n(u\to v)}(s \to u)\, \phi(u \xrightarrow{\kappa_n(1)} v)\, \phi(v \xrightarrow{\mathcal{S}_n} t), \quad \text{(A.3)}$$

because $\phi(u \xrightarrow{\kappa_n(1)} v) = 0$ if $(u,v) \notin \kappa_n$. From (A.1−A.3) we have that:

$$\begin{aligned}
\phi_n(h; * \to t) &= \sum_{s \in V} h_s \left(\phi(s \xrightarrow{\mathcal{S}_n} t) + \sum_{(u,v) \in \kappa_n} \phi_{r_n(u \to v)}(s \to u) \phi(u \xrightarrow{\kappa_n(1)} v) \phi(v \xrightarrow{\mathcal{S}_n} t) \right) \\
&= \sum_{s \in V} h_s \left((J_{\mathcal{S}_n}^{-1})_{t,s} + \sum_{(u,v) \in \kappa_n} \phi_{r_n(u \to v)}(s \to u)\; R_{v,u}\; (J_{\mathcal{S}_n}^{-1})_{t,v} \right),
\end{aligned}$$

where we have used the walk-sum interpretation of $J_{\mathcal{S}_n}^{-1}$ and κ_n. Simplifying further, we have that

$$\begin{aligned}
\phi_n(h; * \to t) &= \left(J_{\mathcal{S}_n}^{-1}\, h_{V_n} \right)_t + \sum_{(u,v) \in \kappa_n} \phi_{r_n(u \to v)}(h; * \to u)\; R_{v,u}\; (J_{\mathcal{S}_n}^{-1})_{t,v} \\
&= \left(J_{\mathcal{S}_n}^{-1}\, h_{V_n} \right)_t + \sum_{(u,v) \in \kappa_n} \widehat{x}_u^{r_n(u \to v)}\; R_{v,u}\; (J_{\mathcal{S}_n}^{-1})_{t,v}. \qquad \text{(A.4)}
\end{aligned}$$

The last equality is from the inductive hypothesis because $0 \leq r_n(u \to v) \leq n-1$. Next, we have that

$$\begin{aligned}
\phi_n(h; * \to t) &= \left(J_{\mathcal{S}_n}^{-1}\, h_{V_n} \right)_t + \sum_{v \in V_n} (J_{\mathcal{S}_n}^{-1})_{t,v} \sum_{\{u | (u,v) \in \kappa_n\}} R_{v,u}\, \widehat{x}_u^{r_n(u \to v)} \\
&= \left(J_{\mathcal{S}_n}^{-1}\, h_{V_n} \right)_t + \sum_{v \in V_n} (J_{\mathcal{S}_n}^{-1})_{t,v}\; M_n(v) \\
&= \widehat{x}_t^{(n)},
\end{aligned}$$

where the second equality is from (4.12), and the third from (4.10). □

A.3 Proof of Proposition 4.3

We prove the following lemma that will be useful later for the proof of the proposition.

Lemma A.1 *Let $w = w_{start} \cdots p \cdot q \cdots w_{end}$ be an arbitrary walk in $\mathcal{W}_n(w_{start} \to w_{end})$, and let $\widetilde{w} = w_{start} \cdots p$ be a* leading *sub-walk of w. There exists a $k_n \leq n$ with $\widetilde{w} \in \mathcal{W}_{k_n}(w_{start} \to p)$ so that at least one of the following conditions is true: $k_n = n$ and the edge $(p,q) \in \mathcal{E}_n$, or $k_n \leq r_n(p \to q)$.*

Proof: The base case is vacuously true because $\mathcal{W}_0(w_{start} \to w_{end}) = \emptyset$. For the inductive hypothesis, assume that the statement is true for $0 \leq n' \leq n-1$. This can be used to prove the statement if $w_{end} \in V_n^c$. Assume that $w \in \mathcal{W}_n(w_{start} \to w_{end})$ with $w_{end} \in V_n$. From the remarks in Section A.2 of this appendix, either $w \in \mathcal{W}(w_{start} \xrightarrow{\mathcal{S}_n} w_{end})$, or $w \in \mathcal{W}_{r_n(u \to v)}(w_{start} \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} w_{end})$ for some unique pair of vertices $u, v \in V$ with $r_n(u \to v) \leq n-1$. If $w \in \mathcal{W}(w_{start} \xrightarrow{\mathcal{S}_n} w_{end})$, then $k_n = n$ and $(p,q) \in \mathcal{E}_n$.

