Embedded Trees: Estimation of Gaussian Processes on Graphs with Cycles

by

Erik B. Sudderth

B.S., Electrical Engineering
University of California at San Diego, 1999

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

February 2002

Author .....
Department of Electrical Engineering and Computer Science
February 4, 2002

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

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

Embedded Trees: Estimation of Gaussian Processes on Graphs with Cycles

by

Erik B. Sudderth

Submitted to the Department of Electrical Engineering and Computer Science
on February 4, 2002, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Structured statistical models play a critical role in controlling the complexity of large-scale estimation problems. Graphical models provide a powerful general framework for encoding this structure. For graphs which are acyclic or tree-structured, there exist efficient exact algorithms for statistical inference. However, real-world phenomena are often only well modeled by graphs with cycles. As direct estimation procedures are generally prohibitively costly for such graphs, it is necessary to develop more efficient iterative alternatives.

In this thesis, we develop a novel iterative estimation algorithm for Gaussian processes defined on graphs with cycles. It operates by exactly solving a series of modified estimation problems on spanning trees embedded within the original cyclic graph. When the algorithm converges, it always computes the correct conditional means. In contrast to many other iterative estimation algorithms, the tree-based procedures we propose may be extended to calculate exact error variances. Although the conditional mean iteration is effective for quite densely connected graphical models, the error variance computation is most efficient for sparser graphs. In this context, we present a modeling example which suggests that very sparsely connected graphs with cycles may provide significant advantages relative to their tree-structured counterparts.

The convergence properties of the proposed tree-based iterations are extensively characterized both analytically and experimentally. We also provide an analysis of the geometric structure underlying these iterations which naturally suggests techniques for accelerating their convergence. Several different acceleration methods are proposed and analyzed, the most effective of which uses the tree-based iterations to precondition the conjugate gradient method. Simulation results are presented showing that for many problems, accelerated tree-based iterations converge much more rapidly than competing techniques.

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

Acknowledgments

This thesis would not have been possible without the encouragement, insight, and guidance of my advisor, Professor Alan Willsky. In my first semester at MIT, he suggested the problems which would eventually be addressed by this thesis, and patiently answered my many questions as I learned about the intracacies of graphical models. As my research progressed, he provided a seemingly limitless supply of new ideas, suggesting connections and interpretations which I would never have made on my own. His thorough revisions of the chapters of this thesis, and willingness to adjust his schedule in order to make these revisions possible, are greatly appreciated.

As the ideas forming this thesis developed, I benefited from innumerable conversations with fellow graduate students. In particular, Martin Wainwright has significantly contributed to the research presented here. We worked together to develop the algorithms which form the core of this thesis, and his initial theoretical results provided a basis for many of my later extensions. He has always been happy to discuss my latest idea or obstacle, and has indelibly shaped my understanding of both statistical inference and Canadians.

Special thanks are also due to Michael Schneider for introducing me to the wide world of numerical linear algebra. Many of the results in this thesis would not have been possible without the references and explanations he provided. In addition, I would like to thank Dewey Tucker and Jason Johnson for patiently listening to me in our weekly grouplet meetings. Each has provided valuable suggestions which have helped clarify my understanding of multiscale and graphical models.

My graduate studies would have been much less enjoyable without the dynamic, supportive work environment provided by the Stochastic Systems Group. I'd especially like to thank my officemates, Alex Ihler and Junmo Kim, for countless interesting conversations. In addition, Taylore Kelly has played an invaluable role in protecting me from the terrors of the MIT bureaucracy. I am grateful that she's just crazy enough to understand me.

The successes I have had in my many years as a student are in large part due

to the love and encouragement of my family. I cannot thank my parents enough for giving me the opportunity to freely pursue my educational goals. This thesis is one more step in a path made possible by their unwavering support.

Finally, I would like to thank my wife Erika for ensuring that the last two years of my life were not entirely consumed by graduate studies. She has an amazing ability to make even the most discouraging problems seem manageable. Her love and understanding have given me the strength to complete this thesis.

Contents

Abstract **3**

Acknowledgments **5**

List of Figures **11**

1 Introduction **13**

1.1 Multiscale Models and Graphs with Cycles 14

1.2 Contributions and Organization 19

2 Background **23**

2.1 Linear Estimation 23

2.2 Graphical Models 25

2.2.1 Graph Separation and Conditional Independence 26

2.2.2 Structured Inverse Covariance Matrices 28

2.2.3 Parameterization of Gaussian Markov Random Fields 30

2.3 Graph–based Inference Algorithms 33

2.3.1 Exact Inference for Tree–Structured Graphs 34

2.3.2 Gaussian Belief Propagation 39

2.3.3 Inference for Graphs with Cycles 43

2.4 Iterative Solution of Linear Systems 45

2.4.1 Stationary Richardson Methods 46

2.4.2 The Conjugate Gradient Iteration 48

2.4.3 Stopping Criteria 50

3 Embedded Trees **51**

3.1 Loopy Graphs and Spanning Trees 52

3.2 Iterative Calculation of Conditional Means 55

3.2.1 Exploiting Embedded Trees 55

3.2.2 Embedded Trees as a Richardson Iteration 58

3.2.3 Motivating Examples 60

3.2.4 Geometric Analysis 65

3.2.5 Connections to Previous Work 68

3.3 Iterative Calculation of Error Covariances 71

3.3.1 Low–Rank Decompositions of Cutting Matrices 73

3.3.2 Embedded Trees as a Series Expansion 77

3.3.3 A Direct Tracking Approach to Error Covariances 79

3.3.4 A Fixed Point Approach to Error Covariances 81

3.4 Further Convergence Analysis 83

3.4.1 A Single Spanning Tree 84

3.4.2 Multiple Spanning Trees 86

3.5 Numerical Examples 88

3.5.1 Simulation Conditions 88

3.5.2 Augmented Multiscale Trees 92

3.5.3 Nearest–Neighbor Grids 98

3.5.4 Observations 101

4 Accelerated Tree–Based Iterations **105**

4.1 Diagonal Acceleration 106

4.1.1 Numerical Examples 107

4.1.2 Discussion 109

4.2 Rank–One Corrections 112

4.2.1 Geometry and Inference on Single Cycle Graphs 112

4.2.2 An Automatic Correction Procedure 114

4.2.3 Numerical Examples 118

4.2.4 Discussion 120
4.3 Spanning Tree Preconditioners 122
4.3.1 Single Tree Preconditioners 122
4.3.2 Multiple Tree Preconditioners 125
4.3.3 Discussion 129

5 Recommendations and Conclusions **131**
5.1 Contributions 131
5.1.1 Embedded Trees Estimation Algorithm 132
5.1.2 Embedded Trees Error Covariance Algorithm 132
5.1.3 Accelerated Tree–Based Iterations 133
5.2 Extensions 134
5.2.1 Inference Algorithms for Graphs with Cycles 134
5.2.2 Multiscale Modeling using Graphs with Cycles 136

A Linear Algebraic Identities **139**
A.1 Inversion of Partitioned Matrices 139
A.2 Eigenvalue Identities 140

B Gaussian Random Vectors **143**
B.1 Information Parameters 143
B.2 Canonical Correlations 145
B.2.1 Calculation of Canonical Correlation Coefficients 146
B.2.2 Interpretation of Canonical Correlation Coefficients 147
B.2.3 Proof of Proposition 2.1 150

C Proofs for Chapter 3 **153**
C.1 Proof of Theorem 3.2 153
C.2 Proof of Lemma 3.4 154
C.3 Proof of Theorem 3.11 155
C.4 Proof of Theorem 3.15 156
C.5 Proof of Theorem 3.16 157

Bibliography **159**

List of Figures

1-1 Sample graphical model structures. Each node represents a random variable. 15

1-2 Multiscale modeling example. Note that adding an extra edge significantly reduces boundary artifacts. 18

2-1 (a) An undirected graphical model representing five random variables. (b) Correspondence between graph separation and conditional independence. 27

2-2 (a) Graphical model representing five jointly Gaussian random vectors. (b) Structure of the corresponding inverse covariance matrix P^{-1}. . . 31

2-3 Graphical model with observation nodes explicitly shown. 34

2-4 Subsets of the observation nodes used by the BP algorithm. 35

3-1 Embedded trees produced by five different cutting matrices $\{K_{\mathcal{T}_i}\}_{i=1}^{5}$ for a nearest–neighbor grid. 53

3-2 Behavior of the ET algorithm on a single 20–node cycle. 63

3-3 Convergence of the ET algorithm on a 25×25 nearest–neighbor grid. 65

3-4 Spanning forests employed by the ADI method. 70

3-5 An undirected graphical model with inverse error covariance $\widehat{J}$, and the corresponding spanning tree produced by the cutting matrix $K_{\mathcal{T}}$. 75

3-6 Augmented multiscale model with three extra fine scale edges, and the two spanning trees used to implement the ET algorithm. 93

3-7 Convergence rates for an augmented multiscale model with homogeneous potentials. 94

3-8 Convergence rates for an augmented multiscale model with disordered potentials. 95
3-9 Convergence rates for the augmented multiscale model of §1.1. 96
3-10 Error covariance methods applied to augmented multiscale models. . 96
3-11 25×25 nearest–neighbor grid, and the two spanning trees used to implement the ET algorithm. 98
3-12 Convergence rates for a 25×25 nearest–neighbor grid with homogeneous potentials. 99
3-13 Convergence rates for a 25×25 nearest–neighbor grid with disordered potentials. 100
3-14 Error covariance methods applied to a 25×25 nearest–neighbor grid. 102

4-1 Diagonal acceleration of the ET algorithm for the augmented multiscale model of Figure 3-6. 108
4-2 Diagonal acceleration of the ET algorithm for a 25×25 nearest–neighbor grid. 110
4-3 Rank one correction of the ET algorithm applied to the augmented multiscale model of Figure 3-6. 118
4-4 Rank one correction of the ET algorithm applied to a 25×25 nearest–neighbor grid. 120
4-5 Single–tree preconditioning of the CG algorithm applied to the augmented multiscale model of Figure 3-6. 125
4-6 Single–tree preconditioning of the CG algorithm applied to a 25×25 nearest–neighbor grid. 126
4-7 Multiple–tree preconditioning applied to a 25×25 nearest–neighbor grid. 128

Chapter 1

Introduction

Gaussian processes play an important role in a wide range of practical, large–scale statistical estimation problems. Moderately sized Gaussian inference problems are easily solved using standard linear algebraic techniques. However, for applications arising in such fields as image processing and oceanography, Gaussian priors are commonly used to model the statistical dependencies among tens or hundreds of thousands of random variables. In such cases, the storage and computational requirements of direct, brute–force modeling and inference techniques are intractably large. Instead, families of structured statistical models, and complementary classes of efficient estimation algorithms, must be developed.

In this thesis, we study Gaussian processes whose statistical properties are constrained by an associated graph. The nodes of the graph represent random variables, while edges specify their statistical interrelationships (see Figure 1-1). A wide range of stochastic processes can be compactly expressed by such graphical models [43, 48]. For example, linear state space models [44], Markov random fields [13], and multiscale autoregressive models [20, 30] are all defined using graphical constraints. When designing graphical models, graph structure naturally captures a fundamental tradeoff between the expressiveness and accuracy of the modeled distribution, and the complexity of statistical inference. At one extreme are tree–structured graphs: although they lead to highly efficient estimation algorithms, their modeling power is rather limited. The addition of edges to the graph tends to increase modeling power, but

also introduces cycles that require the use of more sophisticated, and computationally expensive, inference techniques.

In an effort to mitigate this tradeoff, this thesis focuses on the development of efficient inference algorithms for arbitrarily structured Gaussian graphical models. The proposed algorithms are iterative, computing a sequence of estimates which converges asymptotically to the desired solution. Importantly, given a set of noisy observations, the proposed methods calculate not only the conditional means, but also the associated error variances. These error variances provide important information about the reliability of the computed estimates, but are not correctly calculated by many existing inference algorithms.

Many aspects of the inference algorithms developed in this thesis were strongly motivated by the modeling tradeoffs hinted at above. Thus, we begin with an example, originally presented in [78], which explores these issues in more detail.

1.1 Multiscale Models and Graphs with Cycles

Graphical models, like those shown in Figure 1-1, use edges to represent the statistical structure of the modeled random variables. Although this thesis focuses on Gaussian models, graphical models may also be defined using non–Gaussian random variables [48]. Associated with each edge is a function which provides a local measure of the statistical dependence between the variables it joins. Intuitively, the correlation between any given pair of random variables is determined by the combined effects of all sequences of edges, or paths, connecting them. Typically, variables are most strongly dependent on their nearest neighbors, as correlation tends to decrease with path length. Any probability distribution can be represented by a sufficiently densely connected graphical model. However, the structure provided by a graphical model is only useful when it is relatively sparsely connected, so that each node depends directly on only a few neighbors.

In many application areas, no single statistical model, graphical or otherwise, can be uniquely identified as the "correct" model for the variables of interest. In such

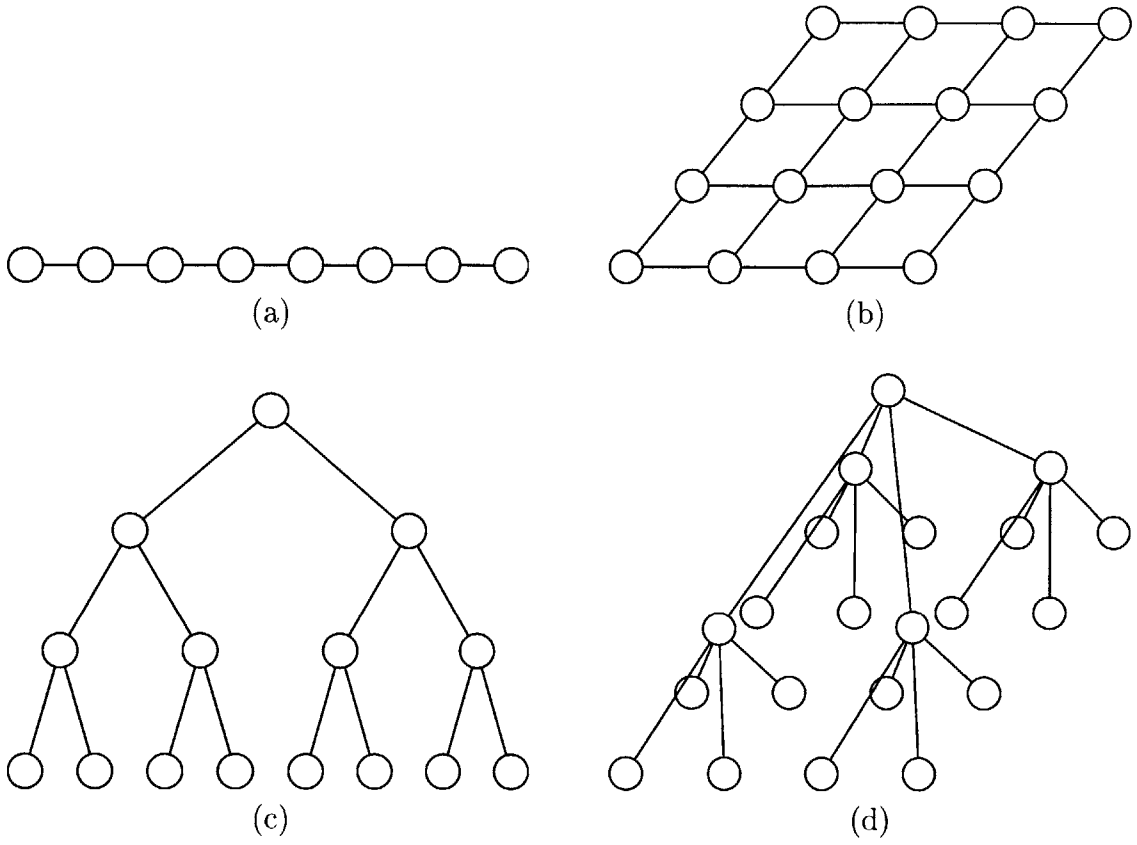


Figure 1-1: Sample graphical model structures. Each node represents a random variable. (a) Markov chain (state space model). (b) Markov random field (2-D nearest neighbor grid). (c) Multiscale autoregressive time series model. (d) Multiscale autoregressive random field model.

cases, many different graphical structures may provide reasonable approximations of the desired dependencies. The challenge, then, is to find graphs which accurately capture important statistical features, but still lead to efficient inference algorithms. For the modeling of one–dimensional time series, linear state space models, which graphically correspond to the Markov chain of Figure 1-1(a), are a common choice. Computationally, they are attractive because efficient and exact inference is possible using the Kalman filter [44]. In addition, their representation of each successive random variable as a noisy function of its immediate predecessors is a good match for many real stochastic processes.

Markov random fields defined on 2-D nearest neighbor grids, as shown in Figure 1-1(b), are a natural generalization of Markov chains for the modeling of spatial processes. The statistical structure they provide, in which each random variable

interacts directly with its nearest spatial neighbors, can lead to a wide range of interesting two–dimensional distributions. Unfortunately, however, the presence of large numbers of cycles in this graph leads to estimation algorithms that are very computationally intensive. In particular, accurate calculation of the error variances associated with a Markov random field model is often intractable. Thus, in many situations, it is necessary to replace Markov random fields with more computationally feasible alternatives.

Multiscale autoregressive models, as illustrated in Figure 1-1(c,d), provide an attractive alternative to traditional Markov models. In such models, additional "coarse scale" nodes are added to the graph which may or may not be directly linked to any measurements.[1] These nodes are auxiliary variables created to explain the "fine scale" stochastic process of primary interest. If properly designed, the resulting tree structure can accurately model a wide range of interesting phenomena. For example, large scale, long–range interactions are quite naturally captured by the presence of variables at multiple scales. In addition, generalizations of the Kalman filter allow optimal estimates, including error variances, to be calculated extremely efficiently [20].

Previous research has demonstrated that multiscale graphical models can provide efficient, effective solutions to a wide range of problems [24, 28, 29, 50, 66]. The most significant weakness revealed by these studies is the presence of boundary artifacts in the computed estimates. The problem is that spatially adjacent, and hence highly correlated, fine–scale nodes may be widely separated in the tree structure. As a result, dependencies between these nodes may be inadequately modeled, causing blocky discontinuities. Blockiness can be reduced by increasing the dimension of the coarse scale nodes. However, the computational cost of inference algorithms is cubic in the variable dimension, so it is usually not computationally feasible to completely eliminate blockiness in this fashion.

One potential solution to the boundary artifact problem is to add edges between

[1] In some applications, coarse scale nodes provide a convenient way for modeling measurements which naturally occur at multiple scales [23, 24]. However, in the example presented here, we are only interested in the covariance matrix induced at the finest scale.

pairs of fine–scale nodes where discontinuities are likely to arise. Such edges should be able to account for short–range dependencies neglected by standard multiscale models. To illustrate this idea, we consider the modeling of the one–dimensional process of length 32 whose exact covariance P is shown in Figure 1-2(a). We approximate this process using two different graphical models: a multiscale tree (Figure 1-2(b)), and a "near–tree" containing an additional edge across the largest fine scale tree boundary (Figure 1-2(c)). In both models, the dimension of the Gaussian variable at each node is constrained to be 2; therefore, the finest scale contains 16 nodes to model all 32 process points.

The tree–based multiscale model was realized using the scale–recursive algorithm presented in [30, 31]. Figure 1-2(d) shows the resulting fine–scale covariance matrix P_{tree}, while Figure 1-2(f) gives the corresponding absolute error $|P - P_{\text{tree}}|$. The tree model matches the desired process statistics relatively well except at the center, where the tree structure causes a boundary artifact. In contrast, Figures 1-2(e) and 1-2(g) show the covariance P_{loop} and absolute error $|P - P_{\text{loop}}|$ which can be attained by the augmented multiscale model. The addition of a single edge has reduced the peak error by 60%, a substantial gain in modeling fidelity. In §3.5.2, we show that the inference algorithms developed in this thesis can efficiently and accurately calculate both conditional means and error variances for this model.

The previous example suggests that very sparsely connected graphs with cycles may offer significant modeling advantages relative to their tree–structured counterparts. Unfortunately, as the resulting graphs do have cycles, the extremely efficient inference algorithms which made tree–structured multiscale models so attractive are not available. The primary goal of this thesis is to develop novel inference techniques which allow estimates, and the associated error variances, to be quickly calculated for the widest possible class of graphs. The algorithms we develop are particularly effective for graphs, like that presented in this example, which are nearly tree–structured. We hope that our results will motivate future studies exploring the design of multiscale models defined on graphs with cycles.

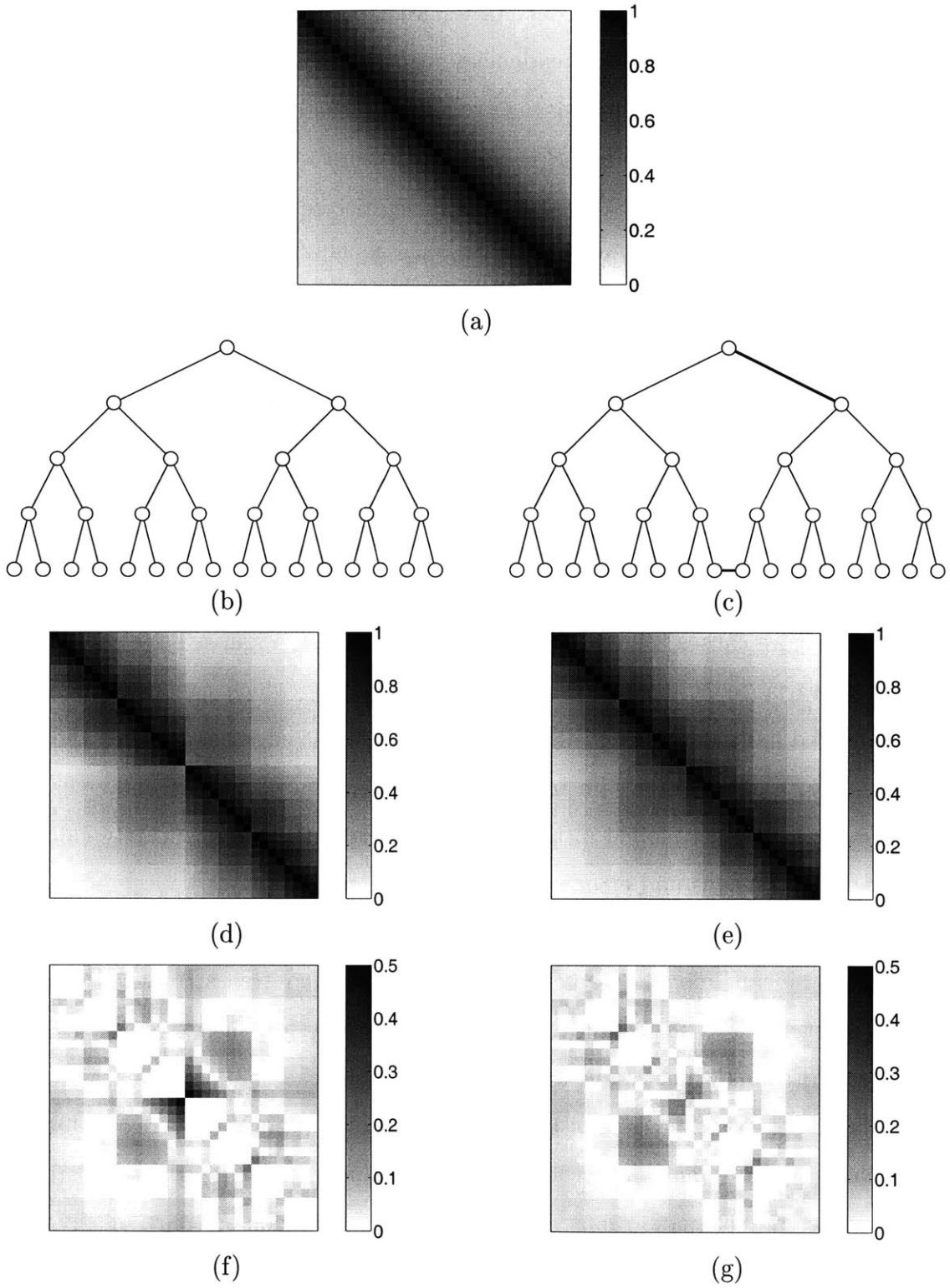


Figure 1-2: Multiscale modeling example. Note that adding an extra edge significantly reduces boundary artifacts. (a) Desired covariance P. (b) Multiscale tree model. (c) Tree augmented by extra edge. The two weakest edges in this graph's single cycle are highlighted (see §3.5.2). (d) Covariance P_{tree} realized by tree model. (e) Covariance P_{loop} realized by graph with cycles. (f) Error $|P - P_{\text{tree}}|$ in tree realization. (g) Error $|P - P_{\text{loop}}|$ in loopy realization.

1.2 Contributions and Organization

For Gaussian graphical models whose graphs are tree–structured, there exist efficient, exact inference algorithms for finding both estimates and error variances. The primary contribution of this thesis is to demonstrate that tree–based inference routines provide a natural basis for the design of estimation algorithms which apply to much broader classes of graphs. All of the algorithms depend on the fact that, within any graph with cycles, there are many embedded tree–structured subgraphs. Each of these embedded trees can be revealed by removing a different subset of the original graph's edges. We show that, by appropriately combining sequences of exact calculations on these embedded trees, it is possible to solve statistical inference problems defined on the original graph with cycles efficiently. Importantly, embedded trees lead to procedures for calculating not only conditional means, but also the associated error variances.

The remainder of this thesis is organized as follows.

Chapter 2, Background

The primary purpose of Chapter 2 is to provide a self–contained introduction to graphical models. After a brief review of linear estimation theory, we demonstrate how graphs are used to define stochastic processes, focusing on Gaussian models. We then discuss the strengths and weaknesses of existing graph–based inference algorithms. In the process, we provide a detailed derivation of the popular "belief propagation" algorithm for exact inference on tree–structured graphs. The derivation is from a probabilistic perspective, and reveals several interesting features not discussed in the standard belief propagation literature. The chapter concludes with a discussion of iterative methods from the numerical linear algebra literature, including the classical Richardson iteration and the conjugate gradient algorithm. Some of the novel algorithms presented in the following chapters are derived as extensions to these more basic methods.

Chapter 3, Embedded Trees

Chapter 3 presents a family of tree–based inference algorithms for solving estimation problems defined on graphs with cycles. The chapter begins with a discussion of the connections between spanning tree distributions and graphs with cycles. Then, we use these connections to develop the embedded trees (ET) algorithm, which computes a sequence of estimates converging to the conditional mean of the loopy estimation problem. A combination of theoretical results and numerical examples are used to demonstrate several properties of the ET iteration, including the important performance improvements obtained by employing multiple embedded trees.

Following the discussion of the ET iteration for estimates, we present a complementary tree–based iterative algorithm for calculating error variances. The computational complexity of this algorithm is proportional to the number of edges which must be removed from the graph with cycles to reveal an embedded tree. Thus, it will be particularly efficient for the very sparse loopy graphs discussed in §1.1. We present several theoretical results which may be used to guarantee the convergence of either inference algorithm. The chapter concludes with a set of simulations demonstrating the effectiveness of the proposed methods.

Chapter 4, Accelerated Tree–Based Iterations

In Chapter 4, we present several techniques which use the basic embedded trees iteration developed in Chapter 3 to create more rapidly convergent algorithms. We start by demonstrating that the standard ET iteration may sometimes be made to converge faster by appropriately modifying the numerical structure of the tree–structured subproblems solved at each iteration. Then, we show that the dynamics of the ET iteration reveal information about the directions in which the errors in the current estimate lie. This leads to a rank–one correction technique which first estimates the dominant error direction, and then removes those errors on subsequent iterations.

We discuss certain inefficiencies with the rank–one acceleration method, which in turn motivate the use of the ET algorithm as a preconditioner for the conjugate

gradient iteration. We demonstrate that for sparsely connected graphs, preconditioners based on a single spanning tree are guaranteed to rapidly converge to the exact solution. Several numerical simulations are then presented showing that even on more densely connected graphs, both single and multiple tree preconditioners can be very effective. The chapter concludes with a discussion of connections to existing preconditioning theory.

Chapter 5, Recommendations and Conclusions

The main contributions of the thesis are summarized in Chapter 5. Several open research problems associated with tree–based inference algorithms for graphs with cycles are proposed. In addition, a variety of open questions associated with the design of graphical models with cycles are presented.

Chapter 2

Background

In this thesis, we develop computationally efficient algorithms for solving Gaussian statistical inference problems defined by *graphical models*. The primary purpose of this chapter is to provide a self–contained introduction to graphical models, emphasizing their powerful ability to express globally consistent probability distributions through a series of local constraints. After a brief review of linear estimation in §2.1, we demonstrate in §2.2 how graphs are associated with probability distributions, focusing on the particular features of Gaussian models. Then, in §2.3, we discuss the strengths and weaknesses of existing inference algorithms for graphical models. In particular, we provide a detailed derivation of a popular inference algorithm known as *belief propagation* which forms the basis for many of the new results in this thesis. The chapter concludes in §2.4 with an introduction to some relevant techniques from the numerical linear algebra literature.

2.1 Linear Estimation

This thesis focuses on a classical problem in estimation theory. We are given a vector of observations y of an unknown random vector x, where x and y are jointly Gaussian with zero mean.[1] Under these assumptions, the conditional distribution

[1]Non–zero means can be easily accounted for by estimating the deviation from the mean using the modified random variables $\bar{x} = (x - E[x])$ and $\bar{y} = (y - E[y])$.

$p(x \mid y) \sim \mathcal{N}\left(\widehat{x}, \widehat{P}\right)$ is also Gaussian, with mean and covariance given by the *normal equations*:

$$\widehat{x} \triangleq E[x \mid y] = \mathcal{P}_{xy}\mathcal{P}_y^{-1}y \tag{2.1}$$

$$\widehat{P} \triangleq E\left[(x-\widehat{x})(x-\widehat{x})^T \middle| y\right] = \mathcal{P}_x - \mathcal{P}_{xy}\mathcal{P}_y^{-1}\mathcal{P}_{xy}^T \tag{2.2}$$

Here, $\mathcal{P}_x \triangleq E\left[xx^T\right]$ and $\mathcal{P}_y \triangleq E\left[yy^T\right]$ denote the covariance matrices of x and y, respectively, and $\mathcal{P}_{xy} \triangleq E\left[xy^T\right]$ denotes their cross–covariance matrix.