If $w \in \mathcal{W}_{r_{n}(u\to v)}(w_{start} \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} w_{end})$, then from the remarks in Section A.2 of this appendix, w can be uniquely decomposed as $w = w_a \cdot w_b \cdot w_c$ with $w_a \in \mathcal{W}_{r_n(u\to v)}(w_{start} \to u)$, $w_b = uv \in \mathcal{W}(u \xrightarrow{\kappa_n(1)} v)$, and $w_c \in \mathcal{W}(v \xrightarrow{\mathcal{S}_n} w_{end})$. Suppose the trailing part $p \cdots w_{end}$ is a sub-walk of w_c, or is equal to w_c. We can uniquely decompose $\widetilde{w}$ as $w_a \cdot w_b \cdot (v \cdots p) \in \mathcal{W}_{r_n(u\to v)}(w_{start} \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_n(1)} v) \otimes \mathcal{W}(v \xrightarrow{\mathcal{S}_n} p)$. This shows that $k_n = n$. Also, $(p,q) \in \mathcal{E}_n$ because $w_c \in \mathcal{W}(v \xrightarrow{\mathcal{S}_n} w_{end})$.

Suppose $p \cdots w_{end}$ is not a sub-walk of w_c; then either $\widetilde{w} = w_a$ or $\widetilde{w}$ must be a leading sub-walk of w_a. If $\widetilde{w} = w_a$, then $(p,q) = (u,v)$ and $k_n = r_n(p \to q)$. If $\widetilde{w}$ is a leading sub-walk of w_a, we can use the inductive hypothesis (because $r_n(u \to v) \le n-1$) to obtain a $k_n = k_{r_n(u\to v)} \le r_n(u \to v) < n$. If $k_n = k_{r_n(u\to v)} = r_n(u \to v)$, then $(p,q) \in \mathcal{E}_{r_n(u\to v)}$ and one can check that $r_n(p \to q) \ge k_n$ (because a post-inference message is passed on edge (p,q) at iteration $r_n(u \to v) = k_n$). Otherwise, $k_n = k_{r_n(u\to v)} \le r_{r_n(u\to v)}(p \to q) \le r_n(p \to q)$. $\square$

Proof of proposition: We provide an inductive proof. Let any two vertices $s, t \in V$ be given. The base case $\mathcal{W}_0(s \to t) \subseteq \mathcal{W}_1(s \to t)$ clearly follows from the fact that $\mathcal{W}_0(s \to t) = \emptyset$ from (4.13). For the inductive hypothesis, assume that $\mathcal{W}_{n'-1}(s \to t) \subseteq \mathcal{W}_{n'}(s \to t)$ for $0 \le n' \le n-1$. If $t \in V_n^c$, the proposition follows because $\mathcal{W}_n(s \to t) = \mathcal{W}_{n-1}(s \to t)$ from (4.14). So we can restrict ourselves to the case that $t \in V_n$. Let some $w \in \mathcal{W}_{n-1}(s \to t)$ be given.

First, we check if $w \in \mathcal{W}(s \xrightarrow{\mathcal{S}_n} t)$. If this is the case, then we are done. If not, w can be uniquely decomposed as $w = w_a \cdot w_b \cdot w_c$, where $w_b \in \mathcal{W}(p \xrightarrow{\kappa_n(1)} q)$, and $w_c \in \mathcal{W}(q \xrightarrow{\mathcal{S}_n} t)$ for some $p, q \in V$. We must show that $w_a \in \mathcal{W}_{r_n(p\to q)}(s \to p)$. But w_a is a leading sub-walk of w. We have from Lemma A.1 that, with respect to the walk-set $\mathcal{W}_{n-1}(s \to t)$, there exists a $k_{n-1} \le n-1$ such that $w_a \in \mathcal{W}_{k_{n-1}}(s \to p)$. If $k_{n-1} = n-1$, then $(p,q) \in \mathcal{E}_{n-1}$ and $r_n(p \to q) = n-1$ (due to post-inference message (4.11)). Hence, $w_a \in \mathcal{W}_{k_{n-1}}(s \to p) = \mathcal{W}_{r_n(p\to q)}(s \to p)$. If $k_{n-1} < n-1$, then $k_{n-1} \le r_{n-1}(p \to q)$ from Lemma A.1. But $k_{n-1} \le r_{n-1}(p \to q) \le r_n(p \to q) \le n-1$ and we can apply the inductive hypothesis to show the relation $\mathcal{W}_{k_{n-1}}(s \to p) \subseteq \mathcal{W}_{r_n(p\to q)}(s \to p)$. Thus, $w_a \in \mathcal{W}_{k_{n-1}}(s \to p) \subseteq \mathcal{W}_{r_n(p\to q)}(s \to p)$. $\square$