For Gaussian problems, the conditional mean $\widehat{x}$ is the best estimate of the unknown vector x under a wide range of error criteria. In particular, it coincides with both the *Bayes' least squares* estimate minimizing the mean squared error $E\left[(x-\widehat{x})^2 \mid y\right]$ and the *maximum a posteriori* (MAP) estimate maximizing $p(\widehat{x} \mid y)$. Even if x and y are not jointly Gaussian, $\widehat{x}$ may still be justified as giving the smallest mean squared error of any linear estimator [45]. For this reason, $\widehat{x}$ is sometimes called the *linear least squares* estimate (LLSE).

Although the estimate $\widehat{x}$ is the primary objective of inference, the covariance $\widehat{P}$ of the conditional distribution also provides important information. From equation (2.2), we see that $\widehat{P}$ is equal to the covariance of $(x-\widehat{x})$, the error between the optimal estimate $\widehat{x}$ and the true value of the unobserved variables x. Therefore, it provides a measure of the reliability of the estimate $\widehat{x}$, and is often referred to as the *error covariance* matrix. Note that for Gaussian problems, $\widehat{P}$ is not a function of the observed vector y, but does depend on the joint statistics of x and y.

In many problem domains, the observations y are naturally expressed as a noise–corrupted linear function of x given by

$$y = Cx + v \tag{2.3}$$

where $v \sim \mathcal{N}(0, R)$ is a zero–mean Gaussian noise process, and x and v are indepen-

dent. In this case, equations (2.1, 2.2) specialize to

$$\widehat{x} = PC^T \left(CPC^T + R\right)^{-1} y \tag{2.4}$$

$$\widehat{P} = P - PC^T \left(CPC^T + R\right)^{-1} CP \tag{2.5}$$

where $P \triangleq E\left[xx^T\right]$ is the prior covariance of the unobserved variables x. Assuming that P and R are both positive definite and hence invertible, we may use the matrix inversion lemma (see Appendix A.1) to rewrite equations (2.4, 2.5) as

$$\left(P^{-1} + C^T R^{-1} C\right) \widehat{x} = C^T R^{-1} y \tag{2.6}$$

$$\widehat{P} = \left(P^{-1} + C^T R^{-1} C\right)^{-1} \tag{2.7}$$

This information form of the normal equations plays a crucial role in computations involving the structured prior models considered later in this thesis.

Although the normal equations give an explicit formula for determining both $\widehat{x}$ and $\widehat{P}$, directly solving them may require an intractable amount of computation for large–scale estimation problems. Let N_x be the dimension of x, and N_y be the dimension of y. If we assume that R is diagonal but all other matrices are full, then equations (2.4, 2.5) require $\mathcal{O}(N_y^3 + N_x^2 N_y)$ operations, while equations (2.6, 2.7) require $\mathcal{O}(N_x^3)$. For practical problems arising in fields such as image processing and oceanography, $N_x \approx N_y \approx 10^5$ and either formulation is intractable. Also, note that in the absence of special structure, simply storing P or $\widehat{P}$ requires $\mathcal{O}(N_x^2)$ bytes, which is often unreasonably large. These costs motivate the development of structured statistical models which yield efficient estimation procedures.

2.2 Graphical Models

Graphical models provide a powerful general framework for encoding the probabilistic structure of a set of random variables [43, 48]. They are ideally suited to applications where the statistical behavior of a large, complex system may be specified in terms

of a combination of local observations and interactions. This local structure allows complicated stochastic processes to be specified very compactly. In addition, efficient inference algorithms can exploit this structure to achieve huge savings over direct computational costs. For these reasons, graphical models have been widely applied in such fields as artificial intelligence [59], error correcting codes [34, 51], speech processing [60], statistical physics [82], image processing [36, 66], remote sensing [24, 29], and computer vision [28, 32, 50].

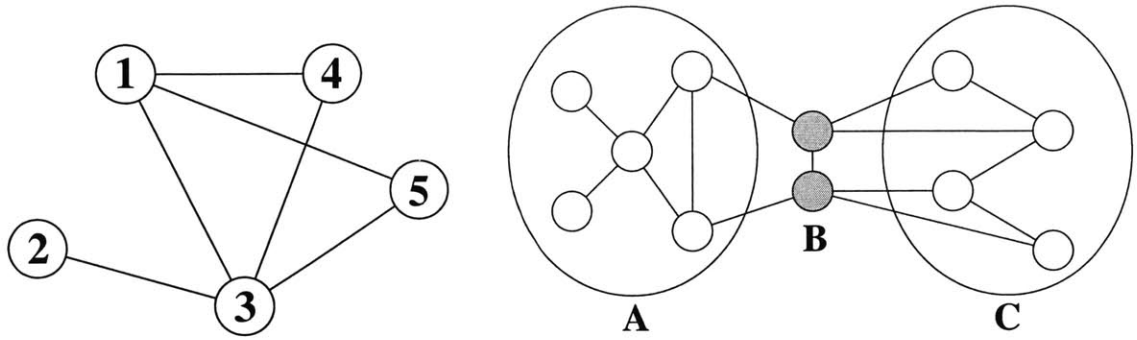
2.2.1 Graph Separation and Conditional Independence

A graph $\mathcal{G}$ consists of a set of nodes or vertices $\mathcal{V}$, and a corresponding set of edges $\mathcal{E}$. Graphical models associate each node $s \in \mathcal{V}$ with a random vector x_s. In general, x_s could be drawn from a wide variety of probability distributions. In this thesis, however, we focus on the case where x_s is a Gaussian random vector of dimension $d_s \triangleq \dim x_s$. For any subset $\mathcal{A} \subset \mathcal{V}$, we denote the set of random variables in $\mathcal{A}$ by $x_{\mathcal{A}} \triangleq \{x_s\}_{s \in \mathcal{A}}$. If the $N \triangleq |\mathcal{V}|$ nodes are indexed by the integers $\mathcal{V} = \{1, 2, \ldots, N\}$, the Gaussian stochastic process defined on the overall graph is given by $x \triangleq \begin{bmatrix} x_1^T & x_2^T & \cdots & x_N^T \end{bmatrix}^T$. Note that $\dim x = \sum_{s \in \mathcal{V}} d_s$.

Graphical models use edges to implicitly specify a set of conditional independencies. Each edge $(s, t) \in \mathcal{E}$ connects two nodes $s, t \in \mathcal{V}$, where $s \neq t$. In this thesis, we exclusively employ *undirected* graphical models for which the edges (s, t) and (t, s) are equivalent.[2] Figure 2-1(a) shows an example of an undirected graphical model representing five different random variables. Such models are also known as *Markov random fields* (MRFs), or for the special case of jointly Gaussian random variables as *covariance selection models* in the statistics literature [26, 48, 69]. For a graphical interpretation of many standard Gaussian data analysis techniques, see [61].

When describing the statistical properties of Markov random fields, the structural properties of the underlying graph play an important role. A *path* between nodes s_0

[2]There is another formalism for associating conditional independencies with graphs which uses directed edges. Any directed graphical model may be converted into an equivalent undirected model, although some structure may be lost in the process [48, 59].



(a) (b)

Figure 2-1: (a) An undirected graphical model representing five random variables. (b) Correspondence between graph separation and conditional independence. The random variables in sets $\mathcal{A}$ and $\mathcal{C}$ are conditionally independent given set $\mathcal{B}$.

and s_T is defined to be a sequence of nodes $s_0, s_1, \ldots, s_T$ such that $(s_{i-1}, s_i) \in \mathcal{E}$ for $i = 1, \ldots, T$, and $(s_{i-1}, s_i) \neq (s_{j-1}, s_j)$ for $i \neq j$. If there is a path between every pair of nodes, $\mathcal{G}$ is *connected.* A *cycle*, or loop,[3] is defined as a path which starts and ends with the same node. If a graph has no cycles, it is said to be *tree–structured.* Note that in a connected tree–structured graph, there is a unique path between each pair of nodes. The *diameter* of a graph is equal to the number of edges in the longest path between any two nodes. Finally, a *clique* is defined to be a set of nodes in which every node is *directly* connected to every other node in the clique. For example, in Figure 2-1(a) the sets $\{x_1\}$, $\{x_1, x_3\}$, $\{x_1, x_3, x_4\}$, and $\{x_1, x_3, x_5\}$ are all cliques. However, $\{x_1, x_3, x_4, x_5\}$ is not a clique because there is no edge between x_4 and x_5.

For Markov random fields defined on undirected graphs, conditional independence is associated with *graph separation.* Suppose that $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ are subsets of the vertex set $\mathcal{V}$. Then set $\mathcal{B}$ is said to separate sets $\mathcal{A}$ and $\mathcal{C}$ if there are no paths between sets $\mathcal{A}$ and $\mathcal{C}$ which do not pass through set $\mathcal{B}$. The stochastic process x is said to be Markov with respect to $\mathcal{G}$ if $x_\mathcal{A}$ and $x_\mathcal{C}$ are conditionally independent given $x_\mathcal{B}$ whenever $\mathcal{A}$ and $\mathcal{C}$ are separated by $\mathcal{B}$. Figure 2-1(b) illustrates this concept. For example, in

[3]In graph theoretic terminology, a loop is an edge connecting a node to itself [48]. However, as graphical models do not have self–connections, in this thesis we use the terms loop and cycle interchangeably, as is standard in the graphical inference literature [52, 77, 79].

Figure 2-1(a) we have

$$p(x_2, x_4, x_5 \mid x_1, x_3) = p(x_2 \mid x_1, x_3)\, p(x_4 \mid x_1, x_3)\, p(x_5 \mid x_1, x_3)$$

If the *neighborhood* of a node s is defined to be $N(s) \triangleq \{t | (s,t) \in \mathcal{E}\}$, the set of all nodes which are directly connected to s, it follows immediately that

$$p\left(x_s \mid x_{\mathcal{V}\setminus s}\right) = p\left(x_s \mid x_{N(s)}\right) \tag{2.8}$$

That is, conditioned on its immediate neighbors, the probability distribution of the random vector at any given node is independent of the rest of the process. In Figure 2-1(a), this property implies that

$$p(x_5 \mid x_1, x_2, x_3, x_4) = p(x_5 \mid x_1, x_3)$$

We may also define a local neighborhood for any $\mathcal{A} \subset \mathcal{V}$ as $N(\mathcal{A}) \triangleq \left\{\bigcup_{s\in\mathcal{A}} N(s) \setminus \mathcal{A}\right\}$, which gives the following generalization of equation (2.8):

$$p\left(x_{\mathcal{A}} \mid x_{\mathcal{V}\setminus s}\right) = p\left(x_{\mathcal{A}} \mid x_{N(\mathcal{A})}\right) \tag{2.9}$$

The nearest–neighbor conditional independencies implied by equation (2.8) are known as *local* Markov relationships, in contrast to the *global* Markov relationships illustrated in Figure 2-1(b). It is possible to construct degenerate probability distributions which satisfy (2.8) for all $s \in \mathcal{V}$, but do not globally associate graph separation with conditional independence. However, for graphical models with positive continuous densities, like the Gaussian Markov random fields considered in this thesis, the global and local Markov properties are equivalent [48].

2.2.2 Structured Inverse Covariance Matrices

Clearly, only a limited subset of Gaussian probability distributions $p(x) \sim \mathcal{N}(0, P)$ will satisfy the conditional independencies implied by a given graph $\mathcal{G}$. Interest-

ingly, in the Gaussian case, the Markov properties of $\mathcal{G}$ constrain the structure of the *inverse* covariance matrix P^{-1}. Most efficient inference algorithms for Gaussian graphical models, including those developed in this thesis, depend fundamentally on the exploitation of this structure.

Suppose the inverse covariance matrix $J \triangleq P^{-1}$ is partitioned into an $N \times N$ grid of submatrices, where the submatrix sizes are chosen to match the dimensions of the node variables $\{x_s\}_{s=1}^N$. Denote the $(s,t)^{th}$ block of J by $J_{s,t}$, so that $J_{s,t}$ is a matrix of size $d_s \times d_t$. Also, let $\{\lambda_i(A)\}$ and $\{\sigma_i(A)\}$ denote the sets of eigenvalues and singular values, respectively, of a matrix A. The following proposition, which is proved for the special case of scalar variables in [48], provides a statistical interpretation of the inverse covariance entries:

Proposition 2.1. For any $s \in \mathcal{V}$, the *conditional* covariance matrix $\mathrm{var}\left(x_s \middle| x_{N(s)}\right)$ may be directly calculated from the corresponding block diagonal entry of $J = P^{-1}$:

$$\mathrm{var}\left(x_s \middle| x_{N(s)}\right) = (J_{s,s})^{-1} \tag{2.10}$$

In addition, for any $s,t \in \mathcal{V}$, the *conditional canonical correlation coefficients* of x_s and x_t, conditioned on their local neighborhood $x_{N(s,t)}$, may be calculated from

$$\left\{\sigma_i\left(\mathrm{var}\left(x_s \middle| x_{N(s,t)}\right)^{-\frac{1}{2}} \mathrm{cov}\left(x_s, x_t \middle| x_{N(s,t)}\right) \mathrm{var}\left(x_t \middle| x_{N(s,t)}\right)^{-\frac{1}{2}}\right)\right\}$$
$$= \left\{\sigma_i\left((J_{s,s})^{-\frac{1}{2}} J_{s,t} (J_{t,t})^{-\frac{1}{2}}\right)\right\} \tag{2.11}$$

Proof. See Appendix B.2.3. □

For an introduction to canonical correlations analysis, see Appendix B.2. For scalar x_s and x_t, equation (2.11) may be specialized to show that the conditional correlation coefficient $\rho_{st|N(s,t)}$ is given by

$$\rho_{st|N(s,t)} \triangleq \frac{\mathrm{cov}\left(x_s, x_t \middle| x_{N(s,t)}\right)}{\sqrt{\mathrm{var}\left(x_s \middle| x_{N(s,t)}\right) \mathrm{var}\left(x_t \middle| x_{N(s,t)}\right)}} = \frac{-J_{s,t}}{\sqrt{J_{s,s} J_{t,t}}} \tag{2.12}$$

Combining Proposition 2.1 with the Markov properties implied by equations (2.8, 2.9), we may show that the inverse covariance matrix has the following structure [48, 69]:

Theorem 2.2. Let $x \sim \mathcal{N}(0, P)$ be a Gaussian stochastic process which is Markov with respect to an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Assume that x is *not* Markov with respect to any $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$ such that $\mathcal{E}' \subsetneq \mathcal{E}$, and partition $J = P^{-1}$ into a $|\mathcal{V}| \times |\mathcal{V}|$ grid according to the dimensions of the node variables. Then for any $s, t \in \mathcal{V}$ such that $s \neq t$, $J_{s,t} = J_{t,s}^T$ will be nonzero if and only if $(s, t) \in \mathcal{E}$.

Proof. Consider any $s, t \in \mathcal{V}$ such that $s \neq t$. Combining equation (2.11) with basic properties of the singular value decomposition, we see that $J_{s,t}$ will be nonzero if and only if $\operatorname{cov}\left(x_s, x_t \middle| x_{N(s,t)}\right) \neq \mathbf{0}$. Since x_s and x_t are jointly Gaussian, it follows that $J_{s,t}$ will be zero if and only if x_s and x_t are conditionally independent given $x_{N(s,t)}$.

Suppose that $(s, t) \notin \mathcal{E}$, then $N(s, t)$ separates nodes s and t, and by the Markov properties of $\mathcal{G}$, x_s and x_t must be conditionally independent. This in turn implies $J_{s,t} = \mathbf{0}$. Alternatively, if $(s, t) \in \mathcal{E}$, then $N(s, t)$ does not separate s and t. By assumption, x is not Markov with respect to the subgraph created by removing edge (s, t), so x_s and x_t must be conditionally dependent, and therefore $J_{s,t} \neq \mathbf{0}$. □

Figure 2-2 illustrates Theorem 2.2 for a small sample graph. In most graphical models, each node is only connected to a small subset of the other nodes. Theorem 2.2 then shows that P^{-1} will be a *sparse* matrix with a small (relative to N) number of nonzero entries in each row and column. This sparsity is the fundamental reason for the existence of the efficient inference algorithms discussed in §2.3.

2.2.3 Parameterization of Gaussian Markov Random Fields

The sparse structure exhibited by the inverse covariance matrix of a Gaussian Markov random field is a manifestation of the constraints which the Markov properties discussed in §2.2.1 place on $p(x)$. Similar constraints also hold for more general undirected graphical models. In particular, the Hammersley–Clifford Theorem relates the Markov properties implied by $\mathcal{G}$ to a factorization of the probability distribution $p(x)$.

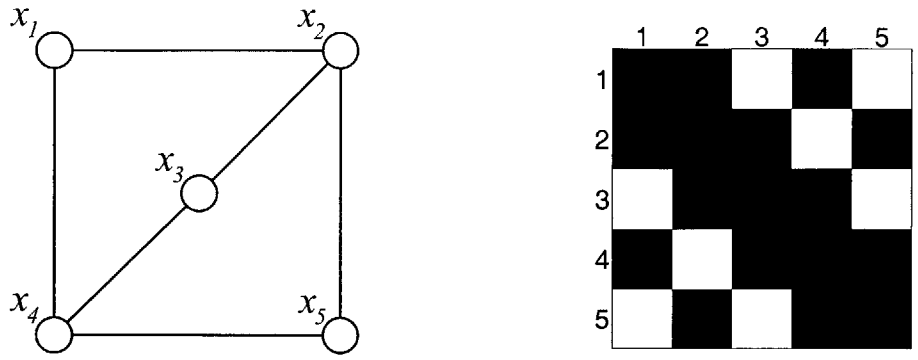


Figure 2-2: (a) Graphical model representing five jointly Gaussian random vectors. (b) Structure of the corresponding inverse covariance matrix P^{-1}, where black squares denote nonzero entries.

If $\mathcal{C}$ is the set of all cliques of $\mathcal{G}$, then a strictly positive distribution $p(x)$ is Markov with respect to $\mathcal{G}$ if and only if it can be written in the factorized form

$$p(x) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(x_C) \tag{2.13}$$

where $\psi_C(x_C)$ is an arbitrary positive function, or clique potential, defined over the elements of the clique C. Z is a normalization constant, sometimes called the partition function. For example, applying the Hammersley–Clifford Theorem to the graph in Figure 2-1(a), we find that the joint distribution $p(x)$ must factorize as

$$p(x_1, x_2, x_3, x_4, x_5) = \frac{1}{Z} \psi_{2,3}(x_2, x_3) \psi_{1,3,4}(x_1, x_3, x_4) \psi_{1,3,5}(x_1, x_3, x_5)$$

There are a variety of proofs of the Hammersley–Clifford Theorem [13, 18, 48]; see Clifford [21] for an interesting historical survey.

In many cases, the prior model is most naturally specified by "pairwise" clique potentials involving pairs of nodes which are connected by edges:

$$p(x) = \frac{1}{Z} \prod_{(s,t) \in \mathcal{E}} \psi_{s,t}(x_s, x_t) \tag{2.14}$$

An arbitrary Markov random field may be represented using only pairwise clique potentials by appropriately clustering nodes in the original graph [84]. However,

some of the graph's structure may be lost in the process.

For Gaussian Markov random fields, the prior distribution $p(x)$ is uniquely specified by either the full covariance matrix P or the inverse covariance matrix $J = P^{-1}$. However, because of the sparse structure implied by Theorem 2.2, J provides the more natural and efficient parameterization. Often, it is convenient to decompose J into (pairwise) clique potentials as in equation (2.14). To each edge $(s,t) \in \mathcal{E}$, we associate a clique potential

$$\psi_{s,t}(x_s, x_t) = \exp\left\{-\frac{1}{2}\begin{bmatrix} x_s^T & x_t^T \end{bmatrix}\begin{bmatrix} J_{s(t)} & J_{s,t} \\ J_{t,s} & J_{t(s)} \end{bmatrix}\begin{bmatrix} x_s \\ x_t \end{bmatrix}\right\} \tag{2.15}$$

where the $J_{s(t)}$ terms are chosen so that for all $s \in \mathcal{V}$,

$$\sum_{t \in N(s)} J_{s(t)} = J_{s,s} \tag{2.16}$$

Straightforward algebraic manipulation shows that any set of clique potentials (2.15) which satisfy the consistency condition (2.16) will define a probability distribution $p(x)$ such that

$$p(x) = \frac{1}{Z}\exp\left\{-\frac{1}{2}x^T P^{-1} x\right\} = \frac{1}{Z}\prod_{s=1}^{N}\prod_{t=1}^{N}\exp\left\{-\frac{1}{2}x_s^T J_{s,t} x_t\right\} =$$
$$\frac{1}{Z}\prod_{(s,t)\in\mathcal{E}}\exp\left\{-\frac{1}{2}\begin{bmatrix} x_s^T & x_t^T \end{bmatrix}\begin{bmatrix} J_{s(t)} & J_{s,t} \\ J_{t,s} & J_{t(s)} \end{bmatrix}\begin{bmatrix} x_s \\ x_t \end{bmatrix}\right\} = \frac{1}{Z}\prod_{(s,t)\in\mathcal{E}}\psi_{s,t}(x_s, x_t) \tag{2.17}$$

where $Z = \left((2\pi)^N \det P\right)^{1/2}$ is a normalization constant.

The preceding construction shows that it is always possible to represent a Gaussian Markov random field using pairwise clique potentials *without* having to cluster nodes. Note that this construction does not guarantee that the local potential matrices $\left[\begin{smallmatrix} J_{s(t)} & J_{s,t} \\ J_{t,s} & J_{t(s)} \end{smallmatrix}\right]$ will be positive definite. In such cases, $\psi_{s,t}(x_s, x_t)$ is not a bounded function of x, and therefore cannot be normalized so that it defines a locally consis-

tent probability distribution.[4] However, the message–passing recursions discussed in §2.3 are valid even if the potential terms are not positive definite.

In the remainder of this thesis, we primarily use inverse covariance matrices $J = P^{-1}$ to parameterize Gaussian Markov random fields. Sometimes, however, it is useful to reparameterize the model in terms of clique potential functions (2.14, 2.15). In such cases, it is important to remember that the decomposition into clique potentials is *not* unique. Also, note that we do not employ the state space formalism traditionally used in the time series literature [44]. State space models correspond to the factorization of $p(x)$ into a product of an initial distribution $p(x_0)$ and a series of one–step transition distributions $p(x_t \mid x_{t-1})$. While such factorizations naturally extend to the partial ordering offered by tree–structured graphs [20], they cannot be consistently constructed for graphs with cycles.

2.3 Graph–based Inference Algorithms

In this section, we apply the graphical model formalism developed in §2.2 to the linear estimation problem introduced in §2.1. Assume that the unobserved random vector $x \sim \mathcal{N}(0, P)$ is a Gaussian process which is Markov with respect to an undirected graph $\mathcal{G}$. Because the variables $\{x_s\}_{s=1}^N$ are unobserved, their associated nodes in $\mathcal{G}$ are called *hidden* nodes.[5] As discussed in §2.2.3, $p(x)$ is parameterized by a *sparse* inverse covariance matrix $J = P^{-1}$. The observations y are assumed to be generated according to equation (2.3).

When the prior distribution $p(x)$ is parameterized by an inverse covariance matrix J, the conditional distribution $p(x \mid y) \sim \mathcal{N}\left(\widehat{x}, \widehat{P}\right)$ is naturally expressed as the solution of the information form of the normal equations (2.6, 2.7). We assume, without loss of generality,[6] that the observation vector y decomposes into a

[4] There exist graphs with cycles for which *no* factorization will have normalizable clique potentials.

[5] In the recursive estimation literature, x_s is the *state* variable associated with node s [20, 44]. However, this terminology is not entirely appropriate for graphs with cycles because $\{x_t | t \in N(s)\}$ are not, in general, conditionally independent given x_s.

[6] Any observation involving multiple hidden nodes may be represented by a clique potential which includes those nodes. For Gaussian graphical models, such potentials simply modify certain entries

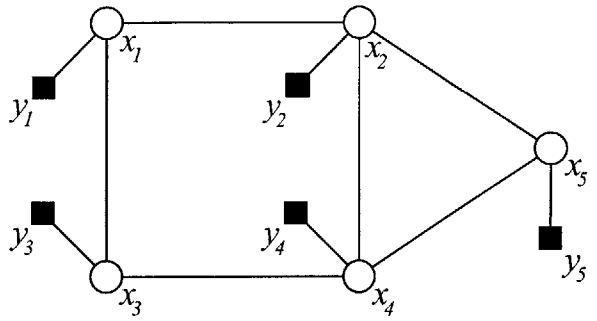


Figure 2-3: Graphical model with observation nodes explicitly shown. Circles represent hidden nodes x_s, while squares represent local observations y_s.

set $\{y_s\}_{s=1}^N$ of *local*, conditionally independent observations of the hidden variables $\{x_s\}_{s=1}^N$, so that $p(y \mid x) = \prod_{s=1}^N p(y_s \mid x_s)$. In this case, $C = \text{diag}(C_1, C_2, \ldots, C_N)$ and $R = \text{diag}(R_1, R_2, \ldots, R_N)$ must be block diagonal matrices, and for all $s \in \mathcal{V}$,

$$y_s = C_s x_s + v_s \tag{2.18}$$

where $v_s \sim \mathcal{N}(0, R_s)$. As shown in Figure 2-3, it is sometimes convenient to augment $\mathcal{G}$ by a set of *observed* nodes representing the local observations $\{y_s\}_{s=1}^N$.

Given this setup, we would like to compute the conditional *marginal* distributions $p(x_s \mid y) \sim \mathcal{N}\left(\widehat{x}_s, \widehat{P}_s\right)$ for all $s \in \mathcal{V}$. Note that each $\widehat{x}_s$ is simply a subvector of $\widehat{x}$, while each $\widehat{P}_s$ is a block diagonal element of $\widehat{P}$. We begin in §2.3.1 by deriving the general integral form of an algorithm, known as *belief propagation* (BP), which efficiently computes $p(x_s \mid y)$ for any tree–structured graph. In §2.3.2, we specialize the BP algorithm to the Gaussian case. Finally, in §2.3.3 we discuss existing exact and approximate inference algorithms for graphs with cycles.

2.3.1 Exact Inference for Tree–Structured Graphs

For graphs whose prior distribution is defined by a Markov chain, there exist efficient recursive algorithms for exactly computing the single–node conditional marginal distributions $p(x_s \mid y)$. If the variables are jointly Gaussian, one obtains a family of state–space smoothing algorithms which combine a standard Kalman filtering recur-

of P^{-1}, and can be easily accounted for by any of the inference algorithms discussed in this thesis.

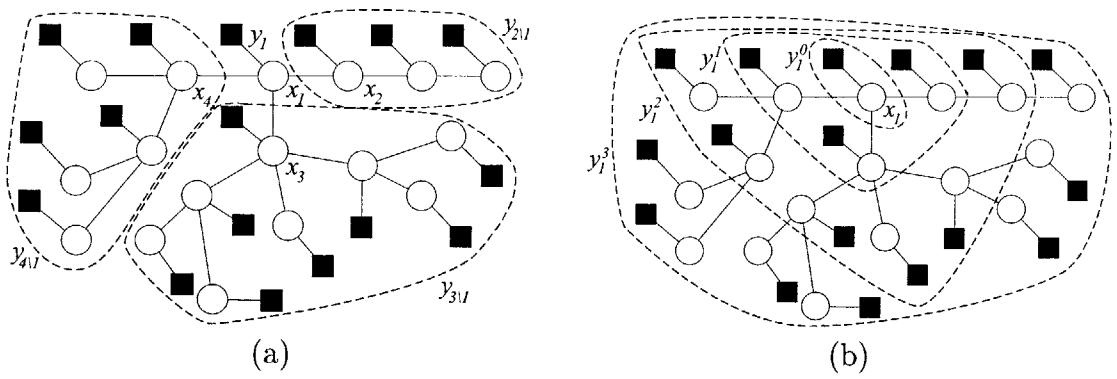


Figure 2-4: Subsets of the observation nodes used by the BP algorithm. (a) Conditioned on the hidden variable x_1, the observation sets $y_{2\setminus 1}$, $y_{3\setminus 1}$, $y_{4\setminus 1}$, and y_1 become independent. (b) At iteration n, the BP messages may be combined to compute $p(x_1 \mid y_1^n)$.

sion with a complementary reverse–time recursion [44]. Discrete–valued hidden nodes lead to the forward–backward (or $\alpha - \beta$) algorithm commonly employed in speech processing algorithms [60]. By generalizing the dynamic programming [10] recursions underlying these time–series algorithms to the partial ordering naturally provided by a tree, one can directly extend them to any graph without cycles. In this section, we derive a particular form of the exact tree–based inference recursions known as the *belief propagation* (BP) algorithm [59]. For alternate, but equivalent, frameworks for inference on graphs, see [2, 20, 47].

For any tree–structured graphical model, it is straightforward to verify that the prior distribution $p(x)$ may be factorized in a symmetric form as

$$p(x) = \prod_{(s,t)\in\mathcal{E}} \frac{p(x_s, x_t)}{p(x_s)\, p(x_t)} \prod_{s\in\mathcal{V}} p(x_s) \tag{2.19}$$

where $p(x_s)$ and $p(x_s, x_t)$ are exact marginal distributions. Equation (2.19) shows that, for graphs without cycles, it is possible to factor $p(x)$ using pairwise clique potentials which are simple functions of the local marginal distributions at neighboring nodes. However, such a factorization does *not* generally exist for graphs with cycles.

For any $s \in \mathcal{V}$ and any $t \in N(s)$, let $y_{s\setminus t}$ be the set of all observation nodes in the tree rooted at node s, excluding those in the subtree rooted at node t. Figure 2-4(a) provides a graphical illustration of this set. Then, using Bayes' rule and the Markov

properties implied by $\mathcal{G}$, we may decompose the marginal distribution $p(x_s \mid y)$ as

$$p(x_s \mid y) = \frac{p(y \mid x_s)\, p(x_s)}{p(y)} = \alpha p(x_s)\, p(y_s \mid x_s) \prod_{t \in N(s)} p\left(y_{t \setminus s} \mid x_s\right) \tag{2.20}$$

where α denotes a normalization constant[7] which is independent of x. From this decomposition, we see that for the purposes of calculating $p(x_s \mid y)$, the conditional likelihood $p\left(y_{t \setminus s} \mid x_s\right)$ is a *sufficient statistic* of the data in the subtree rooted at node t. Using the conditional independencies implied by $\mathcal{G}$, we can derive a self–consistent set of equations relating the conditional likelihoods at neighboring nodes:

$$\begin{aligned}
p\left(y_{t \setminus s} \mid x_s\right) &= \frac{p\left(x_s \mid y_{t \setminus s}\right) p\left(y_{t \setminus s}\right)}{p(x_s)} = \alpha \int_{x_t} \frac{p\left(x_s, x_t \mid y_{t \setminus s}\right)}{p(x_s)}\, dx_t \\
&= \alpha \int_{x_t} \frac{p(x_s, x_t)\, p\left(y_{t \setminus s} \mid x_s, x_t\right)}{p(x_s)}\, dx_t \\
&= \alpha \int_{x_t} \frac{p(x_s, x_t)\, p\left(y_{t \setminus s} \mid x_t\right)}{p(x_s)}\, dx_t \\
&= \alpha \int_{x_t} \left(\frac{p(x_s, x_t)}{p(x_s)\, p(x_t)}\right) p(x_t)\, p(y_t \mid x_t) \prod_{u \in N(t) \setminus s} p\left(y_{u \setminus t} \mid x_t\right)\, dx_t
\end{aligned} \tag{2.21}$$

Note that the prior model appears in equation (2.21) solely in terms of the potential functions found in the canonical tree–based factorization (2.19).

It is possible to modify equations (2.20, 2.21) to use the potential terms appearing in an arbitrary pairwise factorization (2.14). In particular, if we "unwrap" the conditional likelihood terms in equation (2.20) by repeated application of the local consistency conditions (2.21), we obtain

$$\begin{aligned}
p(x_u \mid y) &= \alpha \int_{x_{\mathcal{V} \setminus u}} \prod_{(s,t) \in \mathcal{E}} \frac{p(x_s, x_t)}{p(x_s)\, p(x_t)} \prod_{s \in \mathcal{V}} p(x_s)\, p(y_s \mid x_s)\, dx_{\mathcal{V} \setminus u} \\
&= \alpha \int_{x_{\mathcal{V} \setminus u}} \prod_{(s,t) \in \mathcal{E}} \psi_{s,t}(x_s, x_t) \prod_{s \in \mathcal{V}} p(y_s \mid x_s)\, dx_{\mathcal{V} \setminus u}
\end{aligned} \tag{2.22}$$

where the second equality follows because equations (2.19) and (2.14) provide two

[7]By convention, the normalization constant α is always chosen so that the function it multiplies integrates to unity. Thus, the specific numerical value of α may change from equation to equation.

equivalent representations of the same joint distribution $p(x)$. Thus, it is straightforward to show that the integral (2.22) which correctly computes $p(x_s \mid y)$ will also be produced by the following analogs of equations (2.20, 2.21):

$$p(x_s \mid y) = \alpha p(y_s \mid x_s) \prod_{t \in N(s)} m_{ts}(x_s) \tag{2.23}$$

$$m_{ts}(x_s) = \alpha \int_{x_t} \psi_{s,t}(x_s, x_t)\, p(y_t \mid x_t) \prod_{u \in N(t) \setminus s} m_{ut}(x_t)\, dx_t \tag{2.24}$$

For non–canonical factorizations of $p(x)$, the $m_{ts}(x_s)$ terms will in general not equal the conditional likelihoods $p(y_{t\setminus s} \mid x_s)$. Conceptually, however, $m_{ts}(x_s)$ may still be interpreted as a sufficient statistic of $y_{t\setminus s}$ for the purpose of calculating $p(x_s \mid y)$.

Equations (2.23, 2.24) show that the dependencies between the desired conditional marginal distributions $p(x_s \mid y)$ and sufficient statistics $m_{ts}(x_s)$ may be expressed using only *local* relationships between neighboring nodes. This naturally suggests the development of algorithms which solve these equations using a distributed set of local computations. The belief propagation (BP) algorithm begins by associating the sufficient statistic $m_{ts}(x_s)$ with a *message* that we would like node t to send to node s. This message provides all of the information about x_s which is available from $y_{t\setminus s}$, the subset of observations upon which x_s depends only through its correlation with x_t. Given an algorithm which efficiently calculates all of the messages, the marginal distributions $p(x_s \mid y)$, or "beliefs," are easily found from equation (2.23).

Belief propagation is typically described as a parallel algorithm in which equation (2.24) is iteratively applied, generating a sequence of messages $\{m_{ts}^n(x_s)\}$ which converge to $m_{ts}(x_s)$ as $n \to \infty$. In particular, for all $t \in \mathcal{V}$ and $s \in N(t)$, we initialize $m_{ts}^0(x_s)$ to some arbitrary initial value, typically $m_{ts}^0(x_s) = 1$, and then iteratively apply the following message update equation:

$$m_{ts}^n(x_s) = \alpha \int_{x_t} \psi_{s,t}(x_s, x_t)\, p(y_t \mid x_t) \prod_{u \in N(t) \setminus s} m_{ut}^{n-1}(x_t)\, dx_t \tag{2.25}$$

It is straightforward to show that, after a number of iterations equal to the diameter

D of $\mathcal{G}$, the messages will converge to the unique fixed point of the message update equation (2.25), so that $m_{ts}^{D}(x_s) = m_{ts}(x_s)$. This in turn allows $p(x_s \mid y)$ to be exactly calculated for all $s \in \mathcal{V}$. Even prior to convergence, however, the message values $m_{ts}^{n}(x_s)$ provide useful information. Let y_s^n be the set of observations that are separated from s in $\mathcal{G}$ by at most n edges, as illustrated in Figure 2-4(b). Then, assuming $m_{ts}^{0}(x_s) = 1$, we have

$$p(x_s \mid y_s^n) = \alpha p(y_s \mid x_s) \prod_{t \in N(s)} m_{ts}^{n}(x_s) \tag{2.26}$$

Thus, the parallel BP message updates effectively compute a series of approximations $\{p(x_s \mid y_s^n)\}$ to $p(x_s \mid y)$, each of which optimally incorporates all measurements in an expanding domain of observation.

While intuitively appealing, the parallel BP algorithm is inefficient if implemented directly on a serial computer. In fact, if the message updates (2.25) are *scheduled* in the proper order, each individual message must only be calculated once, and the exact marginals may be found with only $\mathcal{O}(N)$ such updates. One example of an efficient message scheduling begins by choosing a particular node to be the root of the tree. This induces a partial ordering of the nodes in *scale* according to their distance from the root. Each node, besides the root, then has a unique neighboring parent node which is closer to the root; the other neighbors are called child nodes. The inference procedure begins with an upward sweep which calculates all of the messages from child to parent nodes, starting with the most distant scales and proceeding recursively to the root. Then, a downward sweep calculates all of the parent to child messages starting from the root and proceeding outward. For more details, see [20].

The BP update equations (2.25, 2.26) will, in principle, calculate the correct marginals $p(x_s \mid y)$ for an arbitrary tree–structured Markov random field. In practice, however, the message update integral (2.25) is intractable for many distributions $p(x)$. Fortunately, there exist two general classes of tractable distributions. If the hidden variables x_s take values from a discrete set of dimension m, the integrals become sums which can be calculated in $\mathcal{O}(m^2)$ operations per message update. Alternatively,

for jointly Gaussian MRFs, all conditional distributions are also Gaussian, and the integrals may be replaced by recursions on the mean and covariance parameters of these Gaussians. Each message update at node s requires $\mathcal{O}(d_s^3)$ operations. The following section explores these updates in detail.

2.3.2 Gaussian Belief Propagation

In this section, we specialize the BP update equations (2.25, 2.26) to the Gaussian graphical models introduced in previous sections. Recall that, for the purpose of calculating $p(x_s \mid y)$, equation (2.20) shows that the conditional likelihoods $\{p(y_{t\setminus s} \mid x_s)\}_{t \in N(s)}$ provide a set of sufficient statistics, where

$$p(y_{t\setminus s} \mid x_s) = \alpha \frac{p(x_s \mid y_{t\setminus s})}{p(x_s)} \tag{2.27}$$

Because x and y are jointly Gaussian, $p(x_s)$ and $p(x_s \mid y_{t\setminus s})$ are also Gaussian, and therefore have a finite–dimensional parameterization in terms of their mean and covariance. When viewed as a function of x_s for a fixed set of observations y, the likelihood $p(y_{t\setminus s} \mid x_s)$ does *not* correspond to a probability distribution. However, equation (2.27) shows that $p(y_{t\setminus s} \mid x_s)$ is still proportional to an exponentiated quadratic form in x_s. Since the proportionality constant is independent of x_s, we may parameterize $p(y_{t\setminus s} \mid x_s)$ with a "mean" vector and "covariance" matrix, just as we would a Gaussian probability distribution. Furthermore, using the structure of Gaussian clique potentials (2.15), it is straightforward to show that the BP messages $m_{ts}(x_s)$ may also be parameterized by a mean and covariance. This parameterization is important because it allows the integral BP update equation (2.25) to be transformed into a set of linear algebraic equations.

It is possible to develop a set of Gaussian belief propagation equations which act directly on the mean and covariance parameters of the messages $m_{ts}(x_s)$. However, just as §2.2.3 showed that Gaussian prior models are most naturally parameterized by their sparse *inverse* covariance matrix $J = P^{-1}$, Gaussian BP is simplest when the messages are represented in an *information* form [44]. For a Gaussian distribution

with mean μ and covariance $\mathcal{P}$, the information parameters are defined by

$$\vartheta = \mathcal{P}^{-1}\mu \qquad \Lambda = \mathcal{P}^{-1} \tag{2.28}$$

We use the notation $\mathcal{N}^{-1}(\vartheta, \Lambda)$ to indicate a Gaussian probability distribution with information parameters ϑ and Λ. For a more detailed introduction to information parameters, see Appendix B.1. We will denote the information parameters of the BP messages and conditional marginal distributions as

$$m_{ts}^{n}(x_s) = \alpha\mathcal{N}^{-1}(\vartheta_{ts}^{n}, \Lambda_{ts}^{n}) \qquad p(x_s \mid y_s^n) = \mathcal{N}^{-1}(\vartheta_s^n, \Lambda_s^n) \tag{2.29}$$

$$m_{ts}(x_s) = \alpha\mathcal{N}^{-1}(\vartheta_{ts}, \Lambda_{ts}) \qquad p(x_s \mid y) = \mathcal{N}^{-1}(\vartheta_s, \Lambda_s) \tag{2.30}$$

where equation (2.29) gives the values at iteration n, and equation (2.30) gives the corresponding steady–state values. The moment parameters of $p(x_s \mid y) \sim \mathcal{N}\left(\widehat{x}_s, \widehat{P}_s\right)$ are then related to the information parameters as follows:

$$\widehat{x}_s = (\Lambda_s)^{-1}\vartheta_s \qquad \widehat{P}_s = (\Lambda_s)^{-1} \tag{2.31}$$

Notice that Λ's are used to denote the inverse covariance parameters of messages and beliefs, while J's denote elements of the inverse covariance of the prior model.

To derive the Gaussian BP equations, we must determine the information parameterizations of all of the terms appearing in the general BP equations (2.25, 2.26). The pairwise clique potentials are already expressed in an information form by equation (2.15). Using Bayes' rule, we can write the local observation terms as

$$p(y_s \mid x_s) = \alpha\frac{p(x_s \mid y_s)}{p(x_s)} \tag{2.32}$$

Clearly, because $x \sim \mathcal{N}(0, P)$, $p(x_s) = \mathcal{N}^{-1}\left(0, P_{s,s}^{-1}\right)$. Combining the information form of the normal equations (2.6, 2.7) with the local measurement model (2.18), we find that $p(x_s \mid y_s) = \mathcal{N}^{-1}\left(C_s^T R_s^{-1} y_s, P_{s,s}^{-1} + C_s^T R_s^{-1} C_s\right)$. Then, taking the quotient

of these terms using equations (B.8, B.9) gives

$$p\left(y_s \mid x_s\right) = \alpha \mathcal{N}^{-1}\left(C_s^T R_s^{-1} y_s, C_s^T R_s^{-1} C_s\right) \tag{2.33}$$

Note that because the update equations employ likelihoods $p\left(y_s \mid x_s\right)$ rather than conditional densities $p\left(x_s \mid y_s\right)$, we are able to avoid explictly computing the marginal variances $P_{s,s}$ of the prior model.

Now consider the belief update equation (2.26). By repeated application of the rules for combining products of Gaussian densities (see equations (B.8, B.9)), we see immediately that $p\left(x_s \mid y_s^n\right) = \mathcal{N}^{-1}(\vartheta_s^n, \Lambda_s^n)$ is given by

$$\vartheta_s^n = C_s^T R_s^{-1} y_s + \sum_{t \in N(s)} \vartheta_{ts}^n \tag{2.34}$$

$$\Lambda_s^n = C_s^T R_s^{-1} C_s + \sum_{t \in N(s)} \Lambda_{ts}^n \tag{2.35}$$

The message update equation (2.25) is slightly more complicated. By repeated application of the product formulas, we find that

$$\psi_{s,t}\left(x_s, x_t\right) p\left(y_t \mid x_t\right) \prod_{u \in N(t) \backslash s} m_{ut}^{n-1}\left(x_t\right) \propto \mathcal{N}^{-1}\left(\overline{\vartheta}, \overline{\Lambda}\right) \tag{2.36}$$

where

$$\overline{\vartheta} = \begin{bmatrix} \mathbf{0} \\ C_t^T R_t^{-1} y_t + \sum_{u \in N(t) \backslash s} \vartheta_{ut}^{n-1} \end{bmatrix} \tag{2.37}$$

$$\overline{\Lambda} = \begin{bmatrix} J_{s(t)} & J_{s,t} \\ J_{t,s} & J_{t(s)} + C_t^T R_t^{-1} C_t + \sum_{u \in N(t) \backslash s} \Lambda_{ut}^{n-1} \end{bmatrix} \tag{2.38}$$

and $\psi_{s,t}\left(x_s, x_t\right)$ is parameterized as in equation (2.15). Applying the marginalization equations (B.6, B.7) to perform the integration over x_t, we find the following parallel

update equations for the BP messages $m_{ts}^n(x_s) = \alpha \mathcal{N}^{-1}(\vartheta_{ts}^n, \Lambda_{ts}^n)$:

$$\vartheta_{ts}^n = -J_{s,t}\left(J_{t(s)} + C_t^T R_t^{-1} C_t + \sum_{u \in N(t)\backslash s} \Lambda_{ut}^{n-1}\right)^{-1}\left(C_t^T R_t^{-1} y_t + \sum_{u \in N(t)\backslash s} \vartheta_{ut}^{n-1}\right) \quad (2.39)$$

$$\Lambda_{ts}^n = J_{s(t)} - J_{s,t}\left(J_{t(s)} + C_t^T R_t^{-1} C_t + \sum_{u \in N(t)\backslash s} \Lambda_{ut}^{n-1}\right)^{-1} J_{t,s} \quad (2.40)$$

These equations are the Gaussian form of the parallel BP message update equation (2.25). The initial conditions ϑ_{ts}^0 and Λ_{ts}^0 may be arbitrarily chosen; typically, one sets $\vartheta_{ts}^0 = \mathbf{0}$ and $\Lambda_{ts}^0 = \mathbf{0}$. Note that these updates may be easily adapted to other, more efficient message schedules as discussed in §2.3.1. Due to the matrix inversion, the cost of computing the message update at node t is $\mathcal{O}(d_t^3)$. Upon convergence, the information parameters of $p(x_s \mid y)$ may be found from equations (2.34, 2.35).

Equations (2.39, 2.40), which represent the standard form of the Gaussian BP message updates, appear to depend on the chosen factorization of the inverse covariance matrix J into clique potentials. Consider, however, the matrix which is inverted in each of the update equations. Each incoming message Λ_{ut} is a linear combination of $J_{t(u)}$ with some other terms. Looking at these $J_{t(u)}$ together with the locally contributed $J_{t(s)}$ term, we find that

$$J_{t(s)} + \sum_{u \in N(t)\backslash s} J_{t(u)} = \sum_{u \in N(t)} J_{t(u)} = J_{t,t} \quad (2.41)$$

where the last equality follows from the consistency condition (2.16). From this, we see that although the BP messages themselves depend on the choice of clique potentials, the calculated marginal distributions (or beliefs) do not. In fact, one may define an alternate sequence of messages whose update depends directly on the (unfactorized)

block diagonal entries of J as

$$\vartheta_{ts}^n = -J_{s,t}\left(J_{t,t} + C_t^T R_t^{-1} C_t + \sum_{u \in N(t)\backslash s} \Lambda_{ut}^{n-1}\right)^{-1}\left(C_t^T R_t^{-1} y_t + \sum_{u \in N(t)\backslash s} \vartheta_{ut}^{n-1}\right) \quad (2.42)$$

$$\Lambda_{ts}^n = -J_{s,t}\left(J_{t,t} + C_t^T R_t^{-1} C_t + \sum_{u \in N(t)\backslash s} \Lambda_{ut}^{n-1}\right)^{-1} J_{t,s} \quad (2.43)$$

where the local marginal distributions $p(x_s \mid y_s^n) = \mathcal{N}^{-1}(\vartheta_s^n, \Lambda_s^n)$ are now given by

$$\vartheta_s^n = C_s^T R_s^{-1} y_s + \sum_{t \in N(s)} \vartheta_{ts}^n \quad (2.44)$$

$$\Lambda_s^n = J_{s,s} + C_s^T R_s^{-1} C_s + \sum_{t \in N(s)} \Lambda_{ts}^n \quad (2.45)$$

Note that the standard (2.39, 2.40) and factorization independent (2.42, 2.43) forms of the BP message update equations will in general produce different messages $(\vartheta_{ts}^n, \Lambda_{ts}^n)$ at each iteration. However, if the initial conditions are appropriately chosen, both sequences of messages will produce the *same* conditional distributions $p(x_s \mid y_s^n)$ for every node at every iteration. These alternative BP update equations are especially useful when the prior model is specified by an inverse covariance J, because it is possible to use the BP algorithm without first forming clique potentials. In addition, they reveal certain invariants of the BP algorithm which may aid future analyses of its performance on graphs with cycles, as discussed in the following section.

2.3.3 Inference for Graphs with Cycles

As demonstrated in the previous sections, tree–structured graphs lead to exact, efficient inference algorithms. Unfortunately, however, the graphical models which arise in many practical applications have large numbers of cycles. For such graphs, the local conditional independencies that the BP algorithm uses to integrate information from neighboring nodes do not exist, and the derivation in §2.3.1 breaks down.

The *junction tree* algorithm [43, 48] provides one way of extending tree–based in-

ference procedures to more general graphs. It operates in three stages. In the first stage, edges are added to $\mathcal{G}$ until it is *triangulated*.[8] Then, a tree is formed from the maximal cliques of the triangulated graph. Finally, an exact inference algorithm, which is equivalent to belief propagation, is performed on the tree of cliques. The triangulation step ensures that the clique tree satisfies the *running intersection property*, which requires that any variables shared by two clique nodes also be members of each clique on their unique connecting path. This property *must* be satisfied for local computations on the clique tree to produce globally consistent estimates. Unfortunately, for most interesting graphs with cycles, triangulation greatly increases the dimension d of the resulting clique variables. As tree–based inference procedures require $\mathcal{O}(Nd^3)$ operations, the junction tree algorithm is often no more tractable than direct matrix inversion.

Because direct methods are generally intractable, a wide range of iterative inference algorithms have been developed for graphs with cycles. One of the most popular is known as *loopy belief propagation* [79]. The loopy BP algorithm operates by iterating the parallel BP message passing equations, as derived in §2.3.1, on a graph with cycles. Because of the presence of loops, the messages lose their strict probabilistic interpretation, and the exact answer will not be achieved after any finite number of iterations. In some cases, the iterations may not converge at all. However, for many graphs, especially those arising in error–correcting codes [34, 51], loopy BP converges to beliefs which very closely approximate the true conditional marginal distributions [33, 52]. The standard BP derivation, as given in §2.3.1, provides no justification for loopy BP, other than the vague intuition that belief propagation should perform well for graphs whose cycles are "long enough."

For Gaussian Markov random fields, loopy BP has been analyzed in some detail. It was examined by Rusmevichientong and Van Roy [62] for the specific graphs used by turbo codes [51], and for general MRFs by Weiss and Freeman [80]. They show that when loopy BP does converge, it always calculates the correct conditional

[8] In a triangulated graph, every cycle of length four or greater has a chord, that is, an edge joining two nonconsecutive vertices of the cycle.

means. However, the error covariances are incorrect because the correlations induced by the cycles are not properly accounted for. Weiss and Freeman were also able to demonstrate a relationship between fast convergence and good approximations of the error covariances, and to provide a basic set of sufficient, but overly conservative, conditions guaranteeing the convergence of loopy BP.

More generally, researchers are beginning to develop a deeper theoretical understanding of loopy BP's behavior. In particular, it has been shown that belief propagation is intimately connected to the Bethe approximation of statistical physics [82, 83]. Loopy BP can be seen as minimizing a particular approximation to the relative entropy between the estimated beliefs and true pairwise marginal distributions [76]. The BP messages correspond to exponentiated sums of Lagrange multipliers that enforce the pairwise marginalization constraints (e.g. $\sum_{x_t} p(x_s, x_t) = p(x_s)$ for all $(s,t) \in \mathcal{E}$) which any locally consistent inference solution must satisfy [83, 84]. This connection has led to a much better understanding of the nature of the approximation made by loopy BP [76, 77], as well as a class of generalized belief propagation algorithms [83, 84] with superior performance.

2.4 Iterative Solution of Linear Systems

As shown by equation (2.6) of §2.1, the conditional mean $\widehat{x}$ of a Gaussian inference problem can be viewed as the solution of a linear system of equations. Thus, the problem of calculating $\widehat{x}$ is equivalent to the general linear algebraic problem of solving

$$Ax = b \tag{2.46}$$

for some symmetric positive definite matrix A. As discussed in §2.2.2, for the graphical inference problems examined in this thesis A will typically be sparse. Similar sparse, positive definite linear systems arise in many fields of science and engineering, perhaps most commonly from the discretization of elliptic partial differential equations [25]. For this reason, they have been widely studied in the numerical lin-

ear algebra community, and many different algorithms have been proposed for their solution.

In this section, we briefly introduce a few of the most effective and relevant linear algebraic techniques. We begin in §2.4.1 by showing how the simple notion of a matrix splitting may be used to generate an entire family of iterative algorithms. Then, in §2.4.2, we briefly present the conjugate gradient iteration, which is generally considered to be the most effective method for solving sparse positive definite systems. We conclude by discussing stopping criteria for iterative methods. Note that none of the techniques presented in this section directly address the complementary inference problem of calculating entries of the error covariance matrix $\widehat{P}$.

2.4.1 Stationary Richardson Methods

In this section, we consider equation (2.46), assuming only that $A > 0$. The unique solution $x = A^{-1}b$ must clearly satisfy the following equation:

$$x = x + (b - Ax) \tag{2.47}$$

Given some initial guess x^0, equation (2.47) naturally suggests the generation of a sequence of iterates $\{x^n\}_{n=1}^{\infty}$ according to the recursion

$$x^n = x^{n-1} + (b - Ax^{n-1}) = x^{n-1} + r^{n-1} = (I - A)x^{n-1} + b \tag{2.48}$$

where $r^n \triangleq (b - Ax^n)$ is the *residual* at the n^{th} iteration. This recursion is known as a *Richardson iteration* [46, 85]. It is stationary because the previous iterate is multiplied by the same matrix $(I - A)$ at each iteration.

The behavior of a linear system, like that defined by equation (2.48), is determined by its eigenvalues. Let the set of all eigenvalues of a matrix A by denoted by $\{\lambda_i(A)\}$. The spectral radius is then defined as $\rho(A) \triangleq \max_{\lambda \in \{\lambda_i(A)\}} |\lambda|$. It is well known that x^n will converge to the unique fixed point $x = A^{-1}b$, for arbitrary x^0, if and only if $\rho(I - A) < 1$. The asymptotic convergence rate is equal to $\rho(I - A)$.

The dynamic behavior of equation (2.48) depends heavily on A itself. In particular, convergence will be fastest when $\{\lambda_i(A)\}$ are concentrated around 1, while the iteration will diverge if $\rho(A) > 2$. To correct this problem, we can transform the linear system (2.46) into an alternate system which has the same solution, but more favorable spectral properties. In particular, for any invertible matrix M, we may write

$$M^{-1}Ax = M^{-1}b \tag{2.49}$$

The matrix M is known as a *preconditioner* [7, 25], and naturally leads to the following preconditioned Richardson iteration:

$$x^n = (I - M^{-1}A)x^{n-1} + M^{-1}b \tag{2.50}$$

If M is chosen so that the eigenvalues of $M^{-1}A$ are closer to 1 than those of the original system A, the preconditioned iteration (2.50) may converge after a smaller number of iterations. However, each iteration is more costly, because it is necessary to multiply Ax^{n-1} by M^{-1}, or equivalently to solve a linear system of the form $M\bar{x} = \bar{b}$. Thus, in addition to well approximating A, an effective preconditioner must ensure that each evaluation of equation (2.50) is not too difficult.

Preconditioners are frequently generated using a *matrix splitting* [25, 37, 46, 85] $A = M - K$ of the linear system A, where K is chosen so that $M = A+K$ is invertible. For any such splitting, the corresponding preconditioned Richardson iteration is

$$\begin{aligned} x^n &= \left(I - (A+K)^{-1}A\right)x^{n-1} + (A+K)^{-1}b \\ &= (A+K)^{-1}\left(Kx^{n-1} + b\right) \end{aligned} \tag{2.51}$$

Many classic iterative algorithms are generated by appropriately chosen splittings. Let $A = L + D + U$ where L, D, and U are lower triangular, diagonal, and upper triangular, respectively. The *Gauss–Jacobi* method chooses $M = D$ and $K = -(L+U)$. The resulting preconditioner is particularly easy to apply, simply requiring Ax^{n-1} to be rescaled by D^{-1} at each iteration. However, unless A is strongly diagonally dom-

inant, it is generally not very effective. A slightly more sophisticated splitting sets $M = (L + D)$ and $K = -U$. Because M is lower triangular, each iteration of the resulting *Gauss–Seidel* preconditioner is easily found by back–substitution. Finally, the *successive overrelaxation* (SOR) method attempts to accelerate the Gauss–Seidel iteration by adjusting the step size. For more details on these stationary methods, see [7, 25, 37, 85]. Note, however, that none of the classical stationary iterations are competitive with the conjugate gradient method presented in the following section.

2.4.2 The Conjugate Gradient Iteration

Krylov subspace methods, of which the *conjugate gradient* (CG) iteration is a special case, are a class of nonstationary iterative methods commonly used to solve sparse linear systems. The Krylov subspace of dimension k generated by a matrix A and vector r is defined as

$$\mathcal{K}_k(A, r) \triangleq \text{span}\left(r, Ar, A^2r, \ldots, A^{k-1}r\right) \tag{2.52}$$

Krylov subspace methods compute a sequence of iterates x^n which optimize some objective function over Krylov subspaces of successively increasing dimension. The conjugate gradient method examined in this section is by far the most widely used Krylov subspace method for positive definite systems. For more details on CG and other Krylov subspace methods, see [7, 25, 37, 46].

The CG iteration is based on the Krylov subspaces generated by A and an initial residual $r^0 = b - Ax^0$, where x^0 is some initial guess for the solution of equation (2.46). At the n^{th} iteration, CG selects the vector x^n as

$$\begin{aligned} x^n &= \underset{\bar{x} \in x^0 \oplus \mathcal{K}_n(A, r^0)}{\arg\min} \left(A\bar{x} - b\right)^T A^{-1} \left(A\bar{x} - b\right) \\ &= \underset{\bar{x} \in x^0 \oplus \mathcal{K}_n(A, r^0)}{\arg\min} ||A\bar{x} - b||_{A^{-1}} \end{aligned} \tag{2.53}$$

CG is particularly effective because the minimization in equation (2.53) can be performed very efficiently, requiring only a few matrix–vector products involving the

matrix A and some inner products (see [25] or [37] for explicit derivations). For graph–structured inference problems like those introduced in §2.3, the cost of each CG iteration on a graph with N nodes of dimension d is $\mathcal{O}(Nd^2)$ operations. Note that, ignoring finite precision effects, CG is guaranteed to converge after at most $\dim(A)$ iterations. Typically, however, the magnitude $||r^n||$ of the normalized residual drops quite quickly, and fewer iterations are necessary.

The convergence rate of the conjugate gradient iteration may be analyzed using matrix polynomials. As shown by [25, p. 313],

$$\frac{||r^n||_{A^{-1}}}{||r^0||_{A^{-1}}} \leq \min_{p_n \in \mathcal{P}_n} \max_{\lambda \in \{\lambda_i(A)\}} |p_n(\lambda)| \tag{2.54}$$

where $\mathcal{P}_n$ is the set of all n^{th} order polynomials p_n such that $p_n(0) = 1$. Therefore, the convergence of the CG iteration is closely related to how well the eigenspectrum may be fit by a polynomial. If we restrict $\mathcal{P}_n$ to the class of Chebyshev polynomials, the following well–known bound is attained:

$$\frac{||r^n||_{A^{-1}}}{||r^0||_{A^{-1}}} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^n \qquad \kappa \triangleq \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} \tag{2.55}$$

Here, κ is the *condition number* of the matrix A. Note that CG's performance is best when the eigenvalues are tightly clustered so that κ is small.

To improve conjugate gradient's convergence rate, we would like to consider a preconditioned system $M^{-1}Ax = M^{-1}b$ as in §2.4.1. However, even if M is symmetric, $M^{-1}A$ will generally not be, and therefore apparently cannot be solved using CG. Suppose instead that a square root decomposition $M^{-1} = Q^TQ$ of the preconditioner is available. We could then form a symmetric preconditioned system as

$$(QAQ^T)(Q^{-T}x) = Q^Tb \tag{2.56}$$

with the desired eigenspectrum ($\{\lambda_i(QAQ^T)\} = \{\lambda_i(M^{-1}A)\}$). Fortunately, in the case of the CG iteration, it is not necessary to explicitly compute this square root decomposition, because the method can be rewritten so that $M^{-1} = Q^TQ$ is only

applied in its entirety (see [25, 37]). Thus, any symmetric preconditioning matrix may be easily integrated into the conjugate gradients algorithm.

Although CG computes the desired solution $A^{-1}b$, it does not explicitly calculate any entries of A^{-1} in the process, and thus does not directly provide error covariance information. For inference problems defined by the standard normal equations (2.1, 2.2), Schneider [65, 67] has developed a method for extracting the error variances directly from the search directions generated by the CG iteration. Similar methods may also be applied to the information form of the normal equations (2.6, 2.7) considered in this thesis [9, 57]. However, accurate error variances cannot be obtained without reorthogonalization of the search directions, which may greatly increase the computational cost of the CG iteration. In addition, the error variances often converge quite slowly relative to the conditional mean estimates.