A.4 Proof of Proposition 4.4

Let $w = s \cdots u \cdot t$. We provide an inductive proof with respect to the length of w. If every edge is updated infinitely often, it is clear that every node is updated infinitely often. Therefore, the leading length-0 part (s) is computed when s is first updated at some iteration k. By the nesting of the walk-sets $\mathcal{W}_n$ from proposition 4.3, we have that $(s) \in \mathcal{W}_{k'}(s \to s)$ for all $k' \ge k$. Now assume (as the inductive hypothesis) that the leading sub-walk $s \cdots u$ including all but the last step $u \cdot t$ of w is contained in $\mathcal{W}_N(s \to u)$ for some N $(\ge k)$. Given the infinitely-often update property, there exists an $m > N$ such that the edge $(u,t) \in \mathcal{E}_m \cup \kappa_m^{active}$. If $(u,t) \in \kappa_m^{active}$, then $w \in \mathcal{W}_{m-1}(s \to u) \otimes \mathcal{W}(u \xrightarrow{\kappa_m(1)} t) \otimes \mathcal{W}(t \xrightarrow{\mathcal{S}_m} t) \in \mathcal{W}_m(s \to t)$. This can be concluded from (4.13) and because $s \cdots u \in \mathcal{W}_{m-1}(s \to u)$ by the nesting argument $(m-1 \ge N)$ of Proposition 4.3. Again applying the nesting argument, we can prove the proposition because we now have that $w \in \mathcal{W}_n(s \to t)$ for all $n \geq m$. We can use a similar argument to conclude that $w \in \mathcal{W}_n(s \to t)$ for all $n \geq m$, if $(u,t) \in \mathcal{E}_m$. $\square$

A.5 Proof of Theorem 4.2

From Theorem 4.1 and Proposition 4.1, we can conclude that $\widehat{x}^{(n)}$ converges to $J^{-1}h$ element-wise as $n \to \infty$ for $\widehat{x}^{(0)} = 0$. Assume that $\widehat{x}^{(0)} \neq 0$. Consider a shifted linear system $J\widehat{y} = \tilde{h}$, where $\tilde{h} = h - J\widehat{x}^{(0)}$. If we solve this system using the same sequence of operations (subgraphs and failed links) that were used to obtain the iterates $\widehat{x}^{(n)}$, and with $\widehat{y}^{(0)} = 0$, then $\widehat{y}^{(n)}$ converges to the correct solution $J^{-1}h - \widehat{x}^{(0)}$ of the system $J\widehat{y} = \tilde{h}$. We will show that $\widehat{y}^{(n)} = \widehat{x}^{(n)} - \widehat{x}^{(0)}$, which would allow us to conclude that $\widehat{x}^{(n)} \to J^{-1}h$ element-wise as $n \to \infty$ for any $\widehat{x}^{(0)}$. We prove this final step inductively. The base case is clear because $\widehat{y}^{(0)} = 0$. Assume as the inductive hypothesis that $\widehat{y}^{(n')} = \widehat{x}^{(n')} - \widehat{x}^{(0)}$ for $0 \leq n' \leq n-1$. From this, one can check that $\widehat{y}^{(n)}_{V_n^c} = \widehat{x}^{(n)}_{V_n^c} - \widehat{x}^{(0)}_{V_n^c}$. For $t \in V_n$, we have from (4.10,4.12) that:

$$\begin{aligned}
\widehat{y}^{(n)}_t &= (J^{-1}_{\mathcal{S}_n} \cdot \tilde{h}_{V_n})_t + \left(\sum_{(u,v) \in \kappa_n} (J^{-1}_{\mathcal{S}_n})_{t,v} \cdot R_{v,u} \cdot \widehat{y}^{r_n(u \to v)}_u \right) \\
&= (J^{-1}_{\mathcal{S}_n} \cdot h_{V_n})_t + \left(\sum_{(u,v) \in \kappa_n} (J^{-1}_{\mathcal{S}_n})_{t,v} \cdot R_{v,u} \cdot \widehat{x}^{r_n(u \to v)}_u \right) \\
&\qquad - \left(J^{-1}_{\mathcal{S}_n} \cdot (J\widehat{x}^{(0)})_{V_n} \right)_t - \left(\sum_{(u,v) \in \kappa_n} (J^{-1}_{\mathcal{S}_n})_{t,v} \cdot R_{v,u} \cdot \widehat{x}^{(0)}_u \right) \\
&= \widehat{x}^{(n)}_t - \left(J^{-1}_{\mathcal{S}_n} \cdot (J_{V_n,V_n^c} \cdot \widehat{x}^{(0)}_{V_n^c} + J_{V_n,V_n} \cdot \widehat{x}^{(0)}_{V_n} + K_{\mathcal{S}_n} \cdot \widehat{x}^{(0)}_{V_n} + R_{V_n,V_n^c} \cdot \widehat{x}^{(0)}_{V_n^c}) \right)_t \\
&= \widehat{x}^{(n)}_t - \widehat{x}^{(0)}_t.
\end{aligned}$$

The second equality follows from the inductive hypothesis, and the last two from simple algebra. $\square$

A.6 Proof of Theorem 4.3

Before proving the converse, we have the following lemma that is proved in [1].

Lemma A.2 *Suppose J is a symmetric positive-definite matrix, and $J = J_{\mathcal{S}} - K_{\mathcal{S}}$ is some splitting with $K_{\mathcal{S}}$ symmetric and $J_{\mathcal{S}}$ non-singular. Then, $\varrho(J_{\mathcal{S}}^{-1}K_{\mathcal{S}}) < 1$ if and only if $J + 2K_{\mathcal{S}} \succ 0$.*

Proof of converse: Assume that $J = I - R$ is valid but non-walk-summable. Therefore, R must contain some negative partial correlation coefficients (since all valid attractive models, i.e. those containing only non-negative partial correlation coefficients, are walk-summable; see Section 2.6.2). Let $R = R_+ + R_-$ with R_+ containing the positive coefficients and R_- containing the negative coefficients (including the negative sign). Consider a stationary ET iteration (4.2) based on the cutting the negative edges so that $J_{\mathcal{S}} = I - R_+$ and $K_{\mathcal{S}} = R_-$. If $J_{\mathcal{S}}$ is singular, then the iteration is ill-posed. Otherwise, the iteration converges if and only if $\varrho(J_{\mathcal{S}}^{-1} K_{\mathcal{S}}) < 1$ [38, 76]. From Lemma A.2, we need to check the validity of $J + 2K_{\mathcal{S}}$:

$$J + 2K_{\mathcal{S}} = I - R + 2R_- = I - \bar{R}.$$

But $I - \bar{R} \succ 0$ if and only if the model is walk-summable (from Section 2.6.2). Thus, this stationary iteration, if well-posed, does not converge in non-walk-summable models. $\square$

Appendix B

Fisher information in Gaussian models

In this section, we provide a brief derivation of the Fisher information matrix with respect to the moment parameters η in the complete Gaussian model, i.e. a model with moments specified for all vertices V and all pairs of vertices $\binom{V}{2}$.

The entropy of a collection of normally distributed variables with covariance P is given by

$$H(P) = \tfrac{1}{2}(\log\det P + |V| \cdot \log 2\pi e).$$

The gradient of $H(P)$ with respect to P is [10]

$$\nabla_P H(P)(\triangle P) = \tfrac{1}{2}\mathrm{trace}(P^{-1}\triangle P).$$

Further, the Hessian of $H(P)$ with respect to P is [10]

$$\nabla_P^2 H(P)(\triangle P, \triangle Q) = -\tfrac{1}{2}\mathrm{trace}(P^{-1}\triangle P P^{-1}\triangle Q). \tag{B.1}$$

Based on this Hessian formula we can derive the Fisher information in Gaussian models with respect to the moment parameters η.