2.4.3 Stopping Criteria

When applying any of the iterative methods considered in the previous sections, it is necessary to determine when to stop the iteration, i.e. to decide that x^n is a sufficiently close approximation to $A^{-1}b$. Given a tolerance parameter ϵ, we would ideally like to iterate until

$$||A^{-1}b - x^n||_2 \leq \epsilon \tag{2.57}$$

Unfortunately, because $A^{-1}b$ is unavailable, equation (2.57) cannot be evaluated. However, the residual $r^n = b - Ax^n$ *is* available. Therefore, as suggested in [7], we may instead iterate until

$$\frac{||r^n||_2}{||b||_2} \leq \epsilon \tag{2.58}$$

Using the identity $(A^{-1}b - x^n) = A^{-1}r^n$, equation (2.58) is easily shown to yield the following bound on the final error:

$$||A^{-1}b - x^n||_2 \leq \epsilon \sigma_{\min}(A)||b||_2 \tag{2.59}$$

All of the simulations in this thesis use this residual–based stopping criterion.

Chapter 3

Embedded Trees

In this chapter, we develop a set of novel, iterative algorithms for exactly solving Gaussian inference problems defined on graphs with cycles. These algorithms operate by exploiting the presence of tractable substructures within complex graphs. As discussed in §2.3, estimation problems on acyclic, or tree–structured, graphs are easily solved by direct recursive procedures. By removing edges from graphs with cycles, we may reveal a wide variety of trees embedded within the original graph. We demonstrate that this idea may be used to construct sequences of tractable subproblems whose solution rapidly converges to that of the original loopy graph problem.

We begin in §3.1 by developing the idea of embedded trees in more detail. Then, in §3.2, we show how such trees can be exploited to calculate iteratively the conditional mean of a loopy Gaussian inference problem. For appropriately chosen spanning trees, the resulting embedded trees algorithm [78] exhibits rapid convergence rates which compare favorably with existing techniques. In the following section (§3.3), we show how embedded trees may also be used to calculate error covariances correctly for problems on graphs with cycles. Then, in §3.4, we provide a detailed analysis of the convergence behavior of the algorithms presented in the previous two sections. We conclude with a set of numerical experiments comparing the performance of the proposed algorithms to existing techniques.

3.1 Loopy Graphs and Spanning Trees

As discussed in §2.3, inference problems defined on acylic graphs may be efficiently solved by direct recursive algorithms. These algorithms depend on the critical fact that for tree–structured graphs, there is a single unique path between each pair of nodes. This uniqueness allows the correlation between any two nodes to be expressed as a simple function of the pairwise clique potentials associated with the edges connecting them. For graphs with cycles, however, there are many paths between pairs of nodes, and correlations are a complex function of all of these paths. This prevents the construction of a partial node ordering which maintains the conditional independencies necessary for recursive inference.

Exact inference techniques for graphs with cycles, such as the junction tree algorithm, apply recursive inference algorithms to an acyclic graph formed by clustering nodes. By clustering nodes, these algorithms also implicitly cluster the many paths connecting each pair of nodes, properly integrating the correlations produced by those paths. Unfortunately, for complex graphs with many cycles, exactly considering the full correlation structure in parallel is prohibitively costly.

In this thesis, we instead propose a set of inference algorithms which *sequentially* consider different subsets of the correlation paths contained in the original graph. Each iteration of these algorithms involves a set of exact computations on a *spanning tree* of the original graph. For a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a spanning tree $\mathcal{G}_\mathcal{T} = (\mathcal{V}, \mathcal{E}_\mathcal{T})$ is defined to be a connected subgraph ($\mathcal{E}_\mathcal{T} \subset \mathcal{E}$) which has no cycles. As Figure 3-1 illustrates, there are typically a large number of possible spanning trees embedded within graphs with cycles.

For Gaussian graphical models, spanning trees are closely connected to the structural properties of the inverse covariance matrix. Consider a Gaussian stochastic process $x \sim \mathcal{N}^{-1}(0, J)$ which is Markov with respect to an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. By Theorem 2.2, for any $s, t \in \mathcal{V}$ such that $s \neq t$, $J_{s,t}$ will be nonzero if and only if $(s, t) \in \mathcal{E}$. Thus, modifications of the edge set $\mathcal{E}$ are precisely equivalent to changes in the locations of the nonzero off–diagonal entries of J. In particular, consider a mod-

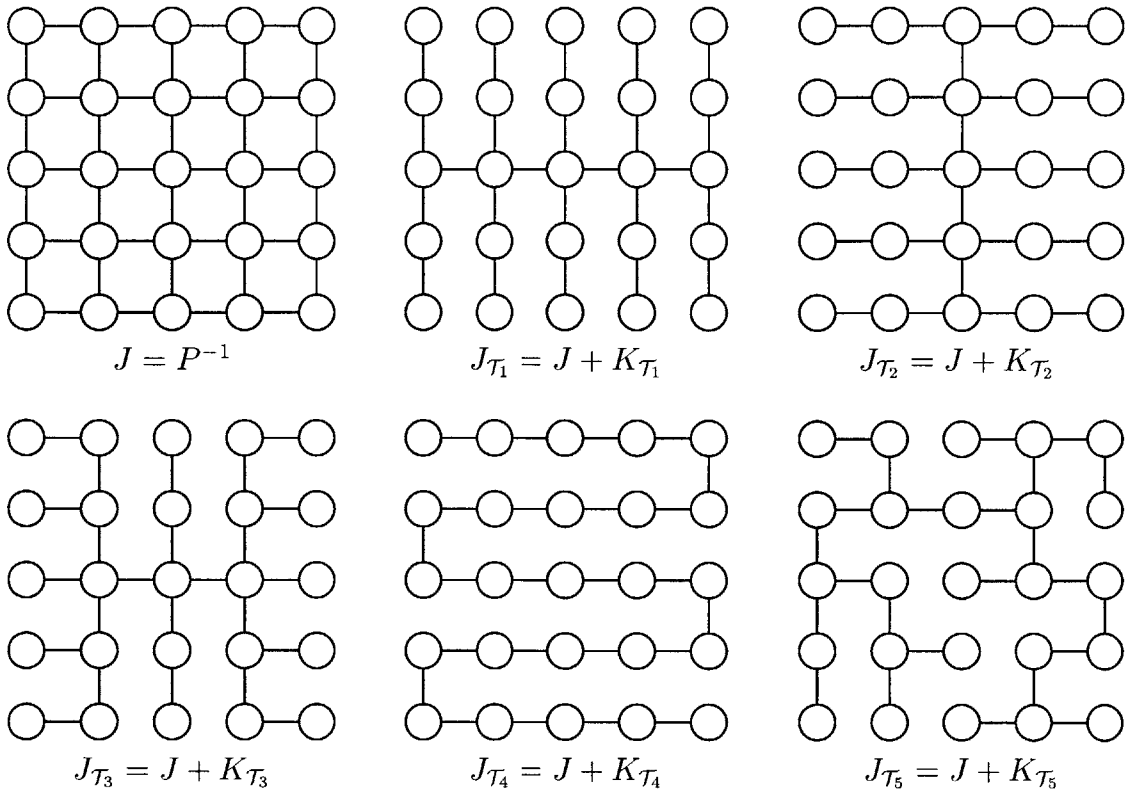


Figure 3-1: Embedded trees produced by five different cutting matrices $\{K_{\mathcal{T}_i}\}_{i=1}^{5}$ for a nearest-neighbor grid. Observation nodes are not shown.

ified stochastic process $x_{\mathcal{T}} \sim \mathcal{N}^{-1}(0, J_{\mathcal{T}})$ which is Markov with respect to a spanning tree $\mathcal{G}_{\mathcal{T}} = (\mathcal{V}, \mathcal{E}_{\mathcal{T}})$. For any such $J_{\mathcal{T}}$, there exists a symmetric matrix $K_{\mathcal{T}}$ such that

$$J_{\mathcal{T}} = J + K_{\mathcal{T}} \tag{3.1}$$

Because it acts to remove edges from the graph, $K_{\mathcal{T}}$ is called a *cutting matrix*. As Figure 3-1 illustrates, different cutting matrices may produce different spanning trees of the original graph. Note that the cutting matrix $K_{\mathcal{T}}$ also defines a matrix splitting $J = (J_{\mathcal{T}} - K_{\mathcal{T}})$ as introduced in §2.4.1.

Certain elements of the cutting matrix are uniquely defined by the choice of spanning tree $\mathcal{G}_{\mathcal{T}}$. For example, each discarded edge constrains the corresponding off-diagonal blocks of $K_{\mathcal{T}}$:

$$(K_{\mathcal{T}})_{s,t} = -J_{s,t} \qquad (s,t) \in \mathcal{E} \setminus \mathcal{E}_{\mathcal{T}} \tag{3.2}$$

However, other entries of $K_{\mathcal{T}}$, such as the diagonal elements, are not constrained by graph structure. Consequently, there exist many different cutting matrices $K_{\mathcal{T}}$, and associated inverse covariances $J_{\mathcal{T}}$, corresponding to a given spanning tree $\mathcal{G}_{\mathcal{T}}$.

In later sections, it will be useful to define a restricted class of *regular* cutting matrices. As with any cutting matrix, the off–diagonal entries corresponding to cut edges are given by equation (3.2). However, all other off–diagonal entries of a regular cutting matrix must be zero. In addition, the block diagonal entries for nodes from which no edge is cut must be zero. Thus, for a given $\mathcal{G}_{\mathcal{T}}$, the corresponding family of regular cutting matrices differs only in the block diagonal entries corresponding to nodes involved in at least one cut edge.

As discussed in the Introduction, many potentially interesting classes of graphical models are "nearly" tree–structured. For such models, it is possible to reveal an embedded spanning tree by removing a small (relative to N) number of edges. Let $E \triangleq |\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}|$ denote the number of discarded or "cut" edges. Clearly, any regular cutting matrix $K_{\mathcal{T}}$ removing E edges may have nonzero entries in at most $2Ed$ columns, implying that $\text{rank}(K_{\mathcal{T}}) \leq \mathcal{O}(Ed)$. Thus, for sparsely connected graphical models where $E \ll N$, cutting matrices may always be chosen to have *low rank*. Many of the algorithms developed later in this thesis explicitly exploit this property, leading to drastically reduced computational costs for sufficiently sparse graphs.

In this section, we have focused on spanning trees as particular examples of subgraphs for which exact calculations are tractable. However, there exist many other tractable options. For example, graphs with a single cycle, which are sometimes known as boundary value systems, may be solved nearly as efficiently as acyclic graphs [53]. More generally, we could consider triangulated graphs of bounded clique size for which exact solution by the junction tree algorithm is feasible. In most cases, the algorithms developed in this thesis do not depend on the details of the belief propagation algorithm, and would extend equally well to these more general exact inference subroutines. For concreteness, however, we will focus most of our attention on tree–structured subgraphs.

3.2 Iterative Calculation of Conditional Means

In this section, we return to the graphical inference problem introduced in §2.3. For the moment, we focus solely on the calculation of the mean $\widehat{x}$ of the conditional distribution $p(x \mid y) \sim \mathcal{N}\left(\widehat{x}, \widehat{P}\right)$. As before, the prior model $x \sim \mathcal{N}(0, P)$ will be parameterized by a sparse inverse covariance matrix $J = P^{-1}$. The normal equations (2.6, 2.7) defining the conditional mean may be rewritten in terms of the inverse error covariance matrix $\widehat{J} \triangleq \widehat{P}^{-1}$ as follows:

$$\widehat{J}\,\widehat{x} = C^T R^{-1} y \tag{3.3}$$

$$\widehat{J} = J + C^T R^{-1} C \tag{3.4}$$

Throughout this section, we let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote the graph with respect to which the conditional distribution $p(x \mid y)$ is Markov. If the observations are local, as in equation (2.18), then $\mathcal{G}$ will match the graphical structure of the prior model $p(x)$. More generally, however, non–local observations may introduce extra edges.

3.2.1 Exploiting Embedded Trees

As demonstrated in §2.3.2, Gaussian inference problems defined on tree–structured graphs may be exactly solved using an efficient set of recursions. In this section, we combine these recursions with the spanning trees introduced in §3.1 to develop an iterative inference algorithm for graphs with cycles. Let $\mathcal{G}_{\mathcal{T}} = (\mathcal{V}, \mathcal{E}_{\mathcal{T}})$ be a spanning tree of $\mathcal{G}$, and $\widehat{J}_{\mathcal{T}}$ the inverse covariance matrix of a stochastic process which is Markov with respect to $\mathcal{G}_{\mathcal{T}}$. Then, letting $K_{\mathcal{T}} = (\widehat{J} - \widehat{J}_{\mathcal{T}})$ be the cutting matrix corresponding to $\widehat{J}_{\mathcal{T}}$, we may rewrite equation (3.3) as

$$\begin{aligned} \left(\widehat{J} + K_{\mathcal{T}}\right)\widehat{x} &= K_{\mathcal{T}}\widehat{x} + C^T R^{-1} y \\ \widehat{J}_{\mathcal{T}}\,\widehat{x} &= K_{\mathcal{T}}\widehat{x} + C^T R^{-1} y \end{aligned} \tag{3.5}$$

By comparison to equation (3.3), we see that equation (3.5) corresponds to a tree–structured Gaussian inference problem, with a set of perturbed observations given by $(K_{\mathcal{T}}\widehat{x} + C^T R^{-1} y)$.

Because the "observations" $(K_{\mathcal{T}}\widehat{x} + C^T R^{-1} y)$ depend on the conditional mean $\widehat{x}$, equation (3.5) does not provide a direct solution to the original inference problem. However, it does suggest a natural iterative solution. Let $\{\mathcal{G}_{\mathcal{T}^n}\}_{n=1}^{\infty}$ be a sequence of spanning trees of $\mathcal{G}$, and $\{K_{\mathcal{T}^n}\}_{n=1}^{\infty}$ a corresponding sequence of cutting matrices such that $\widehat{J}_{\mathcal{T}^n} = (\widehat{J} + K_{\mathcal{T}^n})$ is Markov with respect to $\mathcal{G}_{\mathcal{T}^n}$. Then, from some initial guess $\widehat{x}^0$, we may generate a sequence of iterates $\{\widehat{x}^n\}_{n=1}^{\infty}$ using the recursion

$$\widehat{J}_{\mathcal{T}^n}\widehat{x}^n = K_{\mathcal{T}^n}\widehat{x}^{n-1} + C^T R^{-1} y \tag{3.6}$$

If the cutting matrix $K_{\mathcal{T}^n}$ is chosen so that $\widehat{J}_{\mathcal{T}^n}$ is positive definite, equation (3.6) is precisely equivalent to a Gaussian inference problem defined on a tree–structured Markov random field. It can therefore be solved using the belief propagation recursions introduced in §2.3.2, allowing $\widehat{x}^n$ to be calculated as

$$\widehat{x}^n = \widehat{J}_{\mathcal{T}^n}^{-1}\left(K_{\mathcal{T}^n}\widehat{x}^{n-1} + C^T R^{-1} y\right) \tag{3.7}$$

We will refer to the recursions defined by equation (3.7) as the *embedded trees* (ET) algorithm [78]. Note that this recursion is similar to a Richardson iteration, as introduced in §2.4.1, except that the preconditioning system $\widehat{J}_{\mathcal{T}^n}$ may change from iteration to iteration. The cost of computing $\widehat{x}^n$ from $\widehat{x}^{n-1}$, as in equation (3.7), is $\mathcal{O}(Nd^3 + Ed^2)$, where $N = |\mathcal{V}|$ is the number of nodes, $E = |\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}^n}|$ is the number of cut edges, and d is the dimension of the hidden variables x_s. Typically E is at most $\mathcal{O}(N)$, and the overall cost of each iteration is $\mathcal{O}(Nd^3)$.

In order to make a clear connection to tree–structured inference algorithms, the preceding discussion has assumed that $\widehat{J}_{\mathcal{T}^n}$ is positive definite. In general, though, it is sufficient to choose $K_{\mathcal{T}^n}$ so that $\widehat{J}_{\mathcal{T}^n}$ is invertible. In such cases, the probabilistic interpretation of equations (3.6, 3.7) is no longer clear. However, the Gaussian BP recursions (2.42, 2.43) will still produce the unique algebraic solution to equation (3.6),

and take only $\mathcal{O}(Nd^3)$ operations to do it. Note that determining whether an arbitrary $K_{\mathcal{T}_n}$ leads to an invertible $\widehat{J}_{\mathcal{T}_n}$ is in general a difficult problem. See §3.4 for a variety of easily checkable sufficient conditions which ensure invertibility.

There is an alternative form of the embedded trees mean recursion (3.7) which will be useful in the subsequent analysis. By taking the difference between the relations implied by equation (3.6) at subsequent iterations, we have

$$\widehat{J}_{\mathcal{T}_n}\widehat{x}^n - \widehat{J}_{\mathcal{T}_{n-1}}\widehat{x}^{n-1} = K_{\mathcal{T}_n}\widehat{x}^{n-1} - K_{\mathcal{T}_{n-1}}\widehat{x}^{n-2} \tag{3.8}$$

Noting from equation (3.1) that $\widehat{J}_{\mathcal{T}_n} - K_{\mathcal{T}_n} = \widehat{J}_{\mathcal{T}_{n-1}} - K_{\mathcal{T}_{n-1}}$, we may rewrite (3.8) as

$$\widehat{x}^n = \widehat{x}^{n-1} + \widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_{n-1}} \left(\widehat{x}^{n-1} - \widehat{x}^{n-2}\right) \tag{3.9}$$

$$\left(\widehat{x}^n - \widehat{x}^{n-1}\right) = \widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_{n-1}} \left(\widehat{x}^{n-1} - \widehat{x}^{n-2}\right) \tag{3.10}$$

where the initial condition $(\widehat{x}^1 - \widehat{x}^0)$ is determined according to equation (3.7). Equation (3.10) explicitly reveals the important fact that the dynamics of the ET algorithm depend solely on the chosen set of cutting matrices $K_{\mathcal{T}_n}$. The observations y act only to set the initial conditions. This data independence allows us to analyze the converge properties of the ET iterations solely in terms of the chosen cutting matrices.

Consider the evolution of the error $e^n \triangleq (\widehat{x}^n - \widehat{x})$ between the estimate $\widehat{x}^n$ at the n^{th} iteration and the solution $\widehat{x}$ of the original inference problem. Using equation (3.3), we may rewrite the ET recursion (3.6) as

$$\widehat{J}_{\mathcal{T}_n}\widehat{x}^n = K_{\mathcal{T}_n}\widehat{x}^{n-1} + \widehat{J}\,\widehat{x} = K_{\mathcal{T}_n}\widehat{x}^{n-1} + \left(\widehat{J}_{\mathcal{T}_n} - K_{\mathcal{T}_n}\right)\widehat{x} \tag{3.11}$$

This equation may be rewritten to relate the errors at subsequent iterations:

$$e^n = \widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} e^{n-1} \tag{3.12}$$

Together, equations (3.12) and (3.10) lead to the following result.

Proposition 3.1. For any starting point $\widehat{x}^0$, the conditional mean $\widehat{x}$ of the original inference problem (3.3) is the unique fixed point of the iterates $\{\widehat{x}^n\}_{n=1}^{\infty}$ generated by the embedded trees recursion (3.7). The error $e^n \triangleq (\widehat{x}^n - \widehat{x})$ and first difference $d^n \triangleq (\widehat{x}^n - \widehat{x}^{n-1})$ evolve according to

$$e^n = \left[\prod_{j=1}^{n} \widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right] e^0 \tag{3.13}$$

$$d^n = \left[\prod_{j=2}^{n} \widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_{j-1}}\right] d^1 \tag{3.14}$$

Proof. The uniqueness of the fixed point $\widehat{x}$ follows directly from the invertibility of $\widehat{J}$ and $\widehat{J}_{\mathcal{T}_n} = \widehat{J} + K_{\mathcal{T}_n}$. Equations (3.13) and (3.14) may be derived by induction using equations (3.12) and (3.10), respectively. □

While Proposition 3.1 shows that the ET recursion has a unique fixed point at the optimal solution $\widehat{x}$, it does not guarantee that $\widehat{x}^n$ will converge to that fixed point. In fact, if the cutting matrices $K_{\mathcal{T}_n}$ are poorly chosen, $\widehat{x}^n$ may diverge from $\widehat{x}$ at a geometric rate (see §3.2.3 for an example).

3.2.2 Embedded Trees as a Richardson Iteration

In this section, we examine the connections between the embedded trees algorithm and the Richardson iterations introduced in §2.4.1. In the process, several standard linear algebraic results are used to analyze the convergence of the ET iterations. Throughout this section, we assume that the cutting matrices $K_{\mathcal{T}_n}$ are chosen so that $\widehat{J}_{\mathcal{T}_n}$ is invertible, ensuring that each iterate may be unambiguously calculated using equation (3.7).

One natural implementation of the embedded trees algorithm cycles through a fixed set of T spanning trees $\{\mathcal{G}_{\mathcal{T}_n}\}_{n=1}^{T}$ in a periodic order, so that

$$\mathcal{G}_{\mathcal{T}_{n+kT}} = \mathcal{G}_{\mathcal{T}_n} \qquad k \in \mathbb{Z}^+ \tag{3.15}$$

In this case, both e^n and d^n evolve as linear periodically varying systems whose convergence can by analyzed as follows:

Theorem 3.2. Suppose the embedded trees mean recursion (3.7) is implemented by periodically cycling through T embedded trees, as in equation (3.15). Then the error $e^n \triangleq \widehat{x}^n - \widehat{x}$ and first difference $d^n \triangleq \widehat{x}^n - \widehat{x}^{n-1}$ evolve according to

$$e^{Tn+T} = \left[\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right] e^{Tn} \triangleq \mathbf{E} e^{Tn} \tag{3.16}$$

$$d^{Tn+T+1} = \left[\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_{j+1}}^{-1} K_{\mathcal{T}_j}\right] d^{Tn+1} \triangleq \mathbf{D} d^{Tn+1} \tag{3.17}$$

where $\widehat{J}_{\mathcal{T}_{T+1}} = \widehat{J}_{\mathcal{T}_1}$, and the matrices $\mathbf{E}$ and $\mathbf{D}$ have the same eigenvalues. If $\rho(\mathbf{E}) < 1$, then for arbitrary $\widehat{x}^0$, $e^n \overset{n\to\infty}{\longrightarrow} 0$ at an asymptotic rate of at most $\gamma \triangleq \rho(\mathbf{E})^{\frac{1}{T}}$. Alternatively, if $\rho(\mathbf{E}) > 1$, then $|e^n| \overset{n\to\infty}{\longrightarrow} \infty$ for almost all $\widehat{x}^0$.

Proof. See Appendix C.1. □

From this theorem, it follows that the convergence rate of the ET algorithm may be optimized by choosing the cutting matrices $K_{\mathcal{T}_n}$ such that the spectral radius $\rho(\mathbf{E})$ is as small as possible.

When the ET iteration (3.7) uses the same cutting matrix $K_{\mathcal{T}}$ at every iteration, it is clearly exactly equivalent to the preconditioned Richardson iteration generated by the matrix splitting $\widehat{J} = \widehat{J}_{\mathcal{T}} - K_{\mathcal{T}}$ (see equation (2.51)). The following theorem shows that when the recursion is implemented by periodically cycling through $T > 1$ cutting matrices, we may still recover a stationary Richardson iteration by considering every T^{th} iterate.

Theorem 3.3. Suppose that the ET recursion (3.7) is implemented by periodically cycling through T cutting matrices $\{K_{\mathcal{T}_n}\}_{n=1}^{T}$. Consider the subsampled sequence of estimates $\{\widehat{x}^{nT}\}_{n=0}^{\infty}$ produced at every T^{th} iteration. The ET procedure generating these iterates is exactly equivalent to a preconditioned Richardson iteration

$$\widehat{x}^{Tn} = \left(I - M_T^{-1}\widehat{J}\right)\widehat{x}^{T(n-1)} + M_T^{-1} C^T R^{-1} y \tag{3.18}$$

where the preconditioner M_T^{-1} is defined according to the recursion

$$M_n^{-1} = \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n} M_{n-1}^{-1} + \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} \tag{3.19}$$

with initial condition $M_1^{-1} = \left(\widehat{J} + K_{\mathcal{T}_1}\right)^{-1}$.

Proof. We prove this theorem by induction. When $T = 1$, this result is easily seen to hold by comparison of equations (2.51) and (3.7). Now suppose it holds for $T = n-1$, so that

$$\widehat{x}^{n-1} = \left(I - M_{n-1}^{-1}\widehat{J}\right)\widehat{x}^0 + M_{n-1}^{-1} C^T R^{-1} y$$

Then, applying equation (3.7), the n^{th} ET iterate is given by

$$\begin{aligned}
\widehat{x}^n &= \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} \left(K_{\mathcal{T}_n}\widehat{x}^{n-1} + C^T R^{-1} y\right) \\
&= \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n}\widehat{x}^0 - \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n} M_{n-1}^{-1}\widehat{J}\widehat{x}^0 + \cdots \\
&\qquad \cdots + \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n} M_{n-1}^{-1} C^T R^{-1} y + \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} C^T R^{-1} y \\
&= \widehat{x}^0 - \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1}\widehat{J}\widehat{x}^0 - \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n} M_{n-1}^{-1}\widehat{J}\widehat{x}^0 + \cdots \\
&\qquad \cdots + \left(\left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n} M_{n-1}^{-1} + \left(\widehat{J} + K_{\mathcal{T}_n}\right)^{-1}\right) C^T R^{-1} y \\
&= \left(I - M_n^{-1}\widehat{J}\right)\widehat{x}^0 + M_n^{-1} C^T R^{-1} y
\end{aligned}$$

where M_n^{-1} and M_{n-1}^{-1} are related as in equation (3.19). □

Connections between the ET algorithm and other Richardson iterations are further discussed in §3.2.5.

3.2.3 Motivating Examples

In this section, we provide some examples which serve several purposes. First, they demonstrate that it is typically quite simple to select a set of spanning trees, and a corresponding set of cutting matrices, that lead to a convergent ET iteration. In addition, they show that if the trees are appropriately chosen, the resulting convergence rate compares quite favorably with alternative inference methods. Finally,

they demonstrate a variety of interesting qualitative features which motivate the subsequent sections of this chapter. A more extensive set of simulation results, which evaluate the ET algorithm's performance on a wider range of problems, can be found in §3.5.

Single Cycle Graphs

We begin by considering a very simple graph whose edges form a single 20–node cycle. Although in practice such graphs can be easily solved by direct methods [53], they provide a useful starting point for understanding the behavior of iterative methods. To construct the inverse prior covariance J, for each edge (s,t) we assign a randomly generated clique potential $\psi_{s,t}(x_s, x_t)$ as in equation (2.15):

$$\begin{aligned} \psi_{s,t}(x_s, x_t) &= \exp\left\{ -\frac{1}{2} \begin{bmatrix} x_s & x_t \end{bmatrix} \begin{bmatrix} w_{st} & a_{st} w_{st} \\ a_{st} w_{st} & w_{st} \end{bmatrix} \begin{bmatrix} x_s \\ x_t \end{bmatrix} \right\} \\ &= \exp\left\{ -\frac{1}{2} w_{st} (x_s + a_{st} x_t)^2 \right\} \end{aligned} \tag{3.20}$$

For each edge, w_{st} is sampled from an exponential distribution with mean 1, while a_{st} is set to $+1$ or -1 with equal probability.[1] We then associate a measurement $y_s = x_s + v_s$, $v_s \sim \mathcal{N}(0,1)$, with each node, so that the inverse error covariance matrix is given by $\widehat{J} = J + I$. The example presented in this section is for a single realization from this distribution of prior models. However, the characteristics we discuss have been consistently observed across many empirical trials, and also hold for other prior model distributions.

To reveal a spanning tree of a single cycle graph, the cutting matrix $K_{\mathcal{T}}$ must only remove a single edge. In this section, we restrict ourselves to regular cutting matrices, so that all off–diagonal of entries of $K_{\mathcal{T}}$ will be zero except for the pair of non–zero entries fixed by equation (3.2). However, we are still free to choose the two diagonal entries $(K_{\mathcal{T}})_{s,s}, (K_{\mathcal{T}})_{t,t}$ corresponding to the nodes from which the single edge (s,t) is cut. We consider three different methods for choosing these values:

[1] By construction, $J_{s,s} = \sum_{t \in N(s)} |J_{s,t}|$ for all $s \in \mathcal{V}$, ensuring that J is positive semidefinite.

Positive Semidefinite By setting $(K_{\mathcal{T}})_{s,s} = (K_{\mathcal{T}})_{t,t} = |(K_{\mathcal{T}})_{s,t}|$, we guarantee that $K_{\mathcal{T}}$ is positive semidefinite. This ensures that $\widehat{J}_{\mathcal{T}} = \widehat{J} + K_{\mathcal{T}}$ is positive definite, allowing the probabilistic interpretation of the ET iteration discussed in §3.2.1.

Zero Diagonal If the diagonal of $K_{\mathcal{T}}$ is set to zero, $\widehat{J}_{\mathcal{T}}$ and $\widehat{J}$ will have the same diagonal values. If the cut edge is sufficiently weak, the resulting $\widehat{J}_{\mathcal{T}}$ should well approximate $\widehat{J}$.

Negative Semidefinite By setting $(K_{\mathcal{T}})_{s,s} = (K_{\mathcal{T}})_{t,t} = -|(K_{\mathcal{T}})_{s,t}|$, the cutting matrix acts to entirely remove the clique potential $\psi_{s,t}(x_s, x_t)$ from the graph.

More sophisticated choices of the diagonal entries are discussed in §4.1.

We begin by examining the convergence rate of the ET iteration (3.7) when the same cutting matrix $K_{\mathcal{T}}$ is used at every iteration. In this case, Theorem 3.2 shows that this convergence rate is given by $\gamma = \rho(\mathbf{E})$, where $\mathbf{E} = \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}$. Figure 3-2(a) plots γ as a function of the magnitude $|\widehat{J}_{s,t}|$ of the off–diagonal error covariance entry corresponding to the cut edge. Intuitively, convergence is fastest when the magnitude of the cut edge is small. In addition, we see that the zero diagonal cutting matrices always lead to the fastest convergence rates. Notice that for sufficiently strong edges, negative definite cutting matrices lead to a divergent iteration. Thus, although cutting edges by removing clique potentials makes sense from a probabilistic perspective, the ET algorithm appears to perform best when cutting matrices are chosen so that the overall inverse error covariance matrix $\widehat{J}$ is perturbed as little as possible.

More interesting behavior is obtained when the ET algorithm is implemented by periodically cycling between two cutting matrices $K_{\mathcal{T}_1}, K_{\mathcal{T}_2}$. Theorem 3.2 then shows that the convergence rate is given by $\gamma = \rho(\mathbf{E})^{1/2}$, where $\mathbf{E} = \widehat{J}_{\mathcal{T}_2}^{-1} K_{\mathcal{T}_2} \widehat{J}_{\mathcal{T}_1}^{-1} K_{\mathcal{T}_1}$. Figures 3-2(b,c) plot these convergence rates when $K_{\mathcal{T}_1}$ is chosen to cut the cycle's weakest edge, and $K_{\mathcal{T}_2}$ is varied over all other edges. When plotted against the magnitude of the second edge cut, as in Figure 3-2(b), the γ values display little structure. Figure 3-2(c) shows these same γ values plotted against an index number showing the ordering of edges in the cycle. Edge 7 is the weak edge removed by $K_{\mathcal{T}_1}$. Notice that for the zero diagonal case, cutting the same edge at every iteration

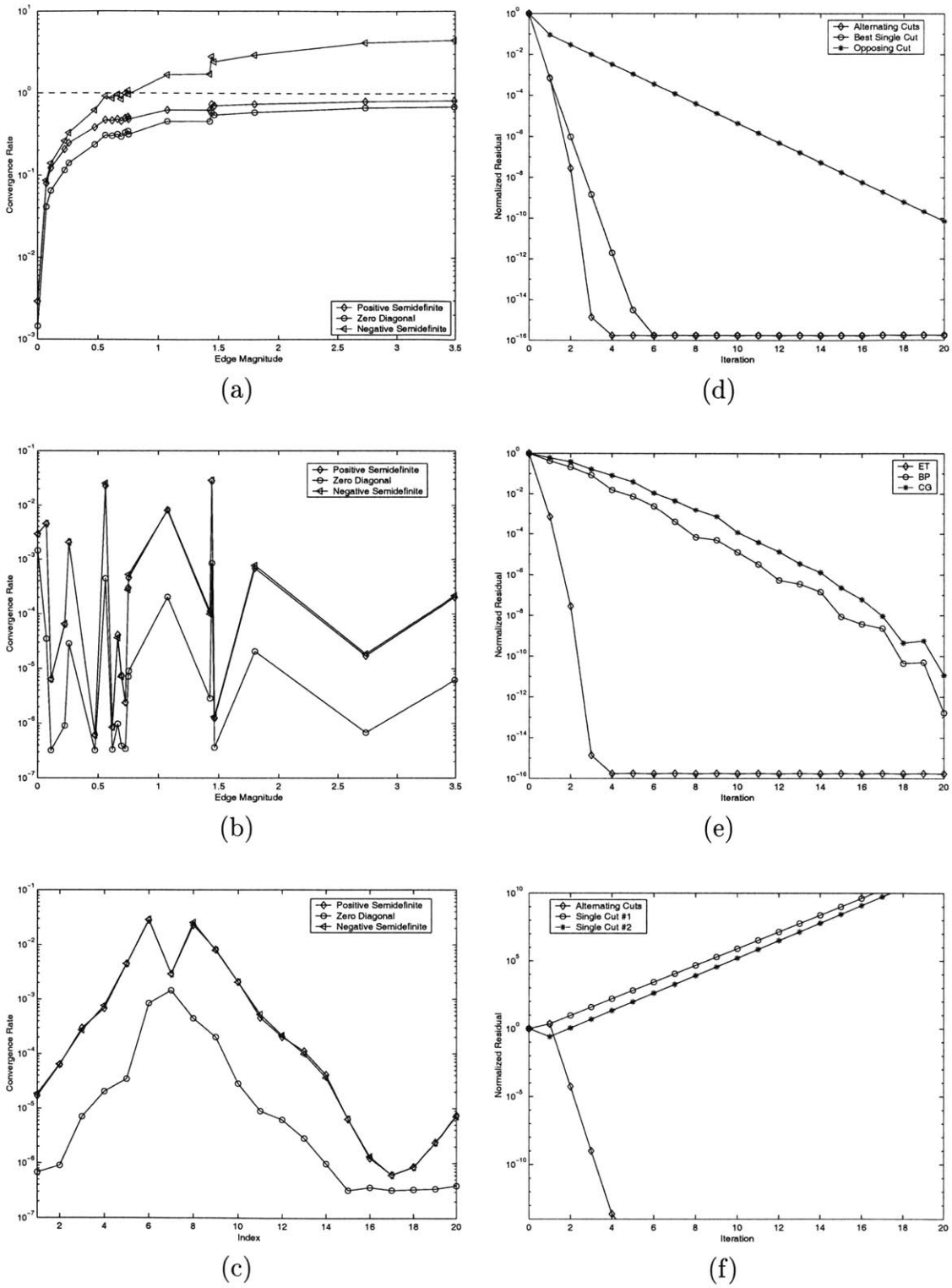


Figure 3-2: Behavior of the ET algorithm on a single 20–node cycle. (a) Single–tree convergence rate versus edge strength. (b) Two–tree convergence rate versus edge strength. (c) Two–tree convergence rate versus edge index. (d) Comparison of zero diagonal single–tree and two–tree iterations. (e) Comparison to belief propagation (BP) and conjugate gradient (CG). (f) Two individually divergent cuts produce a convergent two–tree iteration.

is the *worst* possible choice, despite the fact that every other edge in the graph is stronger and leads to slower single–tree iterations. The best performance is obtained by choosing the second cut edge to be as far from the first edge as possible.

In Figures 3-2(d,e), we examine the convergence behavior of the zero diagonal two–tree iteration corresponding to edges 7 and 19 in more detail. For both figures, the error at each iteration is measured using the normalized residual introduced in §2.4.3. Figure 3-2(d) shows that even though the single–tree iteration generated by edge 19 converges rather slowly relative to the edge 7 iteration, the composite iteration is orders of magnitude faster. In Figure 3-2(e), we compare the performance of the parallel belief propagation (BP) and unpreconditioned conjugate gradient (CG) iterations, showing that for this problem the ET algorithm is much faster.

The previous plots suggest that a two–tree ET iteration may exhibit features quite different from those observed in the corresponding single–tree iterations. This is dramatically demonstrated by Figure 3-2(f), which considers the negative definite cutting matrices corresponding to the two *strongest* edges in the graph. As predicted by Figure 3-2(a), the single–tree iterations corresponding to these edges are divergent. However, because these strong edges are widely separated in the original graph (indexes 1 and 12), they lead to a two–tree iteration which outperforms even the *best* single–tree iteration.

Nearest–Neighbor Grids

In this section, we apply the ET algorithm to a 25×25 nearest–neighbor grid, analogous to the 5×5 grid in Figure 3-1. The prior model J was created by randomly generating clique potentials exactly as in the previous example, and we again associate measurements of uniform quality with each node so that $\widehat{J} = J + I$. From the previous example, we expect the ET iteration to perform best when implemented with multiple trees cutting widely separated edges. Therefore, we consider two different choices of spanning trees corresponding to the first two cutting matrices of Figure 3-1. For the presented results, we set the diagonal entries of the cutting matrices to zero. We also examined the positive semidefinite and negative semidefinite conditions

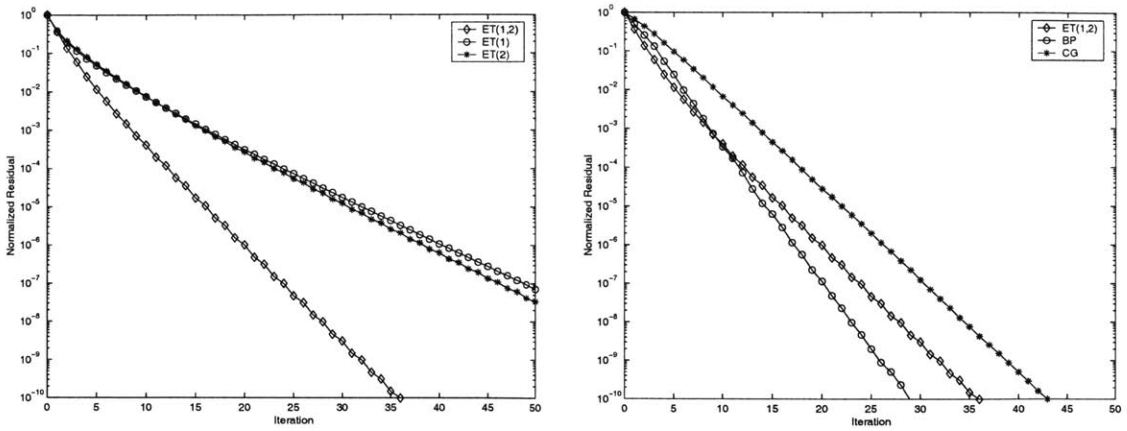


Figure 3-3: Convergence of the ET algorithm on a 25×25 nearest–neighbor grid. (a) Two single–tree iterations (ET(1) and ET(2)) are compared to the corresponding two–tree iteration (ET(1,2)). (b) Comparison to the belief propagation (BP) and conjugate gradient (CG) methods.

introduced in the last section, but found that they again led to inferior performance.

Figure 3-3(a) compares the performance of the two single–tree ET iterations (denoted by ET(1) and ET(2)) to the two–tree iteration (denoted by ET(1,2)). We see that using multiple trees again leads to dramatic improvements. Figure 3-3(b) shows that for this problem, the ET algorithm converges faster than CG, but at an asymptotically slower rate than BP. Thus, the ET algorithm can lead to competitive performance even when large numbers of edges must be removed to reveal spanning trees. The performance of the ET algorithm on nearest–neighbor grids is examined in more detail in §3.5.3.

3.2.4 Geometric Analysis

In the Introduction, we saw that many interesting classes of graphical models are "nearly" tree–structured, in the sense that only a small number of edges must be removed to reveal an embedded spanning tree. As discussed in §3.1, the rank of the cutting matrices $K_{\mathcal{T}}$ associated with such spanning trees is small compared to the total number of nodes in the graph. Motivated by this fact, in this section we demonstrate that low–rank cutting matrices place a set of geometric constraints on the dynamics of the embedded trees recursions. In particular, we show that the errors

$e^n = (\widehat{x}^n - \widehat{x})$ are constrained to lie in a particular subspace of dimension $\text{rank}(K_{\mathcal{T}})$.

Theorem 3.2 demonstrates that, for a given set of cutting matrices $\{K_{\mathcal{T}_n}\}_{n=1}^T$, the ET error dynamics are determined by the $\widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n}$ matrices featured in equation (3.16). From basic linear algebra, we know that $\text{rank}(\widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n}) \leq \text{rank}(K_{\mathcal{T}_n})$. The following lemma provides a decomposition which explicitly reveals the important role played by the eigenstructure of $K_{\mathcal{T}_n}$.

Lemma 3.4. Given two inverse covariance matrices $\widehat{J}$ and $\widehat{J}_{\mathcal{T}_n}$ of dimension Nd, let $K_{\mathcal{T}_n} = (\widehat{J}_{\mathcal{T}_n} - \widehat{J})$ be the associated symmetric cutting matrix. Defining $r \triangleq \text{rank}(K_{\mathcal{T}_n})$, the reduced–rank diagonalization of $K_{\mathcal{T}_n}$ is given by $K_{\mathcal{T}_n} = U_n D_n U_n^T$, where $D_n \in \mathbb{R}^{r \times r}$ is a diagonal matrix containing the nonzero eigenvalues of $K_{\mathcal{T}_n}$ and the columns of $U_n \in \mathbb{R}^{Nd \times r}$ are the corresponding eigenvectors ($U_n^T U_n = I_r$). We then have

$$\widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} = \left(\widehat{P}^{-1} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n} = \widehat{P} U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n\right)^{-1} U_n^T \tag{3.21}$$

Proof. See Appendix C.2. □

Suppose that the ET algorithm is implemented using a single cutting matrix $K_{\mathcal{T}_1}$. Lemma 3.4 then shows that at each iteration, the error vector e^n must reside within the subspace spanned by U_1, the eigenvectors associated with the nonzero eigenvalues of $K_{\mathcal{T}_1}$. When multiple trees are used, the specific subspace in which the error resides is always defined by the most recently applied cutting matrix, and will therefore change from iteration to iteration. However, because each cutting matrix removes the same number of edges to reveal a spanning tree, the dimension of these subspaces will always be $\mathcal{O}(Ed)$.

In §3.2.3, we showed that there can be important interactions between the different spanning trees chosen for a particular periodic implementation of the ET algorithm. The following proposition examines the relationship between different orderings of a given, fixed set of cutting matrices.

Proposition 3.5. Let $\{K_{\mathcal{T}_n}\}_{n=1}^{T}$ be a fixed set of T cutting matrices. Consider the family of circular permutation functions $\sigma_k(n)$ defined by

$$\sigma_k(n) = 1 + ((n + k - 1) \bmod T) \qquad k = 0, 1, \ldots, T-1 \tag{3.22}$$

Then any such circular permutation leaves the eigenvalues of the ET error dynamics matrix $\mathbf{E}$, as defined by equation (3.16), unchanged:

$$\left\{\lambda_i\left(\prod_{n=1}^{T} \widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n}\right)\right\} = \left\{\lambda_i\left(\prod_{n=1}^{T} \widehat{J}_{\mathcal{T}_{\sigma_k(n)}}^{-1} K_{\mathcal{T}_{\sigma_k(n)}}\right)\right\} \tag{3.23}$$

In addition, the reversal of any such permutation retains the same eigenvalues:

$$\begin{aligned}&\left\{\lambda_i\left(\widehat{J}_{\mathcal{T}_{\sigma_k(1)}}^{-1} K_{\mathcal{T}_{\sigma_k(1)}} \widehat{J}_{\mathcal{T}_{\sigma_k(2)}}^{-1} K_{\mathcal{T}_{\sigma_k(2)}} \cdots \widehat{J}_{\mathcal{T}_{\sigma_k(T)}}^{-1} K_{\mathcal{T}_{\sigma_k(T)}}\right)\right\} \\ &\qquad = \left\{\lambda_i\left(\widehat{J}_{\mathcal{T}_{\sigma_k(T)}}^{-1} K_{\mathcal{T}_{\sigma_k(T)}} \widehat{J}_{\mathcal{T}_{\sigma_k(T-1)}}^{-1} K_{\mathcal{T}_{\sigma_k(T-1)}} \cdots \widehat{J}_{\mathcal{T}_{\sigma_k(1)}}^{-1} K_{\mathcal{T}_{\sigma_k(1)}}\right)\right\}\end{aligned} \tag{3.24}$$

Proof. Equation (3.23) can be easily shown be repeated application of the eigenvalue identity $\{\lambda_i(AB)\} = \{\lambda_i(BA)\}$. To show equation (3.24), we note from Lemma 3.4 that $\widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} = \widehat{P} G_{\mathcal{T}_n}$ for an appropriately chosen symmetric matrix $G_{\mathcal{T}_n}$. Then, applying the same eigenvalue identities used in the proof of Theorem 3.2, we have

$$\begin{aligned}&\left\{\lambda_i\left(\widehat{P} G_{\mathcal{T}_T} \cdots \widehat{P} G_{\mathcal{T}_2} \widehat{P} G_{\mathcal{T}_1}\right)\right\} = \left\{\lambda_i\left(\left(\widehat{P} G_{\mathcal{T}_T} \cdots \widehat{P} G_{\mathcal{T}_2} \widehat{P}\right)^T G_{\mathcal{T}_1}\right)\right\} \\ &\quad = \left\{\lambda_i\left(\widehat{P} G_{\mathcal{T}_2} \widehat{P} G_{\mathcal{T}_3} \cdots \widehat{P} G_{\mathcal{T}_T} \widehat{P} G_{\mathcal{T}_1}\right)\right\} = \left\{\lambda_i\left(\widehat{P} G_{\mathcal{T}_1} \widehat{P} G_{\mathcal{T}_2} \cdots \widehat{P} G_{\mathcal{T}_{T-1}} \widehat{P} G_{\mathcal{T}_T}\right)\right\}\end{aligned}$$

Equation (3.24) then follows from equation (3.23). □

Note that this proposition shows that all permutations of at most three cutting matrices will lead to ET recursions with completely equivalent dynamical properties. More generally, the set of all permutations of $T \geq 3$ cutting matrices may be divided into $(T-1)!/2$ classes such that each of the $2T$ permutations within each class are spectrally equivalent.

Proposition 3.5 shows that the eigenvalues governing the ET algorithm's dynamics

are invariant to permutations that leave the set of adjacent pairs of cutting matrices $\{(K_{\mathcal{T}_n}, K_{\mathcal{T}_{n+1}})\}$ unchanged. The following theorem provides a decomposition of the ET error dynamics matrix which explicitly shows the mechanism by which adjacent cutting matrices interact.

Theorem 3.6. Let $\{K_{\mathcal{T}_n}\}_{n=1}^T$ be a set of T cutting matrices, each with a reduced–rank eigendecomposition $K_{\mathcal{T}_n} = U_n D_n U_n^T$ as defined in Lemma 3.4. Then the nonzero eigenvalues of the ET error dynamics matrix $\mathbf{E}$, as defined by equation (3.16), are equal to the eigenvalues of the reduced–dimension matrix $\mathbf{E}_r$ defined by

$$\mathbf{E}_r = \prod_{n=1}^{T} U_{n+1}^T \widehat{P} U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n \right)^{-1} \tag{3.25}$$

where $U_{T+1} \triangleq U_1$.

Proof. This equation follows immediately if we use equation (3.21) to rewrite equation (3.16), and then use the identity $\{\lambda_i(AB)\} = \{\lambda_i(BA)\}$ to switch the position of the U_1^T term. □

Note that the dimension of $\mathbf{E}_r$ is equal to the rank of $K_{\mathcal{T}_1}$. Thus, when combined with Proposition 3.5, equation (3.25) demonstrates that the dynamics of the errors e^n may be expressed by a matrix of dimension $r \triangleq \min_{1 \le n \le T} \operatorname{rank}(K_{\mathcal{T}_n})$. This is possible because e^n is constrained to lie within an r–dimensional subspace of $\mathbb{R}^{Nd}$ at least once during each period of the ET recursions.

3.2.5 Connections to Previous Work

As discussed in §3.2.2, in certain cases the embedded trees recursion can be viewed as a Richardson iteration. Consequently, although the ET iteration is novel, it has a number of interesting connections to other algorithms in the numerical linear algebra literature. In this section, we provide a brief introduction to the most relevant related techniques.

Single Tree Splittings

The discussion in §3.1 focused on spanning tree approximations of graphs with cycles because they lead to the smallest number of cut edges, and therefore hopefully the best approximations. However, the BP inference algorithm may be used to solve a problem defined on a disconnected set of trees, known as a *forest*, just as easily. The simplest such forest is the graph in which every edge is cut, so that the maximal cliques are single nodes. The ET algorithm, when applied to this trivial spanning forest, is exactly equivalent to the classic Gauss–Jacobi iteration presented in §2.4.1.

Although more general tree–structured graph splittings are not as common, they have been investigated by a few authors. Sjogren [68] defines a tree splitting, and uses the resulting Richardson iteration to solve non–negative linear systems arising in the analysis of Markov chains. However, his analysis uses many of the special properties of such non–negative systems, and is not directly applicable to the positive definite systems we consider. He never considers an alternation between different tree splittings. More recently, several authors have investigated the use of a single tree splitting as a preconditioner for the conjugate gradient iteration [8, 16, 17, 19]. This work is discussed in more detail in §4.3.3.

Parallel Multisplittings

The ET algorithm differs from standard Richardson iterations because it may employ a different matrix splitting at each iteration. O'Leary and White [54] also employ multiple matrix splittings to form a single composite iteration. However, they combine their splittings in a parallel rather than a sequential fashion. In particular, given a set of T splittings $\widehat{J} = M_t - K_t$, $t = 1, \ldots, T$, they generate a sequence of iterates $\widehat{x}^n$ according to

$$\widehat{x}^n = \sum_{t=1}^{T} D_t M_t^{-1} K_t \widehat{x}^{n-1} + \sum_{t=1}^{T} D_t M_t^{-1} C^T R^{-1} y \tag{3.26}$$

where D_t are diagonal matrices chosen so that $\sum_t D_t = I$. This iteration is motivated by the fact that each of the splittings may be independently applied at each iteration, allowing easy parallelization.

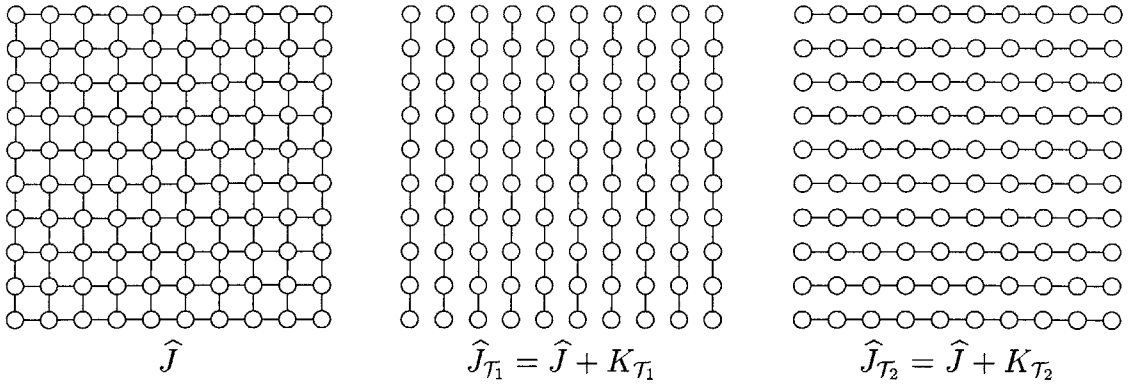


Figure 3-4: Spanning forests employed by the ADI method. $K_{\mathcal{T}_1}$ is chosen to remove all horizontal edges, while $K_{\mathcal{T}_2}$ removes vertical edges.

As shown in [81], parallel multi–splittings may often be shown to be equivalent to a single appropriately chosen splitting. This contrasts with the ET algorithm, as the sequential multiple splitting preconditioners derived in Theorem 3.3 are generally *not* equivalent to any single splitting. Also, note that no mention of tree–structured splittings is made in any of the existing parallel multi–splitting literature. More recently, parallel multi–splittings have been used as preconditioners for Krylov subspace methods [40].

Alternating Direction Implicit Methods

The *Alternating Direction Implicit* (ADI) method was originally proposed by Peaceman and Rachford [58] as an iterative technique for the numerical solution of elliptic and parabolic differential equations. They focus exclusively on problems defined on nearest–neighbor grids, as in Figure 3-4. In the first half of each ADI iteration, the horizontal PDE constraints are ignored, allowing the decoupled tridiagonal systems corresponding to the vertical constraints to be easily solved. The second half of the iteration neglects the vertical constraints to solve the horizontal systems. Although the original notation and formulation is somewhat different, the ADI method is easily shown to be completely equivalent to a two–tree embedded trees iteration where the first and second cutting matrices are chosen to remove the horizontal and vertical edges, respectively. The resulting spanning forests are shown in Figure 3-4.

Most extensions of the ADI method have focused on the calculation of acceleration

parameters to optimize the algorithm's convergence rate [73, 75]. Such optimizations are equivalent to modifications of the diagonal entries of the cutting matrices. However, as discussed in [14], these analyses require that the spanning tree distributions $\widehat{J}_{\mathcal{T}_1}$ and $\widehat{J}_{\mathcal{T}_2}$ share a common set of eigenvectors, a condition which only holds for an extremely limited set of linear systems $\widehat{J}$. Thus, these techniques are not applicable to the irregular, inhomogeneous MRFs which motivated the ET algorithm. It does not appear that anyone has yet considered extending the ADI iteration to more general graphs and spanning trees, as proposed in this thesis.

Practical implementations of the ADI method typically employ it not as a Richardson iteration, but as a preconditioner for a Krylov subspace method [7, 74]. Although very good empirical performance has often been observed, the theory is mostly limited to homogeneous problems. For more information and references concerning ADI methods, see [7, 85].

3.3 Iterative Calculation of Error Covariances

In the previous section, we developed an iterative algorithm to calculate the conditional mean of a Gaussian inference problem defined on a graph with cycles. However, as discussed in §2.1 and §2.3, we would also like to calculate the error variances $\widehat{P}_s$ of the conditional marginal distributions $p(x_s \mid y) \sim \mathcal{N}\left(\widehat{x}_s, \widehat{P}_s\right)$. Due to the linear algebraic structure underlying Gaussian inference problems, any procedure for calculating $\widehat{x}$ may be easily adapted to the calculation of error variances. In particular, suppose that the full error covariance matrix $\widehat{P}$ is partitioned into columns:

$$\widehat{P} = \begin{bmatrix} | & | & & | \\ \hat{p}_1 & \hat{p}_2 & \cdots & \hat{p}_{Nd} \\ | & | & & | \end{bmatrix} \tag{3.27}$$

Then, letting e_i be an Nd dimensional vector of zeros with a one in the i^{th} position, the i^{th} column of $\widehat{P}$ is given by

$$\begin{aligned} \hat{p}_i &= \widehat{P} e_i \\ \widehat{J}\hat{p}_i &= e_i \end{aligned} \tag{3.28}$$

By comparison to equation (3.3), we see that $\hat{p}_i$ is equal to the conditional mean of a particular inference problem defined by the synthetic observation vector e_i. Thus, given an inference procedure like the ET algorithm which calculates conditional means at a cost of $\mathcal{O}(Nd^3)$ per iteration, we may calculate a series of approximations to $\widehat{P}$ using $\mathcal{O}(N^2d^4)$ operations per iteration.

While the procedure described in the previous paragraph is theoretically sound, the computational cost may be too large for many applications. We would ideally like to find an algorithm which only calculates the N desired marginal error variances $\widehat{P}_s$, avoiding the $\mathcal{O}(N^2)$ cost which any algorithm calculating all of $\widehat{P}$ must require. In this section, we show that for graphs which are sufficiently sparse, one can do much better. In particular, if E edges must be removed to reveal an embedded spanning tree, $\{\widehat{P}_s\}_{s\in\mathcal{V}}$ may be calculated using only $\mathcal{O}(NEd^4)$ operations per iteration. As the algorithm depends on the relationship between cutting matrix rank and the number of cut edges introduced in §3.1, we begin by presenting techniques for explicitly constructing rank–revealing decompositions of cutting matrices. Numerical examples demonstrating the performance of the proposed method are given in §3.5.

Throughout the following sections, all cutting matrices are assumed to be regular (see §3.1). In addition, in all cases where cutting matrices are used to implement the ET mean recursion, we assume that the resulting iteration converges for all observation vectors y. Finally, for notational simplicity we will always assume that the initial condition of the ET mean recursion is $\widehat{x}^0 = 0$.

3.3.1 Low–Rank Decompositions of Cutting Matrices

For any regular cutting matrix $K_{\mathcal{T}}$, there exist a wide variety of additive decompositions into rank–one terms:

$$K_{\mathcal{T}} = \sum_i \omega_i u_i u_i^T \qquad\qquad u_i \in \mathbb{R}^{Nd} \tag{3.29}$$

For example, because any symmetric matrix has an orthogonal set of eigenvectors, the eigendecomposition $K_{\mathcal{T}} = UDU^T$ introduced in §3.2.4 is of this form. Often, however, it may not be tractable to calculate this decomposition, since direct eigenanalysis algorithms require $\mathcal{O}(N^3 d^3)$ operations. In this section, we demonstrate that it is always possible to construct such a decomposition using only $\mathcal{O}(Ed)$ vectors u_i. The cost of the explicit construction procedure is at most $\mathcal{O}(Ed^3)$.

Consider a cutting matrix $K_{\mathcal{T}}$ that acts to remove the edges $\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}$ from the graph, and let $E = |\mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}|$. Our decomposition starts by associating the cut edges with local interaction matrices, analogous to the pairwise clique potentials introduced in §2.2.3. Let $K_{s,t}$ denote the $(s,t)^{th}$ block entry of $K_{\mathcal{T}}$. For each $(s,t) \in \mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}$, we define a local cut matrix $\kappa_{s,t}$ given by

$$\kappa_{s,t} = \begin{bmatrix} K_{s(t)} & K_{s,t} \\ K_{t,s} & K_{t(s)} \end{bmatrix} \tag{3.30}$$

where the $K_{s(t)}$ terms are chosen so that for all $s \in \mathcal{V}$,

$$\sum_{(s,t) \in \mathcal{E} \setminus \mathcal{E}_{\mathcal{T}}} K_{s(t)} = K_{s,s} \tag{3.31}$$

Note that the dimension of $\kappa_{s,t}$ is $2d \times 2d$. We may compute an eigendecomposition of $\kappa_{s,t}$ in $\mathcal{O}(d^3)$ operations, which can be written as

$$\kappa_{s,t} = \begin{bmatrix} U_{s(t)} \\ U_{t(s)} \end{bmatrix} \Omega_{s,t} \begin{bmatrix} U_{s(t)}^T & U_{t(s)}^T \end{bmatrix} \tag{3.32}$$

This decomposition of $\kappa_{s,t}$ also provides a partial additive decomposition of the cutting matrix. Letting $\{\omega_i\}_{i=1}^{2d}$ denote the eigenvalues defining the diagonal of $\Omega_{s,t}$, we have

$$\begin{bmatrix} \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & K_{s(t)} & \mathbf{0} & K_{s,t} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & K_{t,s} & \mathbf{0} & K_{t(s)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots \end{bmatrix} = \sum_{i=1}^{2d} \omega_i u_i u_i^T \qquad \begin{bmatrix} | & & | \\ u_1 & \cdots & u_{2d} \\ | & & | \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ U_{s(t)} \\ \mathbf{0} \\ U_{t(s)} \\ \mathbf{0} \end{bmatrix} \tag{3.33}$$

where the zero blocks $\mathbf{0}$ are chosen so that the blocks of $\kappa_{s,t}$ are aligned with the node ordering used to define $K_{\mathcal{T}}$. By repeating this procedure for each cut edge, we may obtain an additive decomposition of the form defined by equation (3.29) which uses a total of $2Ed$ vectors. The cost of constructing this decomposition is $\mathcal{O}(Ed^3)$. The following example illustrates the decomposition procedure in more detail.