First, we recall from Section 2.3.2 that the Hessian of entropy with respect to the moment parameters is the negative Fisher information matrix with respect to moment parameterization:

$$\nabla_\eta^2 H(\eta) = -G^*(\eta), \tag{B.2}$$

where $H(\eta)$ refers to entropy parameterized by the *moment parameters* η rather than the covariance matrix P. Second, the relation between η and $P(\eta)$ is as follows:

$$P(\eta) = \sum_{s \in V} \eta_s \mathbf{e}_s \mathbf{e}_s^T + \sum_{\{s,t\} \in \binom{V}{2}} \eta_{st}(\mathbf{e}_s \mathbf{e}_t^T + \mathbf{e}_t \mathbf{e}_s^T), \tag{B.3}$$

where $\mathbf{e}_s \in \mathbb{R}^{|V|}$ is the vector with a 1 corresponding to the location of vertex s and zero everywhere else. Hence, $\mathbf{e}_s \mathbf{e}_s^T$ and $\mathbf{e}_s \mathbf{e}_t^T + \mathbf{e}_t \mathbf{e}_s^T$ can be viewed as “basis” vectors that are used to construct a covariance matrix given “weights” η_s and η_{st}.

From (B.1−B.3), the element $G^*(\eta)_{st,uv}$ of the Fisher information matrix is given by:

$$\begin{aligned} G^*(\eta)_{st,uv} &= \tfrac{1}{2}\mathrm{trace}\left[P(\eta)^{-1}(\mathbf{e}_s\mathbf{e}_t^T + \mathbf{e}_t\mathbf{e}_s^T)P(\eta)^{-1}(\mathbf{e}_u\mathbf{e}_v^T + \mathbf{e}_v\mathbf{e}_u^T)\right] \\ &= J_{s,u}J_{t,v} + J_{s,v}J_{t,u}, \end{aligned}$$

where $J = P(\eta)^{-1}$. Using similar calculations, we have that $G^*(\eta)_{st,u} = J_{s,u}J_{t,u}$ and $G^*(\eta)_{s,t} = \frac{1}{2}J_{s,t}^2$.

Bibliography

[1] L. Adams. m-Step Preconditioned Conjugate Gradient methods. *SIAM Journal on Scientific and Statistical Computing*, 6:452–463, April 1985.

[2] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, December 1974.

[3] S. Amari. Information geometry on a hierarchy of probability distributions. *IEEE Transactions on Information Theory*, 47(5):1701–1711, July 2001.

[4] B. Awerbuch. Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems. In *Annual ACM Symposium on Theory of Computing*, 1987.

[5] F. Bach and M. Jordan. Thin junction trees. In *Proc. Neural Information Processing Systems*, 2001.

[6] O. Banerjee, L. Ghaoui, A. d'Aspremont, and G. Natsoulis. Convex optimization tehniques for fitting sparse Gaussian graphical models. In *International Conference on Machine Learning*, 2006.

[7] A. Berry, J. R. S. Blair, P. Heggernes, and B. W. Peyton. Maximum Cardinality Search for Computing Minimal Triangulations of Graphs. *Algorithmica*, 39(4):287–298, May 2004.

[8] D. P. Bertsekas, A Nedic, and A. E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.

[9] B. Bollobás. *Modern Graph Theory*. Springer, 1998.

[10] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[11] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Physics*, 7(3):200–217, February 1967.

[12] R. Bru, F. Pedroche, and D. B. Szyld. Overlapping Additive and Multiplicative Schwarz iterations for H-matrices. *Linear Algebra and its Applications*, 393:91–105, 2004.

[13] E. Castillo, J. M. Gutierrez, and A. S. Hadi. *Expert Systems and Probabilistic Network Models*. Springer, 1996.

[14] V. Chandrasekaran, J. K. Johnson, and A. S. Willsky. Estimation in Gaussian Graphical Models using Tractable Subgraphs: A Walk-Sum Analysis. January 2007. submitted.

[15] N. N. Chentsov. A systematic theory of exponential families of probability distributions. *Theory of Probability and its Applications*, 11, 1966.

[16] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. Information Theory*, 14(3), May 1968.

[17] W. C. Chu. *Speech Coding Algorithms: Foundation and Evolution of Standardized Coders*. Wiley-Interscience, 2003.

[18] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. Wiley-Interscience, New York, 1998.

[19] P. H. Cootner, editor. *The Random Character of Stock Market Prices*. MIT press, 1964.

[20] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Interscience, 1991.