Example 3.7. Consider the five–node undirected graphical model, and corresponding spanning tree, shown in Figure 3-5. Suppose that the inverse error covariance matrix $\widehat{J}$, and desired cutting matrix $K_{\mathcal{T}}$, are given by

$$\underbrace{\begin{bmatrix} 3 & 1 & -2 & 0 & 0 \\ 1 & 2 & 0 & 1 & -2 \\ -2 & 0 & 3 & -1 & 0 \\ 0 & 1 & -1 & 5 & -3 \\ 0 & -2 & 0 & -3 & 4 \end{bmatrix}}_{\widehat{J}} + \underbrace{\begin{bmatrix} 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \end{bmatrix}}_{K_{\mathcal{T}}} = \underbrace{\begin{bmatrix} 3 & 0 & -2 & 0 & 0 \\ 0 & 2 & 0 & 1 & 0 \\ -2 & 0 & 3 & -1 & 0 \\ 0 & 1 & -1 & 5 & -3 \\ 0 & 0 & 0 & -3 & 4 \end{bmatrix}}_{\widehat{J}_{\mathcal{T}}} \tag{3.34}$$

Note that $K_{\mathcal{T}}$ cuts two edges. Since the diagonal entries of $K_{\mathcal{T}}$ are zero, the local cut interaction matrices $\{\kappa_{1,2}, \kappa_{2,5}\}$, and their corresponding eigendecompositions, may

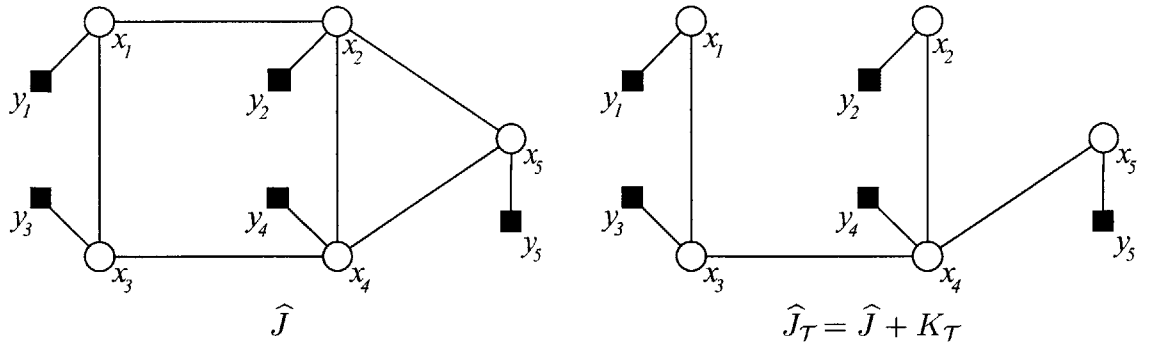


Figure 3-5: An undirected graphical model with inverse error covariance $\widehat{J}$, and the corresponding spanning tree produced by the cutting matrix $K_{\mathcal{T}}$.

be written as

$$\kappa_{1,2} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}^T \tag{3.35}$$

$$\kappa_{2,5} = \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}^T \tag{3.36}$$

Then, using equation (3.33), we see that $K_{\mathcal{T}}$ may be represented as in equation (3.29) using the four rank–one terms given by

$$\begin{bmatrix} | & | & | & | \\ u_1 & u_2 & u_3 & u_4 \\ | & | & | & | \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} \qquad \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 2 \\ -2 \end{bmatrix} \tag{3.37}$$

The preceding construction may be used to decompose an arbitrary cutting matrix into a sum of $2Ed$ rank–one terms. However, if the block diagonal entries of the cutting matrix are chosen so that $\text{rank}(\kappa_{s,t}) = d$, the number of terms may be reduced to only Ed. This reduction may be achieved for scalar hidden variables by simply setting $K_{s(t)} = K_{t(s)} = |K_{s,t}|$, as illustrated below:

Example 3.8. Consider the same graphical model, inverse error covariance $\widehat{J}$, and cut edges used in Example 3.7. We begin by modifying the local cut interaction matrices from equations (3.35, 3.36) so that they have rank one:

$$\kappa_{1,2} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} = 2 \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \tag{3.38}$$

$$\kappa_{2,5} = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix} = 4 \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \tag{3.39}$$

Then, again using equation (3.33), we may transform these local decompositions into a two–vector representation of the cutting matrix:

$$\begin{bmatrix} | & | \\ u_1 & u_2 \\ | & | \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} \tag{3.40}$$

From equation (3.29), it is straightforward to show that these vectors lead to the following alternative spanning tree distribution $\widehat{J}_{\mathcal{T}}$:

$$\underbrace{\begin{bmatrix} 3 & 1 & -2 & 0 & 0 \\ 1 & 2 & 0 & 1 & -2 \\ -2 & 0 & 3 & -1 & 0 \\ 0 & 1 & -1 & 5 & -3 \\ 0 & -2 & 0 & -3 & 4 \end{bmatrix}}_{\widehat{J}} + \underbrace{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 3 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 2 \end{bmatrix}}_{K_{\mathcal{T}}} = \underbrace{\begin{bmatrix} 4 & 0 & -2 & 0 & 0 \\ 0 & 5 & 0 & 1 & 0 \\ -2 & 0 & 3 & -1 & 0 \\ 0 & 1 & -1 & 5 & -3 \\ 0 & 0 & 0 & -3 & 6 \end{bmatrix}}_{\widehat{J}_{\mathcal{T}}} \tag{3.41}$$

Thus, by modifying the diagonal entries of $K_{\mathcal{T}}$, we have halved the number of terms in the cutting matrix decomposition.

When more than one edge is cut from the same node, there may be interactions which cause $\text{rank}(K_{\mathcal{T}})$ to be less than Ed. In such cases, it may be desirable to

further reduce the number of terms in the additive decomposition (3.29). This can be done by replacing the pairwise interaction matrices (3.30) with matrices combining the effects of several different cut edges. The resulting eigendecompositions, whose cost is cubic in the number of edges simultaneously considered, may sometimes reveal linear dependencies between the pairwise interactions, allowing a reduction in the number of terms. For densely connected graphs such as nearest–neighbor grids, such linear dependencies occur generically, and considering groups of edges may provide noticeable gains. However, for more sparse random fields like those discussed in the Introduction, the number of linear dependencies is typically negligible, and the method in this section is nearly optimal.

3.3.2 Embedded Trees as a Series Expansion

In order to apply the low–rank decompositions constructed in the previous section, we return to the embedded trees mean recursion (3.7) derived in §3.2.1. For simplicity, in this section we focus on the ET iteration generated by a single cutting matrix $K_{\mathcal{T}}$ chosen so that $\rho\left(\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}\right) < 1$. We will return to the multiple tree case in the following section.

When $\widehat{x}^0 = 0$, the subsequent iterations $\widehat{x}^n$ of the ET algorithm may be expressed as a series of linear function of the observation vector y. By repeated application of equation (3.7), it is straightforward to show that these functions are given by

$$\begin{aligned}
\widehat{x}^1 &= \widehat{J}_{\mathcal{T}}^{-1}C^T R^{-1}y \\
\widehat{x}^2 &= \left[\widehat{J}_{\mathcal{T}}^{-1} + \widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}\widehat{J}_{\mathcal{T}}^{-1}\right] C^T R^{-1}y \\
\widehat{x}^3 &= \left[\widehat{J}_{\mathcal{T}}^{-1} + \widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}\widehat{J}_{\mathcal{T}}^{-1} + \widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}\widehat{J}_{\mathcal{T}}^{-1}\right] C^T R^{-1}y \\
&\vdots \\
\widehat{x} &= \widehat{P}C^T R^{-1}y
\end{aligned} \tag{3.42}$$

Using an induction argument similar to that found in the proof of Theorem 3.3, the

linear function defining the n^{th} iterate $\widehat{x}^n$ can be shown to equal

$$\widehat{x}^n = \left[\widehat{J}_{\mathcal{T}}^{-1} + F_n\right] C^T R^{-1} y \tag{3.43}$$

where the matrix F_n satisfies the recursion

$$F_n = \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} \left[\widehat{J}_{\mathcal{T}}^{-1} + F_{n-1}\right] \tag{3.44}$$

with the initial condition $F_1 = \mathbf{0}$. Since we are assuming convergence, which from Theorem 3.2 guarantees that $\widehat{x}^n \stackrel{n\to\infty}{\longrightarrow} \widehat{x}$ for any observation vector y, the sequence of matrices defined by equation (3.42), or equivalently by equations (3.43, 3.44), must converge to $\widehat{P}$:

$$\widehat{P} = \sum_{n=0}^{\infty} \widehat{J}_{\mathcal{T}}^{-1} \left[K_{\mathcal{T}} \widehat{J}_{\mathcal{T}}^{-1}\right]^n = \lim_{n\to\infty} \widehat{J}_{\mathcal{T}}^{-1} + F_n \tag{3.45}$$

In fact, these matrices correspond exactly to the series expansion of $\widehat{P}$ generated by the following fixed–point equation:

$$\begin{aligned} \left(\widehat{P}^{-1} + K_{\mathcal{T}}\right) \widehat{P} &= I + K_{\mathcal{T}} \widehat{P} \\ \widehat{P} &= \widehat{J}_{\mathcal{T}}^{-1} + \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} \widehat{P} \end{aligned} \tag{3.46}$$

The matrix sequence defined by equation (3.45) may be derived by repeatedly using equation (3.46) to expand itself.

Clearly, if we could somehow track the terms generated by the ET series expansion (3.45), we could recover the desired error covariance matrix $\widehat{P}$. Equation (3.43) suggests a natural method for doing this. In particular, since the diagonal entries of $\widehat{J}_{\mathcal{T}}^{-1}$ are calculated by the BP algorithm, we may calculate $\{\widehat{P}_s\}_{s\in\mathcal{V}}$ by tracking the evolution of F_n. Note that equation (3.44) shows that $\mathrm{rank}(F_n) \leq \mathrm{rank}(K_{\mathcal{T}})$. In the following section, we show how a low–rank decomposition of $K_{\mathcal{T}}$, as derived in §3.3.1, may be used to track a similar low–rank decomposition of F_n. The computational cost of such a direct tracking procedure is $\mathcal{O}(NE^2d^5)$ operations per iteration. Then, in §3.3.4 we discuss a related, but not entirely equivalent, algorithm which

reduces the cost to $\mathcal{O}(NEd^4)$ operations per iteration by exploiting the fixed point equation (3.46).

3.3.3 A Direct Tracking Approach to Error Covariances

In this section, we provide a direct method for tracking the sequence of linear functions implicitly generated by the ET mean recursion (3.7). We begin by extending the series expansion equations derived in the previous section to cases where the cutting matrix is allowed to vary from iteration to iteration. In general, the n^{th} iterate $\widehat{x}^n$ is given by the following analogs of equations (3.43, 3.44):

$$\widehat{x}^n = \left[\widehat{J}_{\mathcal{T}_n}^{-1} + F_n\right] C^T R^{-1} y \tag{3.47}$$

$$F_n = \widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} \left[\widehat{J}_{\mathcal{T}_{n-1}}^{-1} + F_{n-1}\right] \qquad F_1 = \mathbf{0} \tag{3.48}$$

Because we have assumed the ET iteration is convergent, Proposition 3.1 ensures that

$$\lim_{n\to\infty} \left(\widehat{J}_{\mathcal{T}_n}^{-1} + F_n\right) = \widehat{P} \tag{3.49}$$

From equation (3.48), we see immediately that $\text{rank}(F_n) \leq \text{rank}(K_{\mathcal{T}_n})$. The following theorem uses this fact, combined with the low–rank cutting matrix decomposition presented in §3.3.1, to efficiently track an analogous decomposition of F_n.

Theorem 3.9. Consider a sequence of regular cutting matrices $\{K_{\mathcal{T}_n}\}_{n=1}^{\infty}$ chosen so that the corresponding embedded trees recursion (3.7) converges for any initial condition $\widehat{x}^0$. Assuming each matrix cuts E edges, it may be decomposed into $N_K = \mathcal{O}(Ed)$ rank–one terms as

$$K_{\mathcal{T}_n} = \sum_{i=1}^{N_K} \omega_{n,i} u_{n,i} u_{n,i}^T \tag{3.50}$$

Using this decomposition, the matrices F_n defined by equation (3.48) may be similarly decomposed as

$$F_n = \sum_{i=1}^{N_K} \omega_{n,i} \left(\widehat{J}_{\mathcal{T}_n}^{-1} u_{n,i}\right) f_{n,i}^T \tag{3.51}$$

where the vectors $f_{n,i}$, $1 \leq i \leq N_K$, may be recursively updated according to

$$f_{n,i} = \widehat{J}_{\mathcal{T}_{n-1}}^{-1} u_{n,i} + \sum_{j=1}^{N_K} \omega_{n-1,j} \left(u_{n-1,j}^T \widehat{J}_{\mathcal{T}_{n-1}}^{-1} u_{n,i} \right) f_{n-1,j} \tag{3.52}$$

The total cost of updating $\{f_{n,i}\}_{i=1}^{N_K}$ is $\mathcal{O}(NE^2d^5)$ operations per iteration.

Proof. We show that the desired decomposition (3.51) holds by induction. Clearly, F_1 may be represented in this form by taking $f_{1,i} = \mathbf{0}$. Now assume such a decomposition holds for F_{n-1}, and consider the calculation of F_n according to equation (3.48):

$$\begin{aligned} F_n &= \widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} \left[\widehat{J}_{\mathcal{T}_{n-1}}^{-1} + F_{n-1} \right] \\ &= \widehat{J}_{\mathcal{T}_n}^{-1} \left(\sum_{i=1}^{N_K} \omega_{n,i} u_{n,i} u_{n,i}^T \right) \left[\widehat{J}_{\mathcal{T}_{n-1}}^{-1} + \sum_{j=1}^{N_K} \omega_{n-1,j} \left(\widehat{J}_{\mathcal{T}_{n-1}}^{-1} u_{n-1,j} \right) f_{n-1,j}^T \right] \\ &= \sum_{i=1}^{N_K} \omega_{n,i} \left(\widehat{J}_{\mathcal{T}_n}^{-1} u_{n,i} \right) \left[\widehat{J}_{\mathcal{T}_{n-1}}^{-1} u_{n,i} + \sum_{j=1}^{N_K} \omega_{n-1,j} f_{n-1,j} \left(u_{n-1,j}^T \widehat{J}_{\mathcal{T}_{n-1}}^{-1} u_{n,i} \right) \right]^T \end{aligned}$$

Therefore, decomposition (3.51) holds assuming $f_{n,i}$ are defined as in equation (3.52).

The computational cost of equation (3.52) is dominated by the summation. There are $\mathcal{O}(Ed)$ terms in the sum, each of which requires the solution of a tree–structured linear system, which in turn takes $\mathcal{O}(Nd^3)$ operations. Thus, each $f_{n,i}$ vector may be updated in $\mathcal{O}(NEd^4)$ operations. Because there are a total of $\mathcal{O}(Ed)$ vectors to update, the total cost per iteration is $\mathcal{O}(NE^2d^5)$. □

Suppose that equation (3.52) has been used to track the $\{f_{n,i}\}_{i=1}^{N_K}$ vectors defining a reduced–rank representation of F_n. The block diagonal elements of F_n may then be calculated in $\mathcal{O}(NEd^3)$ operations using equation (3.51). By adding these elements to the block diagonal elements of $\widehat{J}_{\mathcal{T}_n}^{-1}$, which are calculated in $\mathcal{O}(Nd^3)$ operations by the BP algorithm, we may find the block diagonals of $(\widehat{J}_{\mathcal{T}_n}^{-1} + F_n)$. Thus, by equation (3.49), the recursion in Theorem 3.9 may be used to iteratively calculate the desired marginal error variances $\{\widehat{P}_s\}_{s \in \mathcal{V}}$ at a cost of $\mathcal{O}(NE^2d^5)$ per iteration.

If the ET algorithm is implemented by cycling periodically through a fixed set of trees, as in equation (3.15), many of the terms in equation (3.52) may be precom-

puted and reused from iteration to iteration. However, there will still be $\mathcal{O}(E^2d^2)$ weighted inner products $(u_{n-1,j}^T \widehat{J}_{\mathcal{T}_{n-1}}^{-1} u_{n,i})$ to precompute at a cost of $\mathcal{O}(Nd^3)$ each. In addition, at each iteration it is still necessary to compute $\mathcal{O}(Ed)$ weighted sums of Nd–dimensional vectors $f_{n-1,j}$ to find each of the $\mathcal{O}(Ed)$ new $f_{n,i}$ vectors. Thus, the overall order of the tracking procedure will still be $\mathcal{O}(NE^2d^5)$ per iteration.

When the ET algorithm is implemented with only a single tree, further simplifications are possible. In particular, there will be only $\mathcal{O}(Ed)$ distinct weighted inner products $(u_{n-1,j}^T \widehat{J}_{\mathcal{T}_{n-1}}^{-1} u_{n,i})$ to precompute, since $u_{n-1,i} = u_{n,i}$. Unfortunately, the need to take $\mathcal{O}(E^2d^2)$ weighted sums of Nd–dimensional vectors remains, and thus the $\mathcal{O}(NE^2d^5)$ cost per iteration of equation (3.52) will not change. In the following section, however, we show that a different view of the single–tree expansion does indeed lead to a more efficient error covariance algorithm.

3.3.4 A Fixed Point Approach to Error Covariances

In this section, we present an alternative iterative algorithm for the calculation of error covariances. The direct tracking algorithm of §3.3.3 was based on the idea of tracking the ET series expansion generated by a *sequence* of cutting matrices. In contrast, the algorithm presented in this section solves a set of synthetic inference problems derived from a *single* cutting matrix $K_{\mathcal{T}}$. These synthetic problems may be solved using any of the iterative inference algorithms presented in this thesis, including an ET iteration which uses cutting matrices different from $K_{\mathcal{T}}$.

We begin by considering the fixed point equation (3.46) characterizing the ET series expansion for the case of a single embedded tree $K_{\mathcal{T}}$. Using the low–rank decomposition of $K_{\mathcal{T}}$ presented in §3.3.1, we have

$$\begin{aligned} \widehat{P} &= \widehat{J}_{\mathcal{T}}^{-1} + \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} \widehat{P} \\ &= \widehat{J}_{\mathcal{T}}^{-1} + \widehat{J}_{\mathcal{T}}^{-1} \left(\sum_i \omega_i u_i u_i^T \right) \widehat{P} \\ &= \widehat{J}_{\mathcal{T}}^{-1} + \sum_i \omega_i \left(\widehat{J}_{\mathcal{T}}^{-1} u_i \right) \left(\widehat{P} u_i \right)^T \end{aligned} \tag{3.53}$$

As discussed in §3.3.1, the decomposition of $K_{\mathcal{T}}$ may always be chosen to have at most $\mathcal{O}(Ed)$ vectors u_i. Consider the terms in equation (3.53) individually:

$\widehat{J}_{\mathcal{T}}^{-1} \implies$ Block diagonal elements may be computed by the BP algorithm in $\mathcal{O}(Nd^3)$ operations

$\widehat{J}_{\mathcal{T}}^{-1} u_i \implies$ Each of these $\mathcal{O}(Ed)$ products may be found by the BP algorithm in a total of $\mathcal{O}(NEd^4)$ operations

$\widehat{P} u_i \implies$ Each of these $\mathcal{O}(Ed)$ products may be found using any algorithm for calculating conditional means on the original graph

Combining these observations leads to the following result:

Theorem 3.10. Consider the decomposition of $\widehat{P} = \widehat{J}^{-1}$ defined by equation (3.53), where the regular cutting matrix $K_{\mathcal{T}}$ cuts E edges. Suppose that an iterative algorithm is available which can solve systems of the form $\widehat{J}\bar{x} = \bar{b}$ at a cost of $\mathcal{O}(Nd^3)$ operations per iteration. Then, following a one–time computation requiring $\mathcal{O}(NEd^4)$ operations, a sequence of approximations $\widehat{P}_s^n$ to the marginal error variances $\widehat{P}_s$ may be found at a total cost of $\mathcal{O}(NEd^4)$ operations per iteration.

There are many options for the iterative inference algorithm assumed in Theorem 3.10. If the ET recursion is implemented using the same single cutting matrix $K_{\mathcal{T}}$ used in the fixed point decomposition (3.53), it is straightforward to show that the approximation sequence $\widehat{P}_s^n$ will exactly equal that produced by the direct tracking algorithm presented in §3.3.3. In general, however, faster convergence will be attained by using multiple trees, or even an alternate inference algorithm like belief propagation or conjugate gradients. In such cases, no precise correspondence with the direct tracking procedure can be made.

Although equation (3.53) was motivated by the ET algorithm, nothing about this equation requires that the cutting matrix produce a tree–structured graph. All that is necessary is that the remaining edges form a structure for which exact error covariance calculation is tractable. In addition, note that equation (3.53) gives the correct perturbation of $\widehat{J}_{\mathcal{T}}^{-1}$ for calculating the entire error covariance $\widehat{P}$, not just the block diagonals $\widehat{P}_s$. Thus, if the BP algorithm is extended to calculate a particular set of off–diagonal elements of $\widehat{J}_{\mathcal{T}}^{-1}$, the exact values of the corresponding entries of

$\widehat{P}$ may be found in the same manner.

3.4 Further Convergence Analysis

In §3.2.1 and §3.2.2, we provided a basic characterization of the convergence properties of the embedded trees mean recursion (3.7). For the important special case where the algorithm is implemented by cycling through a fixed set of T cutting matrices $\{K_{\mathcal{T}_n}\}_{n=1}^{T}$, Theorem 3.2 completely characterizes its behavior. In particular, the algorithm's dynamics are determined by the eigenvalues of $\mathbf{E} \triangleq \prod_{n=1}^{T} \widehat{J}_{\mathcal{T}_n} K_{\mathcal{T}_n}$. Convergence of $\widehat{x}^n$ to the solution $\widehat{x}$ of the original estimation problem (3.3), for arbitrary $\widehat{x}^0$, will be achieved if and only if $\rho(\mathbf{E}) < 1$.

Unfortunately, while the conditions of Theorem 3.2 are precise, they are not directly useful in most applications of the ET algorithm. The problem is that the matrix $\mathbf{E}$ is never explicitly calculated by the ET recursion. In principle, given a particular set of cutting matrices, we could form $\mathbf{E}$ and compute its eigenvalues. However, any direct implementation of this calculation would require $\mathcal{O}(N^3)$ operations, and would be intractably complex for the large–scale graphical inference problems which originally motivated the ET algorithm.

In this section, we present a variety of more tractable criteria for checking whether a given set of cutting matrices, when applied periodically as in equation (3.15), lead to a convergent ET recursion. For the special case of a single spanning tree, there exists a very simple necessary and sufficient condition which allows convergence to be much more easily verified. When using multiple trees, tractable necessary conditions are difficult to derive. However, we present a variety of sufficient conditions which provide guidance for the types of cutting matrices which we should expect to perform well in practice.

3.4.1 A Single Spanning Tree

In this section, we examine the ET algorithm when the same cutting matrix $K_{\mathcal{T}}$ is used at every iteration. In this case, the recursion defined by equation (3.7) becomes

$$\widehat{x}^n = \widehat{J}_{\mathcal{T}}^{-1}\left(K_{\mathcal{T}}\widehat{x}^{n-1} + C^T R^{-1} y\right) \tag{3.54}$$

where $\widehat{J}_{\mathcal{T}} = (\widehat{J} + K_{\mathcal{T}})$ is assumed to be nonsingular. As discussed in §3.2.2, the single–tree ET recursion is exactly equivalent to a standard stationary Richardson iteration with preconditioner $\widehat{J}_{\mathcal{T}}$. The matrix $\mathbf{E}$ governing convergence is given by

$$\mathbf{E} = \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} = \left(\widehat{J} + K_{\mathcal{T}}\right)^{-1} K_{\mathcal{T}} \tag{3.55}$$

We would like to find conditions which ensure that $\rho(\mathbf{E}) < 1$. The following theorem, taken from [1], provides a simple necessary and sufficient condition for the convergence of iteration (3.54).

Theorem 3.11. Let A be a symmetric positive definite matrix, and K be a symmetric cutting matrix such that $(A + K)$ is nonsingular. Then

$$\rho\left((A+K)^{-1}K\right) < 1 \qquad \text{if and only if} \qquad A + 2K > 0$$

Proof. See Appendix C.3. □

This result is very closely related to the P–regular splitting theorem of Ortega [55, pp. 122–123], as well as the classic Householder–John theorem [56].

When specialized to the single–tree embedded trees iteration (3.54), Theorem 3.11 gives the following important corollaries:

Corollary 3.12. If the embedded trees algorithm is implemented with a single positive semidefinite cutting matrix $K_{\mathcal{T}}$, the resulting iteration will be convergent for any positive definite inverse error covariance $\widehat{J}$.

Proof. If $\widehat{J}$ is positive definite and $K_{\mathcal{T}}$ is positive semidefinite, then $\left(\widehat{J} + 2K_{\mathcal{T}}\right)$ will clearly be positive definite. □

Corollary 3.13. Suppose that $\widehat{J}$ is diagonally dominant, so that

$$\widehat{J}_{s,s} > \sum_{t \in N(s)} \left| \widehat{J}_{s,t} \right| \tag{3.56}$$

for all $s \in \mathcal{V}$. Then any regular cutting matrix with non–negative diagonal entries will produce a convergent embedded trees iteration.

Proof. From the definition in §3.1, regular cutting matrices only modify the off–diagonal entries of $\widehat{J}$ by setting certain elements to zero. Therefore, the entries set to zero in $\left(\widehat{J} + K_{\mathcal{T}}\right)$ will simply have their signs flipped in $\left(\widehat{J} + 2K_{\mathcal{T}}\right)$, leaving the summation in equation (3.56) unchanged. Then, by the assumption that $K_{\mathcal{T}}$ has non–negative diagonal entries, we are assured that $\left(\widehat{J} + 2K_{\mathcal{T}}\right)$ is diagonally dominant, and hence positive definite. □

Note that Corollary 3.13 ensures that if $\widehat{J}$ is diagonally dominant, setting the diagonal entries of $K_{\mathcal{T}}$ to zero, as suggested in §3.2.3, will give a convergent iteration.

Although Theorem 3.11 completely characterizes the conditions under which the single tree ET iteration converges, it says nothing about the resulting convergence rate. The following theorem, adapted from results in [6], allows the convergence rate $\rho(\mathbf{E})$ to be bounded in certain circumstances.

Theorem 3.14. When implemented with a single positive semidefinite cutting matrix $K_{\mathcal{T}}$, the convergence rate of the embedded trees algorithm is bounded by

$$\frac{\lambda_{\max}(K_{\mathcal{T}})}{\lambda_{\max}(K_{\mathcal{T}}) + \lambda_{\max}(\widehat{J})} \leq \rho\left(\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}\right) \leq \frac{\lambda_{\max}(K_{\mathcal{T}})}{\lambda_{\max}(K_{\mathcal{T}}) + \lambda_{\min}(\widehat{J})}$$

Proof. Because $\widehat{J}_{\mathcal{T}}^{-1}$ is positive definite and $K_{\mathcal{T}}$ is positive semidefinite, by Lemma 2.2(c) of Axelsson [6] we have

$$\lambda_{\min}(\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}) \geq \lambda_{\min}(\widehat{J}_{\mathcal{T}}^{-1}) \lambda_{\min}(K_{\mathcal{T}}) \geq 0$$

This implies that $\rho\left(\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}\right) = \lambda_{\max}\left(\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}\right)$. The desired result then follows by applying the bounds of Axelsson Theorem 2.2 to the maximal eigenvalue. □

Note that increasing the diagonal entries of $K_{\mathcal{T}}$ will also increase the upper bound on $\rho\left(\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}\right)$ provided by Theorem 3.14. This matches the observation made in §3.2.3 that positive semidefinite cutting matrices tend to produce slower convergence rates. Unfortunately, Theorem 3.14 does not address the indefinite cutting matrices which led to the best empirical convergence rates in §3.2.3.

3.4.2 Multiple Spanning Trees

In this section, we examine the behavior of the ET algorithm when it is implemented by periodically cycling through a fixed set of spanning trees, as in equation (3.15). We begin by considering the case where the algorithm alternates between a pair of spanning trees, so that the ET iteration equation (3.7) becomes

$$\widehat{x}^{2n+1} = \widehat{J}_{\mathcal{T}_1}^{-1}\left(K_{\mathcal{T}_1}\widehat{x}^{2n} + C^T R^{-1} y\right) \tag{3.57a}$$

$$\widehat{x}^{2n+2} = \widehat{J}_{\mathcal{T}_2}^{-1}\left(K_{\mathcal{T}_2}\widehat{x}^{2n+1} + C^T R^{-1} y\right) \tag{3.57b}$$

The matrix $\mathbf{E}$ governing convergence is then given by

$$\mathbf{E} = \widehat{J}_{\mathcal{T}_2}^{-1}K_{\mathcal{T}_2}\widehat{J}_{\mathcal{T}_1}^{-1}K_{\mathcal{T}_1} = \left(\widehat{J} + K_{\mathcal{T}_2}\right)^{-1}K_{\mathcal{T}_2}\left(\widehat{J} + K_{\mathcal{T}_1}\right)^{-1}K_{\mathcal{T}_1} \tag{3.58}$$

When multiple trees are used, it is difficult to determine a simple set of necessary and sufficient convergence conditions. However, the following theorem, which is a slight generalization of results in [75], provides a simple set of sufficient conditions which ensure that $\rho(\mathbf{E}) < 1$.

Theorem 3.15. Consider the embedded trees iteration generated by a pair of cutting matrices $\{K_{\mathcal{T}_1}, K_{\mathcal{T}_2}\}$, as in equation (3.57). Suppose that the following three matrices are positive definite:

$$\widehat{J} + K_{\mathcal{T}_1} + K_{\mathcal{T}_2} > 0 \qquad \widehat{J} + K_{\mathcal{T}_1} - K_{\mathcal{T}_2} > 0 \qquad \widehat{J} - K_{\mathcal{T}_1} + K_{\mathcal{T}_2} > 0$$

Then the resulting iteration is convergent ($\rho(\mathbf{E}) < 1$).

Proof. See Appendix C.4. □

The conditions of this theorem show that in the multiple tree case, there may be important interactions between the cutting matrices which effect the convergence of the composite iteration. Note that it is *not* necessary for the cutting matrices to be individually convergent, as characterized by Theorem 3.11, in order for the conditions of Theorem 3.15 to be satisfied.

More generally, we would like to be able to ensure that the ET iteration is convergent when implemented using an arbitrary number of spanning trees. The following theorem employs a singular value analysis of the ET convergence matrix $\mathbf{E}$ to provide such a set of conditions:

Theorem 3.16. Suppose that the ET iteration (3.7) is implemented by periodically cycling through a set of T spanning trees defined by the cutting matrices $\{K_{\mathcal{T}_j}\}_{j=1}^{T}$. Define $K_{\mathcal{T}_{T+1}} \triangleq K_{\mathcal{T}_1}$. Then if any of the following three sets of *non–equivalent* conditions holds for $j = 1, \ldots, T$, we are guaranteed that $\rho(\mathbf{E}) < 1$:

$$\begin{aligned}
\sigma_{\max}\left((\widehat{J} + K_{\mathcal{T}_j})^{-1} K_{\mathcal{T}_j}\right) < 1 \quad &\Longleftrightarrow \quad (\widehat{J} + K_{\mathcal{T}_j})^2 > K_{\mathcal{T}_j}^2 \\
\sigma_{\max}\left((\widehat{J} + K_{\mathcal{T}_{j+1}})^{-1} K_{\mathcal{T}_j}\right) < 1 \quad &\Longleftrightarrow \quad (\widehat{J} + K_{\mathcal{T}_{j+1}})^2 > K_{\mathcal{T}_j}^2 \\
\sigma_{\max}\left((\widehat{J} + K_{\mathcal{T}_j})^{-1} K_{\mathcal{T}_{j+1}}\right) < 1 \quad &\Longleftrightarrow \quad (\widehat{J} + K_{\mathcal{T}_j})^2 > K_{\mathcal{T}_{j+1}}^2
\end{aligned}$$

Proof. See Appendix C.5. □

Unfortunately, in comparison to the conditions presented for the two–tree case in Theorem 3.15, Theorem 3.16 is quite weak. For example, if all cutting matrices are chosen to be the same, Theorem 3.15 reduces to the tight eigenvalue condition presented in §3.4.1. In contrast, all three conditions in Theorem 3.16 reduce to the singular value bound $\sigma_{\max}\left((\widehat{J} + K_{\mathcal{T}})^{-1} K_{\mathcal{T}}\right) < 1$, which is overly conservative. We hypothesize that a stronger set of conditions, analogous to those of Theorem 3.15, hold in the multitree case. Thus far, however, we have been unable to generalize the derivation presented in Appendix C.4.

3.5 Numerical Examples

In this section, we present a variety of numerical simulations chosen to provide insight into the behavior of the algorithms developed in this chapter. We consider both the embedded trees algorithm for computing conditional means developed in §3.2, and the fixed point error covariance algorithm presented in §3.3.4. The results demonstrate the important performance gains produced by the use of multiple spanning trees. In addition, they help to characterize the types of problems for which the ET iteration provides the greatest gains relative to alternate inference techniques.

We begin in §3.5.1 with a discussion of the techniques used to construct and test the random problems which provide most of this section's examples. Then, in §3.5.2, we present a set of simulations on augmented multiscale trees similar to the model presented in the Introduction. We also examine inference algorithm performance on the specific model of §1.1, which provides a more numerically stressing test case. To contrast with the sparsely connected multiscale models, §3.5.3 shows a complementary set of simulations on nearest–neighbor grids. We conclude in §3.5.4 with some observations motivated by these results.

3.5.1 Simulation Conditions

Prior and Measurement Model Construction

With the exception of a set of tests on the multiscale model from §1.1, all of the simulations in the following sections are performed on graphs with fixed structures but randomly chosen distributions. The random variables at each of the nodes in the randomly generated models have dimension $d = 1$. The inverse prior covariance matrix J is constructed by assigning a clique potential $\psi_{s,t}(x_s, x_t)$, as in equation (2.15), to each edge (s, t). We have empirically examined a wide range of methods for choosing these potentials. For the results presented here, however, we focus on two extreme cases which serve to demonstrate a wide range of qualitatively interesting phenomena.

Homogeneous Potentials In the first case, we create a homogeneous prior model by associating the following fixed clique potential with each edge:

$$\psi_{s,t}(x_s, x_t) = \exp\left\{-\frac{1}{2}\begin{bmatrix} x_s & x_t \end{bmatrix}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} x_s \\ x_t \end{bmatrix}\right\} = \exp\left\{-\frac{1}{2}(x_s - x_t)^2\right\} \quad (3.59)$$

By construction, for every node $s \in \mathcal{V}$, $J_{s,s} = \sum_{t \in N(s)} |J_{s,t}|$, ensuring that the resulting inverse prior covariance matrix J is positive semidefinite. We focus on positive semidefinite potentials, rather than positive definite ones, because they lead to the strongest correlations, and therefore the most interesting test cases.

This prior is attractive in the sense that neighboring x_s and x_t are more likely to have similar values. In applications, priors like this are commonly used to enforce smoothness constraints. Note that, if the underlying graph is sufficiently regular (for example, a nearest–neighbor grid with dense measurements of uniform quality), homogeneous models of this form are most efficiently solved through non–iterative methods [25]. However, homogeneous test cases are still useful because they may reveal aspects of iterative methods which are not immediately apparent with more disordered priors. In addition, many of the examples we consider have irregular graphs or sparse measurements, and therefore cannot be solved via direct decomposition.

Disordered Potentials For the second class of prior models, we create a disordered inverse covariance matrix by choosing random clique potentials. For each edge, the clique potential is of the form considered for the examples in §3.2.3:

$$\begin{aligned} \psi_{s,t}(x_s, x_t) &= \exp\left\{-\frac{1}{2}\begin{bmatrix} x_s & x_t \end{bmatrix}\begin{bmatrix} w_{st} & a_{st}w_{st} \\ a_{st}w_{st} & w_{st} \end{bmatrix}\begin{bmatrix} x_s \\ x_t \end{bmatrix}\right\} \\ &= \exp\left\{-\frac{1}{2}w_{st}(x_s + a_{st}x_t)^2\right\} \end{aligned} \quad (3.60)$$

Here, w_{st} is sampled from an exponential distribution with mean 1, while a_{st} is set to $+1$ or -1 with equal probability. As in the homogeneous case, the construction procedure ensures that $J_{s,s} = \sum_{t \in N(s)} |J_{s,t}|$ for all $s \in \mathcal{V}$, so that J is positive semidef-

inite.

While the average strength of these potentials is the same as for the homogeneous case, the exponential distribution leads to a large degree of variability. When $a_{st} = -1$ the potential is attractive, while when $a_{st} = +1$ it is repulsive, favoring neighboring nodes with opposite signs. For the examples we have examined, we have found that nearly all samples from this class of prior models lead to very similar performance curves for each of the iterative methods. Thus, in each case we only present results for a single typical sample from this class.

Measurements We examine both types of prior models under different measurement conditions. In all cases, each measurement is of the form $y_s = x_s + v_s$, $v_s \sim \mathcal{N}(0, \sigma^2)$, where σ^2 is constant across the graph. In some examples, however, measurements are only available at a sparse subset of the nodes. We consider both moderate ($\sigma^2 = 1$) and high ($\sigma^2 = 10$) levels of noise. We do not examine the low-noise case, because for such models $\widehat{J}$ is strongly diagonally dominant and inference is extremely easy.

Convergence Rates

For each test case, we examine several methods for computing conditional means. For the ET algorithm (see §3.2), we consider two spanning trees for each graph, and compare the corresponding pair of single–tree iterations (denoted by ET(1) and ET(2)) to the two–tree iteration (denoted by ET(1,2)) created by alternating between trees. In all cases, we present results for regular cutting matrices with zero diagonals, because the positive semidefinite and negative semidefinite cases introduced in §3.2.3 lead to inferior performance. We compare the ET algorithm's performance to the parallel belief propagation (BP) and unpreconditioned conjugate gradient (CG) methods (see Chapter 2) by plotting the evolution of the normalized residual introduced in §2.4.3.

Error Covariances

The error covariance simulations focus on the performance of the fixed point algorithm developed in §3.3.4. An implementation of this algorithm requires two components: a cutting matrix decomposition to generate the terms in the fixed point equation (3.53), and an iterative algorithm for computing conditional means on the original graph with cycles. For the iterative algorithm, we use the same ET(1,2) iteration employed for the convergence rate examples. The cutting matrix decomposition is constructed by modifying the diagonal entries of the ET(1) cutting matrix, as in Example 3.8, so that only a single term is required for each cut edge.

For each simulation, we compare the fixed point method to the results produced by direct iterative calculation of each column of the error covariance matrix (see §3.3), where ET(1,2) is again used as the iterative method. We also show the approximate error covariances calculated by the loopy belief propagation algorithm, in order to provide an indication of the accuracy gained by using an exact error covariance method. Note that we do not present simulations for the direct tracking error covariance method of §3.3.3, because although this algorithm played an important role in motivating the development of the fixed point method, its quadratic dependence on E, the number of cut edges, makes it clearly inferior.

Comparisons of the three error covariance methods are made by plotting the normalized error metric

$$\frac{\left(\sum_{s\in\mathcal{V}}\left|\widehat{P}_s^n-\widehat{P}_s\right|^2\right)^{\frac{1}{2}}}{\left(\sum_{s\in\mathcal{V}}\left|\widehat{P}_s\right|^2\right)^{\frac{1}{2}}} \tag{3.61}$$

where $\widehat{P}_s$ are the true error variances, and $\widehat{P}_s^n$ the approximations at the n^{th} iteration. To compare the different methods fairly, these errors are plotted as a function of the number of equivalent BP iterations. For an N node graph where E edges are removed to reveal a spanning tree, each iteration of the fixed point method has a cost equivalent to E BP iterations. In contrast, the unstructured calculation of the columns of $\widehat{P}$ has a per–iteration cost equivalent to N BP iterations.

3.5.2 Augmented Multiscale Trees

Inspired by the modeling example presented in the Introduction, in this section we examine the performance of our inference algorithms on graphs created by adding a few additional fine–scale edges to a multiscale tree. We refer to the resulting graphs as *augmented multiscale* models. All of the randomly generated models have a prior covariance with the structure shown in Figure 3-6. The same figure also shows the two spanning trees used for all implementations of the ET algorithm. In every presented example, measurements are only available at the 32 finest scale nodes. At the end of each of the following two subsections, we also discuss inference algorithm performance on the particular multiscale model of §1.1.

Convergence Rates

Figure 3-7 compares the various inference algorithms on the multiscale model of Figure 3-6 with the homogeneous potentials of equation (3.59). At both noise levels, the two–tree ET iteration significantly outperforms both of the single–tree iterations. We hypothesize that the ET(1) iteration improves over the ET(2) iteration because the first spanning tree is better balanced. At moderate noise levels ($\sigma^2 = 1$), ET(1,2) clearly outperforms both CG and BP. At higher noise levels ($\sigma^2 = 10$), all of the algorithms take longer. Note that CG completely fits the eigenspectrum of this homogeneous graph after 43 iterations, leading to superlinear convergence. However, the ET(1,2) algorithm is able to achieve a residual on the order of 10^{-10} after only 28 iterations.

In Figure 3-8, we show results for the same graphical structure, but with disordered potentials chosen as in equation (3.60). The ET(1,2) iteration again provides dramatic gains over the single–tree iterations, especially in the high–noise case. Interestingly, both ET and BP are more effective on this disordered problem, while CG performed much better in the homogeneous case. Note that ET(1,2) significantly outperforms BP, most likely because the tree–structured updates allow information to be transmitted across the graph much more rapidly. In addition, we observe that

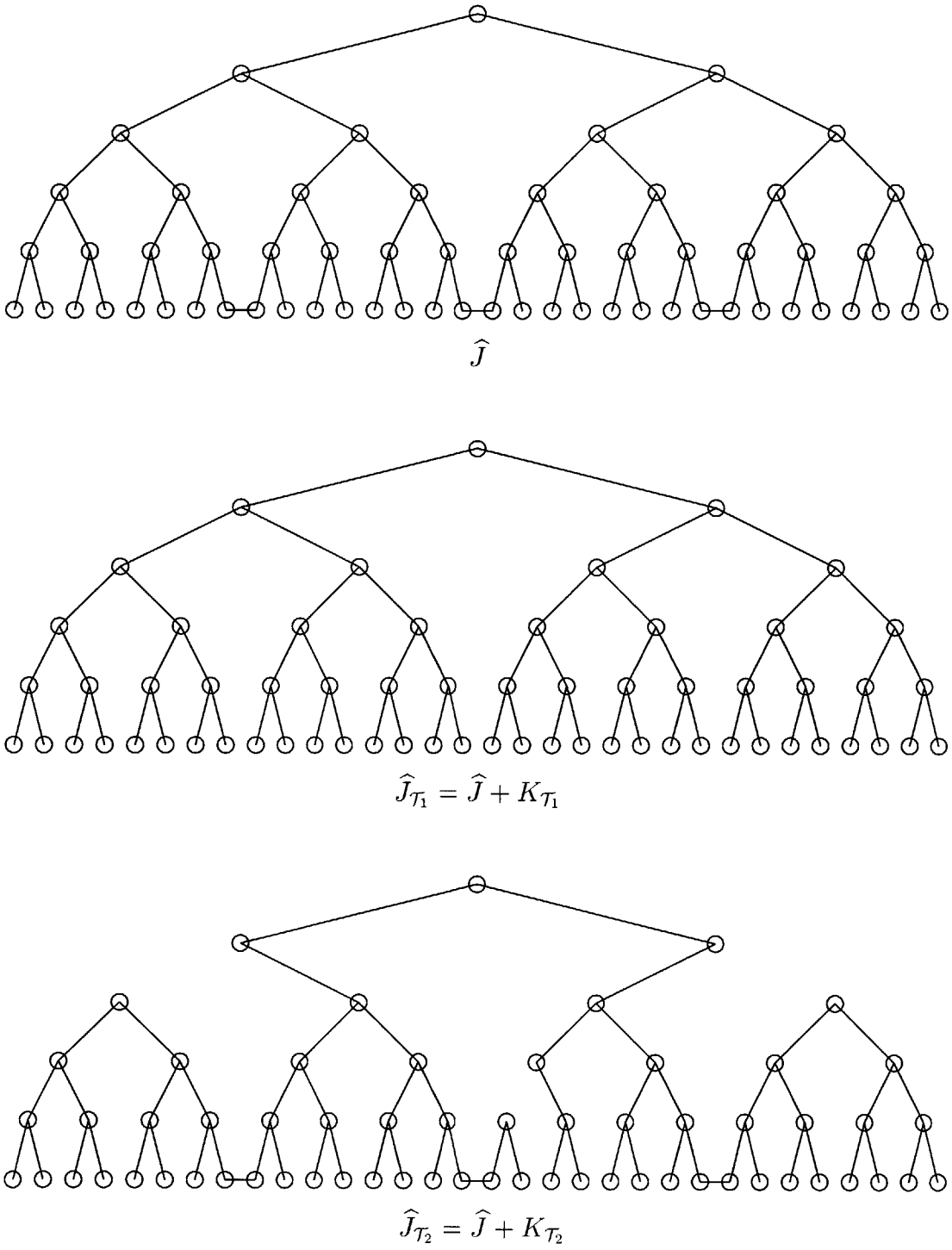
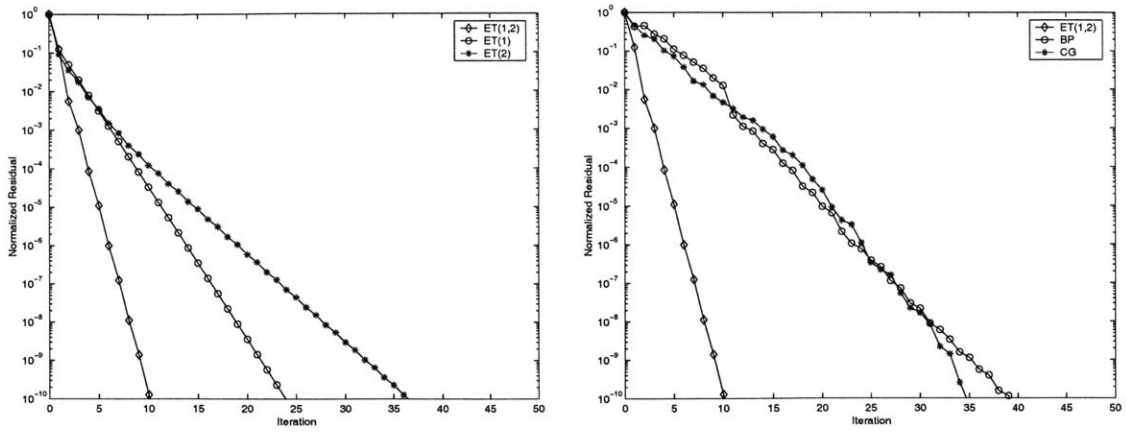
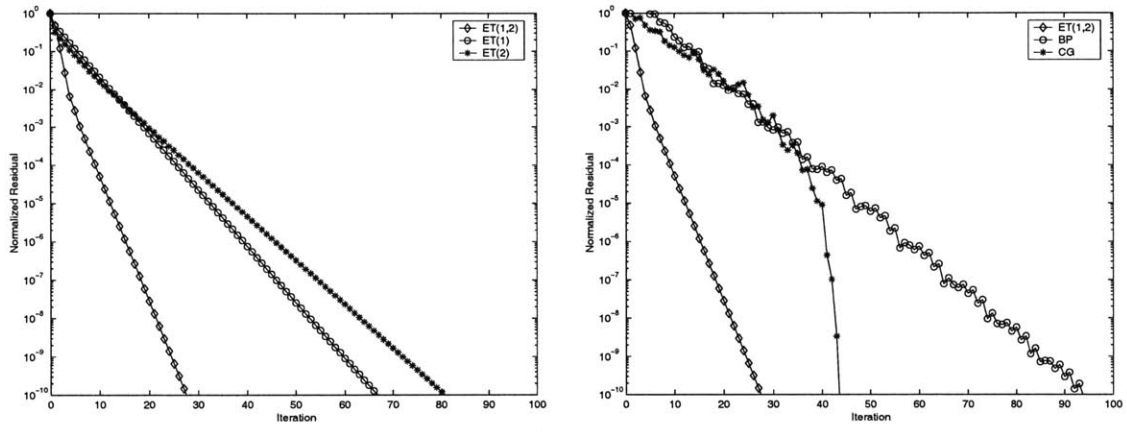


Figure 3-6: Augmented multiscale model with three extra fine scale edges, and the two spanning trees used to implement the ET algorithm. Observations (not shown) are only available at the 32 finest scale nodes.



$\sigma^2 = 1$

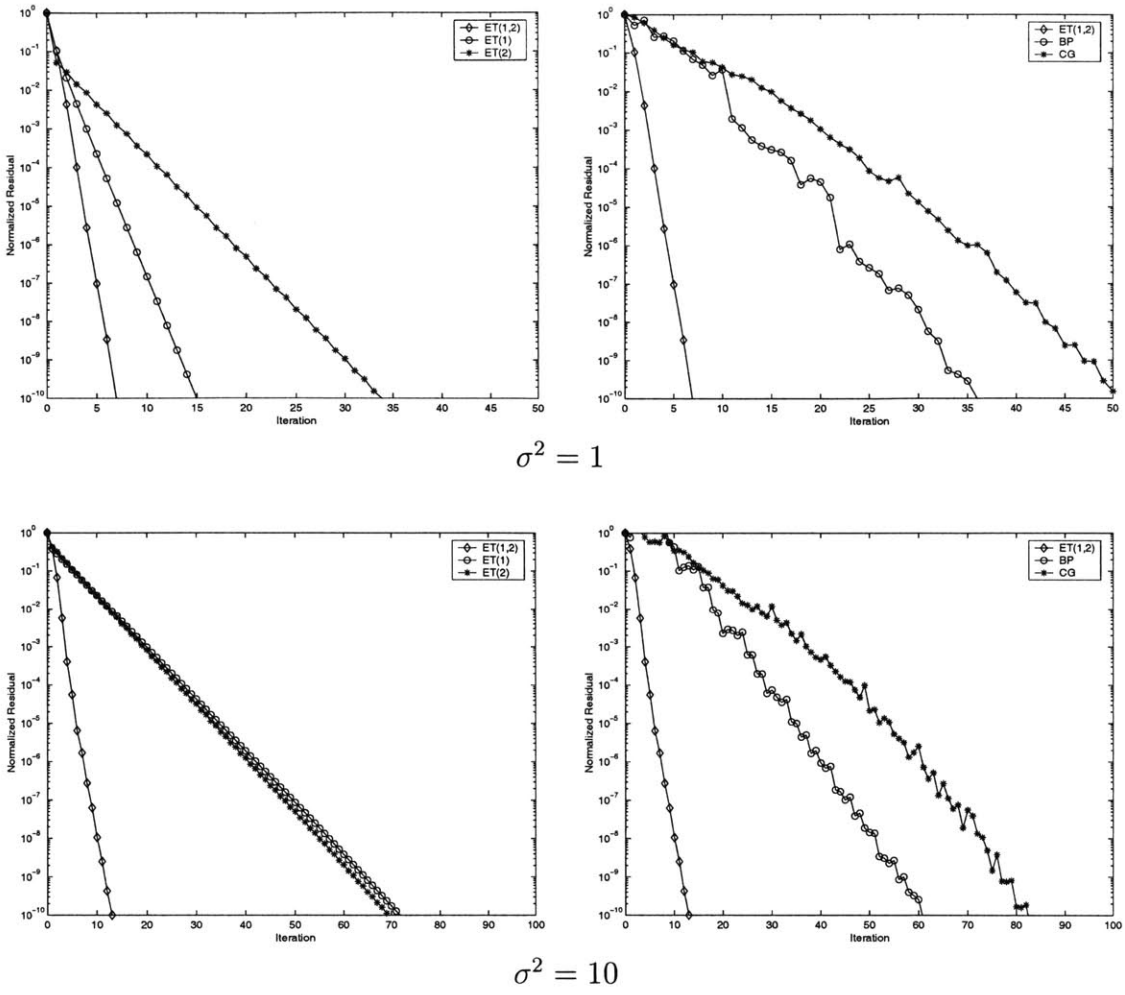


$\sigma^2 = 10$

Figure 3-7: Convergence rates for an augmented multiscale model with homogeneous potentials. Fine scale measurements of two different noise levels are compared.

for this problem, the BP and CG algorithms make fairly erratic iteration–by–iteration progress, greatly reducing the residual on some iterations but increasing it on others. In contrast, as predicted by Theorem 3.2, the ET iteration reduces the error by roughly a constant factor at each iteration.

In addition to the simulations on randomly generated multiscale models, we have compared the same inference algorithms on the multiscale prior constructed in §1.1. To do this, we associated a two–dimensional observation vector $y_s = x_s + v_s$, $v_s \sim \mathcal{N}\left(0, \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]\right)$, with each of the 16 finest scale nodes, and assumed no observations of the coarser scale nodes were available. Note that all of the randomly generated models we have considered in this section are nearly diagonally dominant and nu-



$\sigma^2 = 1$

$\sigma^2 = 10$

Figure 3-8: Convergence rates for an augmented multiscale model with disordered potentials. Fine scale measurements of two different noise levels are compared.

merically well–conditioned. In contrast, because the introductory modeling example was constructed by perturbing a singular tree–structured model, the resulting inverse covariance matrix is fairly ill–conditioned ($\kappa = 7.4 \times 10^4$).

As shown in Figure 3-9, the larger condition number leads to problems for both the BP and CG methods. In particular, the BP algorithm never fully converges, while CG converges very erratically, taking 215 iterations to reach a normalized residual of 10^{-10}. In contrast, the ET algorithm leads to extremely rapid convergence, and produces a consistent decrease in the error after every iteration. To produce the displayed results, we employed cutting matrices with zero diagonals which alternated between cutting the two weakest edges (see Figure 1-2) in the multiscale model's single

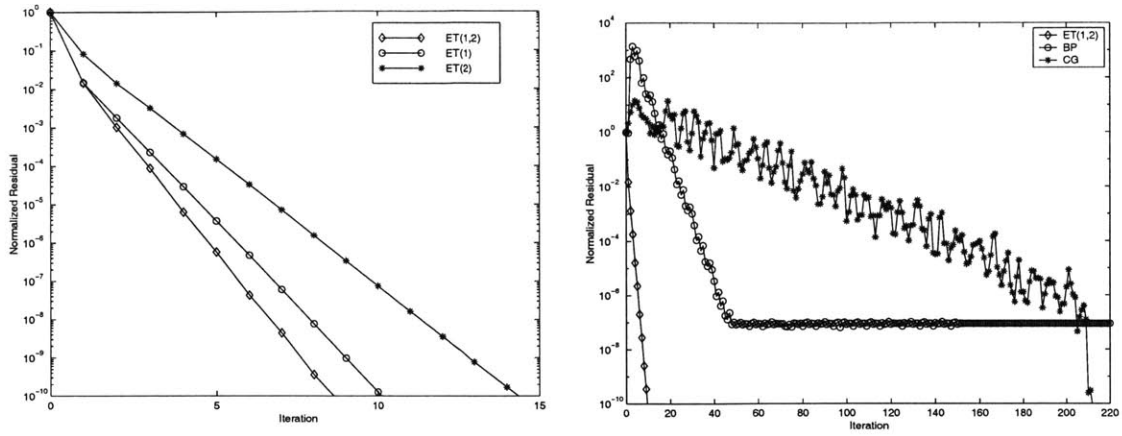
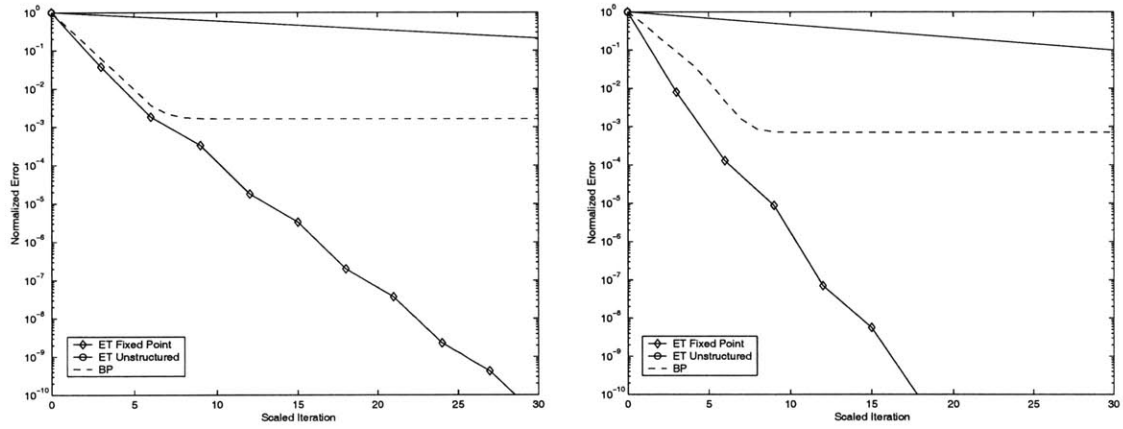
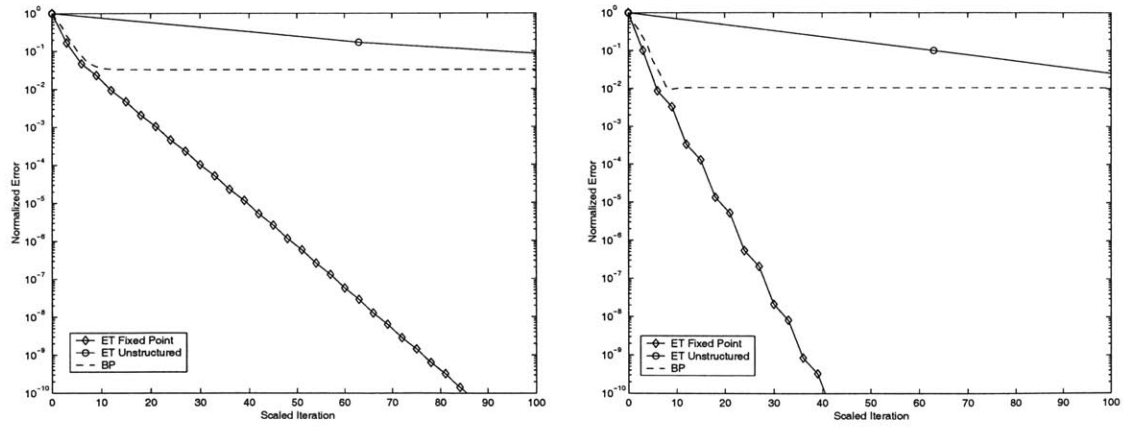


Figure 3-9: Convergence rates for the augmented multiscale model of §1.1.



$\sigma^2 = 1$



$\sigma^2 = 10$

Figure 3-10: Error covariance methods applied to augmented multiscale models. The left column shows the results for homogeneous potentials, and the right for disordered potentials.

cycle. We also tested two–tree iterations which cut stronger edges, and found that the ET algorithm always converged, in most cases nearly as quickly as the iteration presented here.

Error Covariances

Figure 3-10 compares the performance of the different error covariance algorithms on the same augmented multiscale models examined in the previous subsection. Each iteration of the fixed point and unstructured error covariance methods improves the error covariance estimates by approximately the same amount that the ET(1,2) method improved the conditional mean estimates on the corresponding model. However, because only 3 of the edges connecting this model's 63 nodes must be removed to reveal a spanning tree, the fixed point algorithm's per–iteration cost is approximately 21 times lower. For these models, the ET fixed point method is also clearly superior to the BP algorithm, because after each iteration, it gives a better estimate of the error variances than the BP algorithm can produce in an equivalent computational time. For the high–noise models in particular, the BP algorithm's final error variances are fairly inaccurate, while the ET fixed point method can provide much higher accuracy with a small amount of additional computation.

We also applied the ET fixed point error covariance algorithm to the multiscale model of §1.1. The resulting iteration converged at a rate very similar to that displayed by the embedded trees conditional mean iteration for this graph (see Figure 3-9). Because only a single edge had to be cut, the total cost of each fixed point iteration was quite small, providing significant gains over the unstructured calculation of the entire error covariance matrix. Due to the BP algorithm's irregular convergence for this model, it again gave inferior error variance estimates at each iteration. The asymptotic normalized error (see equation (3.61)) of the BP error variance estimates was 7.0×10^{-3}, while the ET fixed point method produced more accurate error variances after only a single iteration.