[21] C. Crick and A. Pfeffer. Loopy Belief Propagation as a basis for communication in sensor networks. In *Uncertainty in Artificial Intelligence*, 2003.

[22] I. Csiszár. I-divergence geometry of probability distributions and minimization problems. *Annals of Probability*, 3(1):146–158, February 1975.

[23] G. B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963.

[24] V. Delouille, R. Neelamani, and R. Baraniuk. Robust Distributed Estimation using the Embedded Subgraphs Algorithm. *IEEE Transactions on Signal Processing*, 54:2998–3010, August 2006.

[25] A. Dembo and O. Zeitouni. *Large Deviations Techniques and Applications*. Academic Press, 1999.

[26] A. Dempster. Covariance selection. *Biometrics*, 28(1), 1972.

[27] R. Diestel. *Graph Theory*. Springer, 2000.

[28] M. Dudik and R. Schapire. Maximum entropy distribution estimation with generalized regularization. In *Conference on Learning Theory*, 2006.

[29] P. Fieguth, W. Karl, and A. Willsky. Efficient multiresolution counterparts to variational methods for surface reconstruction. *Computer Vision and Image Understanding*, 70(2):157–176, May 1998.

[30] P. W. Fieguth, W. C. Karl, A. S. Willsky, and C. Wunsch. Multiresolution optimal interpolation and statistical analysis of TOPEX/POSEIDON satellite altimetry. *IEEE Transactions on Geoscience and Remote Sensing*, 33:280–292, March 1995.

[31] A. B. Frakt, H. Lev-Ari, and A. S. Willsky. A generalized Levinson algorithm for covariance extension with application to multiscale autoregressive modeling. *IEEE Transactions on Information Theory*, 49(2):411–424, February 2003.

[32] A. Frommer and D. B. Szyld. H-Splittings and Two-stage iterative methods. *Numerische Mathematik*, 63:345–356, 1992.

[33] R. G. Gallager, P. A. Humblet, and P. M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Transactions on Programming Languages and Systems*, 5(1):66–77, January 1983.

[34] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5:721–741, June 1984.

[35] J. W. Gibbs. *Elementary Principles of Statostical Mechanics*. Yale University press, 1902.

[36] R. Godement. *Analysis I*. Springer-Verlag, 2004.

[37] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer, 2001.

[38] G. H. Golub and C. H. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, Maryland, 1990.

[39] T. Gu, X. Liu, and X. Chi. Relaxed Parallel Two-Stage Multisplitting Methods II: Asynchronous Version. *International Journal of Computer Mathematics*, 80(10):1277–1287, October 2003.

[40] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, U.K., 1990.

[41] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, U.K., 1994.

[42] E. Jaynes. Information theory and statistical mechanics. *Physical Review*, 16(4), 1957.

[43] F. Jensen and F. Jensen. Optimal Junction Trees. In *Uncertainty in Artificial Intelligence*, 1994.

[44] J. W. John. *Multidimensional Signal, Image, and Video Processing and Coding*. Academic press, 2006.

[45] J. K. Johnson, V. Chandrasekaran, and A. S. Willsky. Learning Markov Structure by Maximum Entropy Relaxation. In *Artificial Intelligence and Statistics*, March 2007.

[46] M. I. Jordan. *An Introduction to Graphical Models*. MIT press. to be published.

[47] M. I. Jordan. Graphical Models. *Statistical Science (Special Issue on Bayesian Statistics)*, 19:140–155, 2004.

[48] T. Kailath, A. H. Sayed, and B. Hassibi. *Linear Estimation*. Prentice Hall, 2000.

[49] D. Karger and N. Srebro. Learning Markov networks: maximum bounded tree-width graphs. In *Symposium on Discrete Algorithms*, 2001.

[50] W. Karush. Minima of Functions of Several Variables with Inequalities as Side Constraints. Master's thesis, University of Chicago, 1939.

[51] U. Kjaerulff. Reduction of computational complexity in bayesian networks through removal of weak dependencies. In *Proc. Uncertainty in Artificial Intelligence*, 1994.

[52] H. W. Kuhn and A. W. Tucker. Nonlinear Programming. In *Proceedings of Second Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, 1951.

[53] S. L. Lauritzen. *Graphical Models*. Oxford University Press, Oxford, U.K., 1996.

[54] S. Lee, V. Ganapathi, and D. Koller. Efficient structure learning of Markov networks using ℓ_1 regularization. In *Neural Information Processing Systems*, 2006.

[55] Z. Liang and P. C. Lauterbur. *Principles of Magnetic Resonance Imaging: A Signal Processing Perspective*. Wiley-IEEE press, 1999.

[56] D. M. Malioutov, J. K. Johnson, and A. S. Willsky. Walk-Sums and Belief Propagation in Gaussian Graphical Models. *Journal of Machine Learning Research*, 7:2003–2030, October 2006.

[57] S. Mallat. *A Wavelet Tour of Signal Processing*. Springer, 1998.

[58] R. J. McEliece, D. J. C. McKay, and J. F. Cheng. Turbo decoding as an instance of Pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communication*, 16(2):140–152, February 1998.

[59] M. Narasimhan and J. Bilmes. Optimal sub-graphical models. In *Proc. Neural Information Processing Systems*, 2005.

[60] L. Nemhauser and G. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience, 1999.

[61] G. Parisi. *Statistical Field Theory*. Addison-Wesley, 1988.

[62] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kauffman, San Mateo, CA, 1988.

[63] K. Plarre and P. Kumar. Extended message passing algorithm for inference in loopy gaussian graphical models. *Ad Hoc Networks*, 2004.

[64] H. V. Poor and G. W. Wornell, editors. *Wireless Communications: Signal Processing Perspectives*. Prentice Hall, 1998.

[65] R. T. Rockafellar. *Convex Analysis*. Princeton University press, 1996.

[66] W. Rudin. *Principles of Mathematical Analysis*. Mc-Graw Hill, New York, 1976.

[67] P. Rusmevichientong and B. Van Roy. An Analysis of Turbo Decoding with Gaussian densities. In *Neural Information Processing Systems*, 2000.

[68] L. K. Saul and M. I. Jordan. Exploiting Tractable Substructures in Intractable Networks. In *Neural Information Processing Systems*, 1995.

[69] L. Scharf. *Statistical Signal Processing*. Prentice-Hall, Upper Saddle River, NJ, 2002.

[70] R. Shamir. Advanced topics in graph algorithms. Technical report, Telaviv University, 1994.

[71] T. Speed and H. Kiiveri. Gaussian Markov probability distributions over finite graphs. *Annals of Statistics*, 14(1):138–150, March 1986.

[72] W. G. Strang and G. J. Fix. *An Analysis of the Finite Element method*. Wellesley Cambridge Press, 1973.

[73] E. B. Sudderth. Embedded Trees: Estimation of Gaussian Processes on Graphs with Cycles. Master's thesis, Massachusetts Institute of Technology, 2002.

[74] E. B. Sudderth, M. J. Wainwright, and A. S. Willsky. Embedded Trees: Estimation of Gaussian processes on graphs with cycles. *IEEE Transactions on Signal Processing*, 52(11):3136–3150, November 2004.

[75] R. Szeliski. Bayesian modeling of uncertainty in low-level vision. *Journal of Computer Vision*, 5(3):271–301, November 1990.

[76] R. S. Varga. *Matrix Iterative Analysis*. Springer-Verlag, New York, 2000.

[77] M. Wainwright, P. Ravikumar, and J. Lafferty. Inferring graphical model structure using ℓ_1-regularized pseudo-likelihood. In *Neural Information Processing Systems*, 2006.

[78] M. J. Wainwright. *Stochastic Processes on Graphs with Cycles: Geometric and Variational approaches*. PhD thesis, Massachusetts Institute of Technology, 2002.

[79] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 49:1120–1146, May 2003.

[80] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, University of California - Berkeley, September 2003.

[81] Y. Weiss and W. T. Freeman. Correctness of belief propagation in gaussian graphical models of arbitrary topology. *Neural Computation*, 13:2173–2200, 2001.

[82] D. Weitz. Counting independent sets up to the tree threshold. In *ACM Symposium on Theory of computing*, 2006.

[83] A. S. Willsky. Multiresolution Markov models for signal and image processing. *Proceedings of the IEEE*, 90:1396–1458, August 2002.

[84] J. Woods. Markov Image Modeling. *IEEE Transactions on Automatic Control*, 23:846–850, October 1978.

[85] S. J. Wright. *Primal-dual Interior-point methods*. Society of Industrial and Applied Mathematics, 1997.