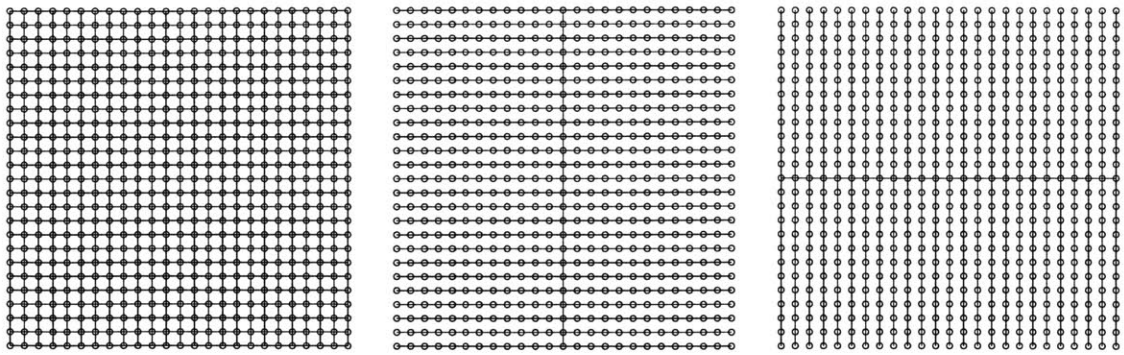


Figure 3-11: 25×25 nearest–neighbor grid, and the two spanning trees used to implement the ET algorithm. Observations are not explicitly shown.

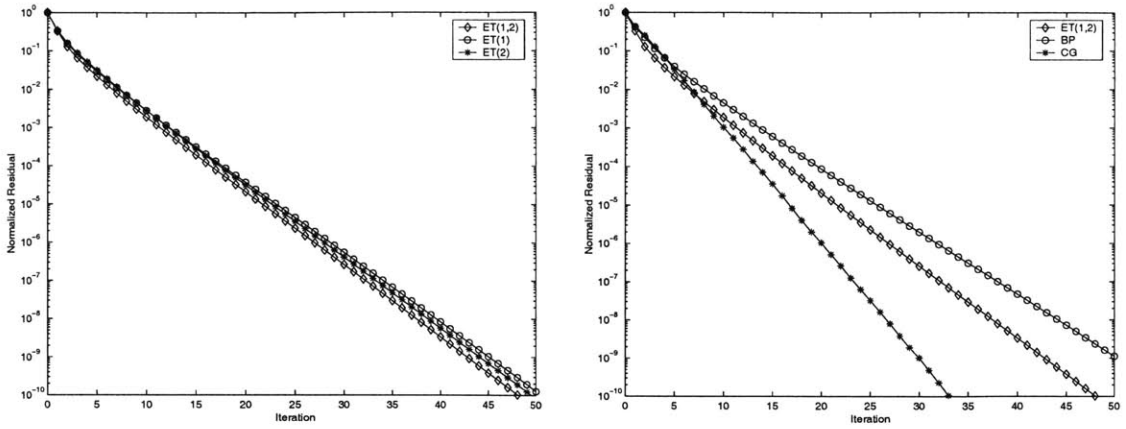
3.5.3 Nearest–Neighbor Grids

In order to contrast with the sparsely connected multiscale models examined in the previous section, we now compare the various inference methods on a set of 25×25 nearest neighbor grids. Figure 3-11 shows this graph, as well as the two spanning trees used for all implementations of the ET algorithm. For each of the two prior model clique potential types, we show inference results for a set of dense measurements at both moderate ($\sigma^2 = 1$) and high ($\sigma^2 = 10$) noise levels. We also examine inference performance when moderately noisy measurements are only available at a randomly chosen 20% of the nodes.

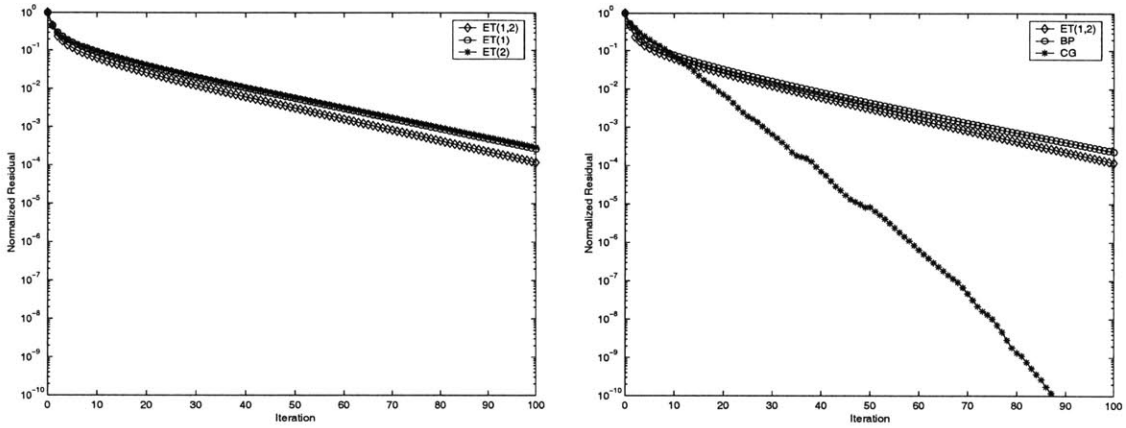
Convergence Rates

Figure 3-12 compares the various inference algorithms on a 25×25 grid with the homogeneous potentials of equation (3.59). In contrast with the multiscale model results, there is very little difference between the single–tree and two–tree ET iterations. Both the ET and BP algorithms have asymptotic convergence rates that compare quite poorly with CG, especially when noise is high or measurements are sparse. Interestingly, however, ET and BP perform better on the first few iterations before being overtaken by CG.

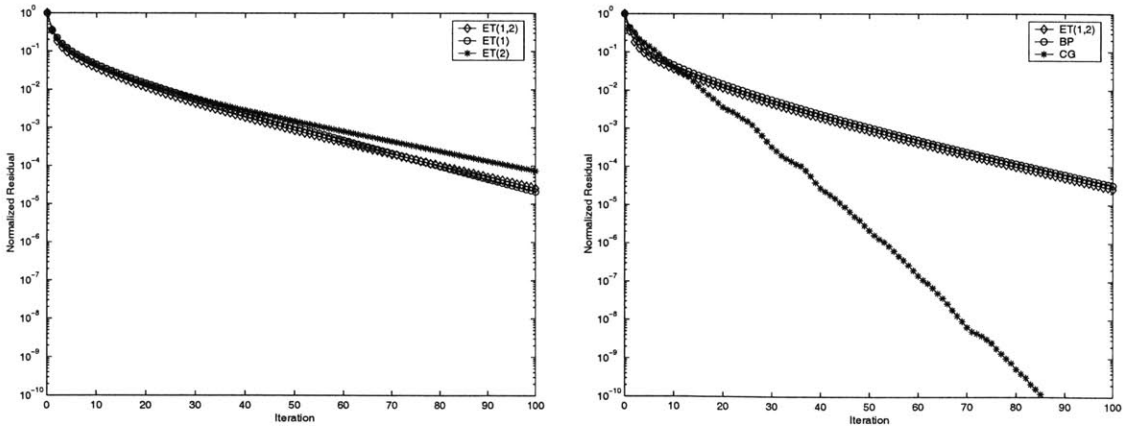
In Figure 3-13, we show results for a 25×25 grid with disordered potentials chosen as in equation (3.60). Unlike the homogeneous case, for this problem the



$\sigma^2 = 1$ (dense)

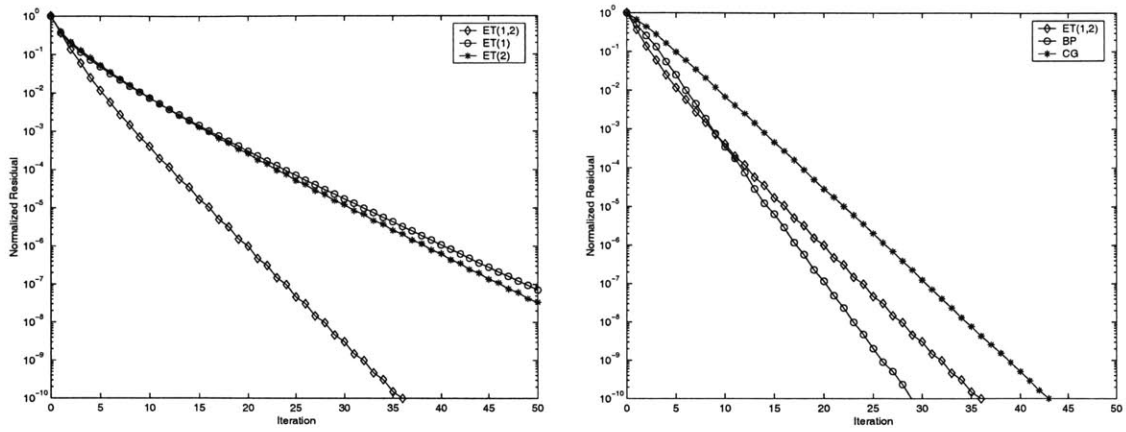


$\sigma^2 = 10$ (dense)

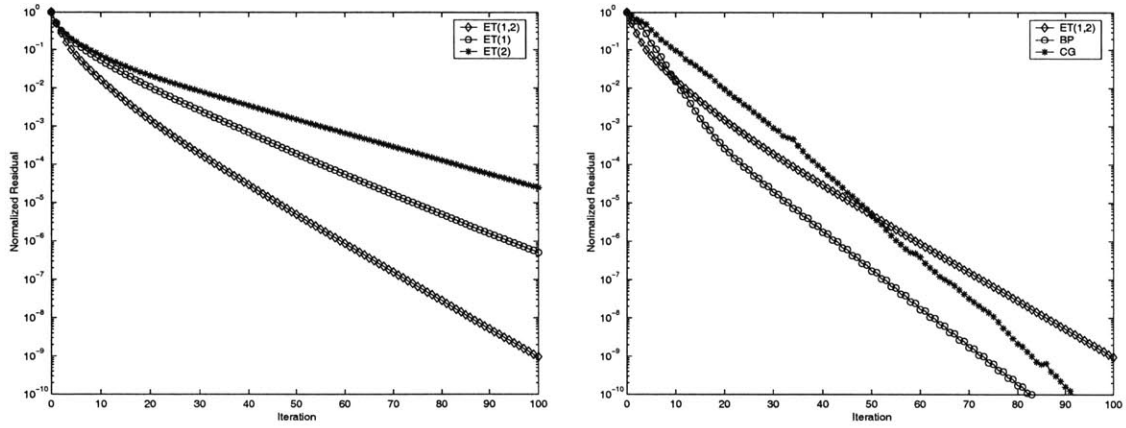


$\sigma^2 = 1$ (sparse)

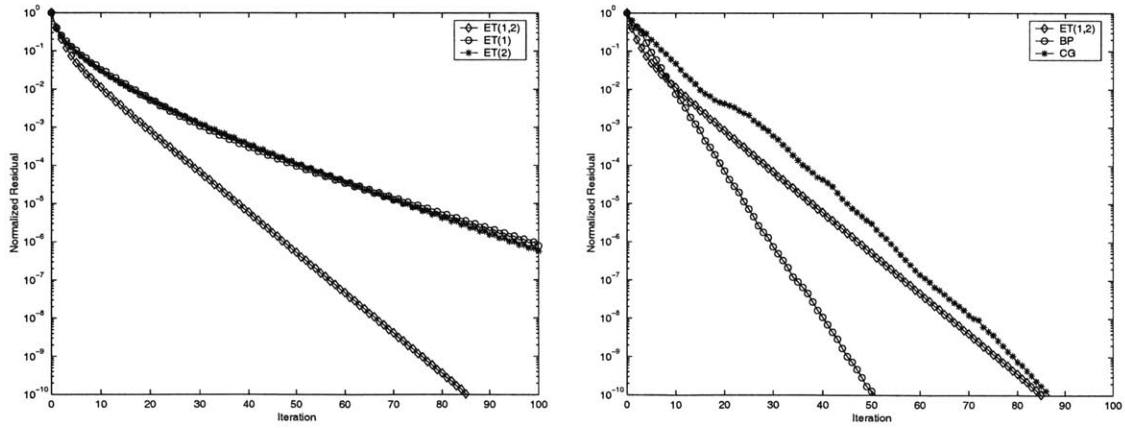
Figure 3-12: Convergence rates for a 25×25 nearest–neighbor grid with homogeneous potentials. Three measurement models are compared: dense measurements with moderate and high noise levels, and sparse measurements with moderate noise.



$\sigma^2 = 1$ (dense)



$\sigma^2 = 10$ (dense)



$\sigma^2 = 1$ (sparse)

Figure 3-13: Convergence rates for a 25×25 nearest–neighbor grid with disordered potentials. Three measurement models are compared: dense measurements with moderate and high noise levels, and sparse measurements with moderate noise.

ET(1,2) iteration does provide significant gains relative to the single–tree iterations. With dense measurements, all three algorithms lead to similar asymptotic convergence rates, with BP performing moderately better. In the sparse measurement case, however, BP performs significantly better.

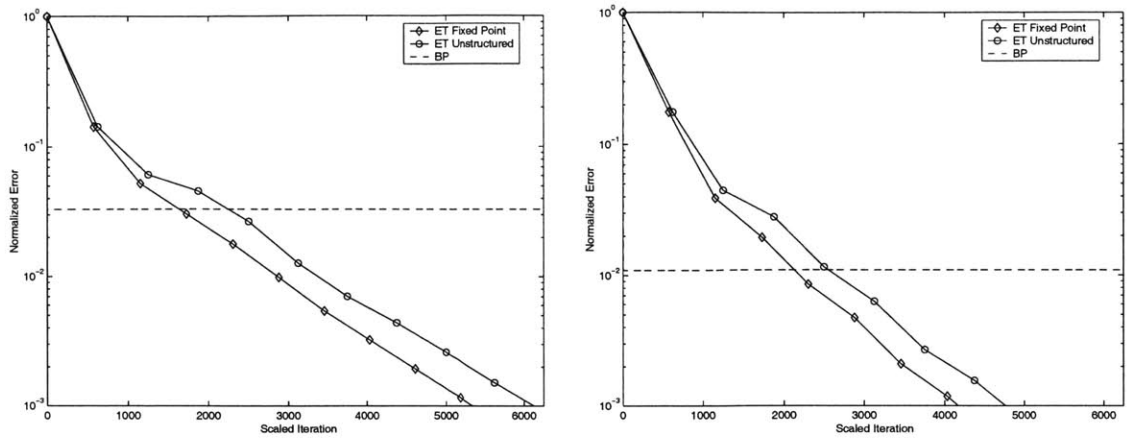
Error Covariances

Figure 3-14 compares the performance of the different error covariance algorithms on the 25×25 grid. As with the multiscale model, each iteration of the fixed point method improves the error covariance estimates by an amount comparable to the conditional mean improvement produced by the corresponding ET(1,2) iteration. However, because the grid is very densely connected, each iteration of the fixed point method has a cost equivalent to $24^2 = 576$ BP iterations. This is only marginally better than the $N = 625$ relative cost of the unstructured, column–by–column, iterative calculation of the entire error covariance matrix.

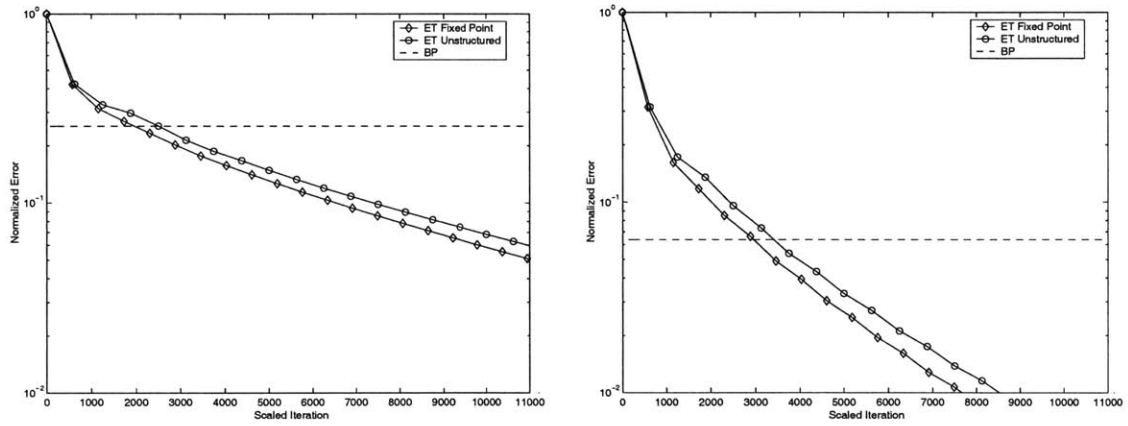
Asymptotically, $\mathcal{O}(N)$ edges must be cut from a grid to reveal a spanning tree, and the fixed point method will provide no gains. Also, note that the BP approximations to the error covariances, while quickly computed, are relatively inaccurate. Thus, none of these algorithms can efficiently and accurately calculate error variances on large grids.

3.5.4 Observations

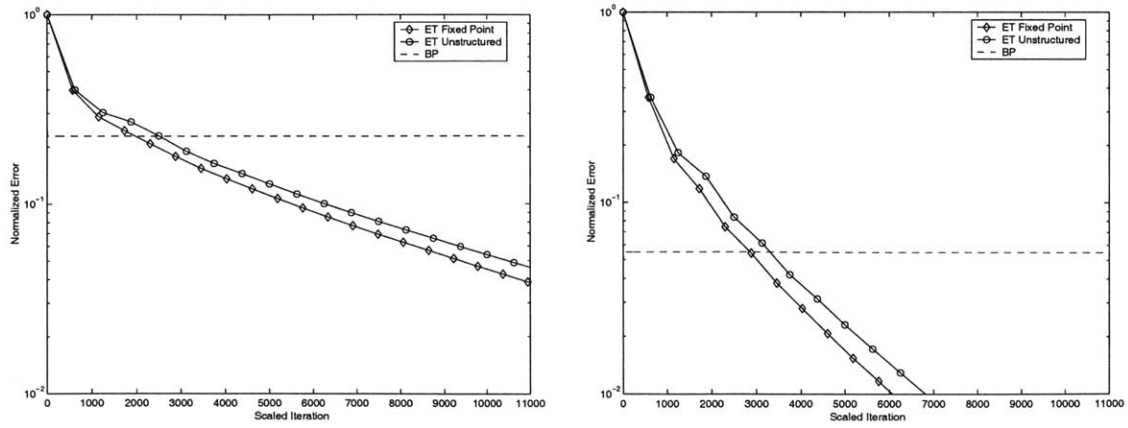
From the preceding simulations, several features of the embedded trees algorithm are immediately apparent. Clearly, the ET iteration converges fastest when it is possible to choose spanning trees which well approximate the full graph's structure. In particular, for the sparsely augmented multiscale models, the ET algorithm's convergence is dramatically faster than either BP or CG. Even for more densely connected models, however, the two–tree ET iteration is often competitive with other techniques. The two–tree case is especially intriguing, because for models where there is irregularity in either the graph structure or the potential strengths, it typically converges *much*



$\sigma^2 = 1$ (dense)



$\sigma^2 = 10$ (dense)



$\sigma^2 = 1$ (sparse)

Figure 3-14: Error covariance methods applied to a 25×25 nearest–neighbor grid. The left column shows the results for homogeneous potentials, and the right for disordered potentials.

faster than either of the corresponding single-tree iterations.

Although the ET algorithm's performance is somewhat effected by numerical potential values, it is far more dependent on the structure of the underlying graph. In contrast, as suggested by the convergence rate bounds in §2.4.2, the conjugate gradient iteration is relatively robust to graph structure variations, but is more sensitive to the numerical properties of the overall inverse error covariance matrix. Thus, the ET algorithm may provide an attractive alternative for the ill–conditioned problems which CG finds most difficult.

Unlike the conditional mean iteration, the embedded tree–based error covariance algorithm is only effective for sparsely connected models. Unfortunately, for graphs where $\mathcal{O}(N)$ edges must be cut, the per–iteration cost is simply too large. However, for augmented multiscale models such as that presented in the Introduction and examined at the end of §3.5.2, it rapidly converges to error covariance approximations which are typically much more accurate than those provided by loopy belief propagation.

Chapter 4

Accelerated Tree–Based Iterations

In the preceding chapter, we presented and analyzed an embedded tree based algorithm for solving Gaussian inference problems defined on graphs with cycles. In this chapter, we present a series of increasingly sophisticated techniques which use the basic ET algorithm to develop more rapidly convergent iterations. Throughout, we focus solely on methods for calculating conditional means. Note, however, that any of the accelerated iterations we propose may also be easily integrated into the error covariance algorithm of §3.3.4, leading to similar performance improvements.

We begin in §4.1 by considering the relationship between the ET algorithm and the diagonal entries of the corresponding cutting matrices. We demonstrate a simple method for setting these diagonal values which often leads to accelerated convergence relative to the zero–diagonal cutting matrices considered in Chapter 3. Then, in §4.2, we show that the dynamics of the ET iteration reveal information about the subspace containing the errors in the current estimate. This observation leads to the development of a rank–one correction procedure which first estimates the principal error direction, and then cancels those errors on subsequent iterations.

Although the rank–one procedure is effective in some cases, we show that it does not make full use of the error information revealed by the ET iterates. This motivates the use of the ET algorithm as a preconditioner for the conjugate gradient iteration, which we consider in §4.3. We demonstrate that for graphs which are nearly tree–structured, preconditioning by a single spanning tree can be guaranteed to provide

good performance. For more densely connected graphs, we show experimental results which suggest that both single and multiple tree preconditioners can be effective. The chapter concludes with a discussion of connections to existing preconditioning theory.

4.1 Diagonal Acceleration

As discussed in §3.1, associated with any spanning tree $\mathcal{G}_\mathcal{T}$ of a Gaussian graphical model, there is an entire family of tree–structured inverse covariances $J_\mathcal{T}$, each corresponding to a particular choice of cutting matrix $K_\mathcal{T}$. Later sections of Chapter 3 demonstrated that in the context of the embedded trees algorithm, it is often desirable to choose cutting matrices that are as sparse as possible. This constraint naturally motivated the class of regular cutting matrices (see §3.1), which minimize the number of nonzero off–diagonal entries.

Within the class of regular cutting matrices, however, we are still free to set the diagonal entries corresponding to nodes from which edges are cut. In the examples of §3.2.3, we briefly examined three intuitive choices for these diagonal entries, and concluded that among these options, setting the diagonals to zero consistently gave the best performance. This choice was partially justified by theoretical results in §3.4 which showed that for diagonally dominant models, cutting matrices with non–negative diagonals are guaranteed to produce a convergent ET iteration. In addition, ET iterations based on two zero–diagonal cutting matrices led to very rapid convergence for most (but not all) of the simulations presented in §3.5.

In this section, we examine the relationship between cutting matrix diagonals and the ET algorithm in more detail. We begin by revisiting some of the more difficult problems from §3.5. We demonstrate that by choosing cutting matrices with relatively small *negative* diagonal values, we may often produce two–tree iterations with dramatically improved convergence rates. We then discuss connections to other linear algebraic acceleration techniques, and provide some guidance for how effective diagonal values should be chosen.

4.1.1 Numerical Examples

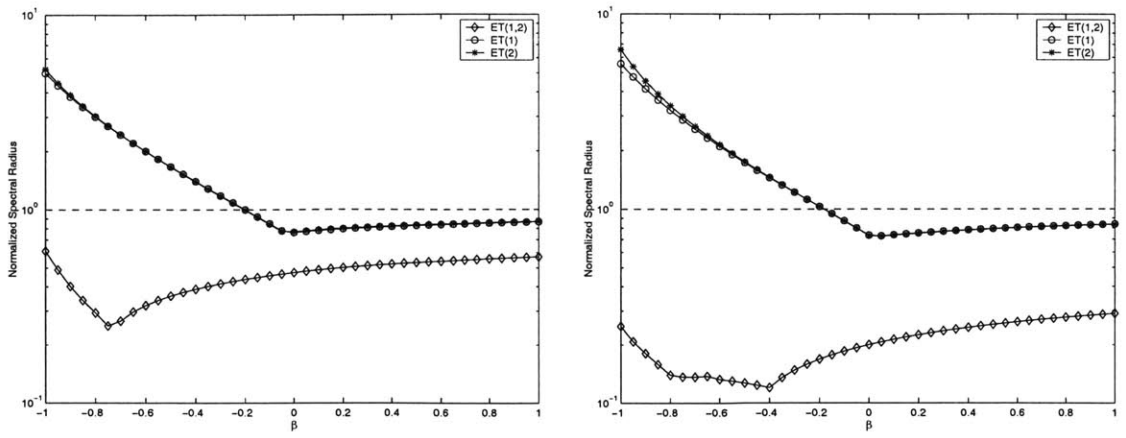
In this section, we empirically investigate the properties of a single–parameter family of regular cutting matrices. Given a fixed spanning tree $\mathcal{G}_{\mathcal{T}}$, and the corresponding uniquely defined off–diagonal cutting matrix entries, we set the diagonal entries of $K_{\mathcal{T}}$ according to the following equation:

$$(K_{\mathcal{T}})_{s,s} = \beta \sum_{t \in N(s)} \left| (K_{\mathcal{T}})_{s,t} \right| \tag{4.1}$$

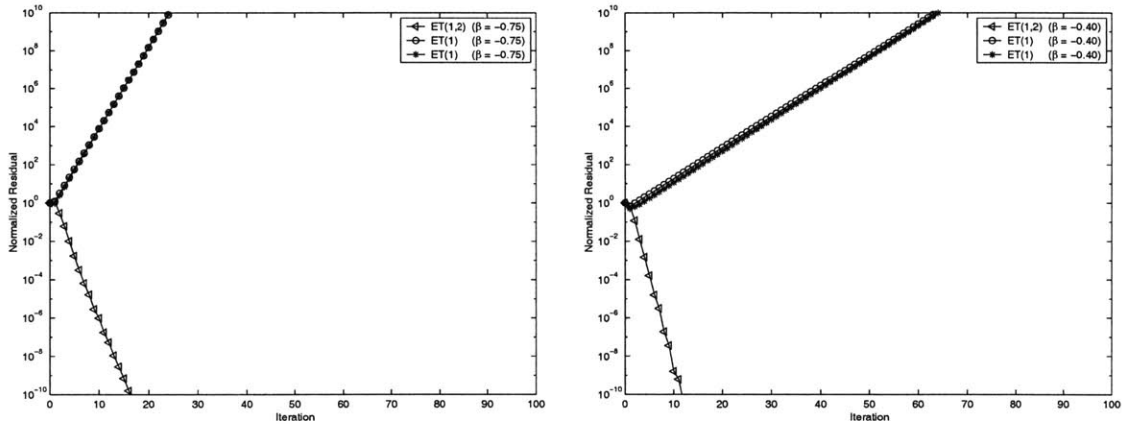
Here, β is a scalar parameter indexing the elements of this family, which we will refer to as the *diagonal acceleration* parameter. When $\beta = 0$, we recover the zero–diagonal case used for most of the simulations in Chapter 3. Similarly, the choices $\beta = +1$ and $\beta = -1$ correspond to the positive semidefinite and negative semidefinite cases considered in §3.2.3. We have also experimented with other methods for setting the diagonal entries of $K_{\mathcal{T}}$, such as adding multiples of the identity matrix, but have found that equation (4.1) provides the most consistent, effective results.

Although we have examined the effects of diagonal acceleration on all of the models from §3.5, we focus here on the high–noise cases for which the zero–diagonal ET iteration performed most poorly. Figure 4-1 presents the simulation results for the augmented multiscale models considered in §3.5.2 in both the homogeneous and disordered potential cases. Recall from Theorem 3.2 that the asymptotic convergence rate of the ET algorithm is determined by the normalized spectral radius $\rho(\mathbf{E})^{\frac{1}{T}}$. As shown in Figure 4-1(a), this convergence rate depends strongly on the value of β. For all values of $\beta \geq 0$, both the single–tree and two–tree iterations are convergent. However, as β is increased the convergence rate becomes increasingly slow. Note that in the single–tree case, this convergence is guaranteed by the diagonal dominance of $\widehat{J}$ (see Corollary 3.13).

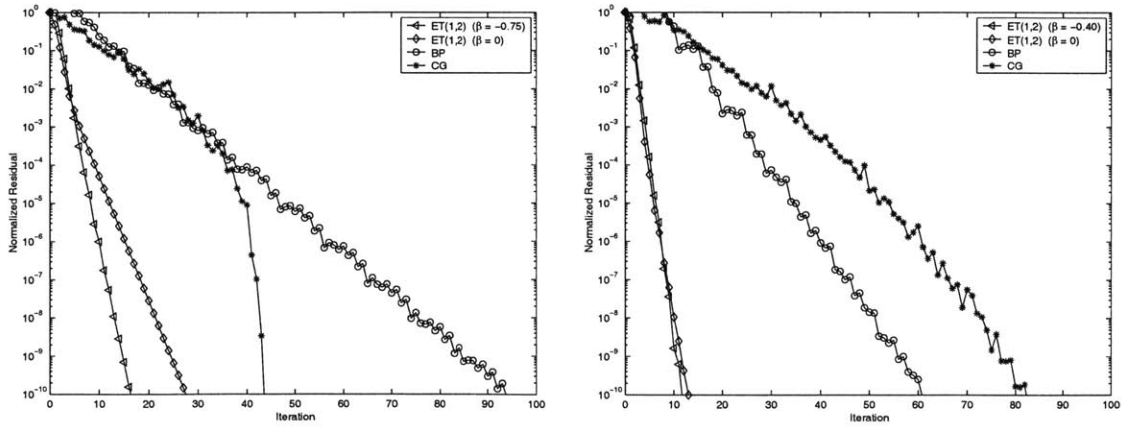
As β is decreased below zero, the spectral radius for the single–tree iterations rapidly increases, so that they quickly become divergent. In the two–tree case, however, the spectral radius monotonically *decreases* for small negative values of β, leading to more rapid convergence. For both potential types, there is an optimal



(a)



(b)



(c)

Figure 4-1: Diagonal acceleration of the ET algorithm for the augmented multiscale model of Figure 3-6 ($\sigma^2 = 10$) with homogeneous potentials (left column) or disordered potentials (right column). (a) Convergence rates for different diagonal acceleration parameters β. (b) Divergent single–tree iterations lead to rapidly convergent two–tree iterations. (c) Comparison of the diagonally accelerated ET iteration to the zero–diagonal iteration, BP, and CG.

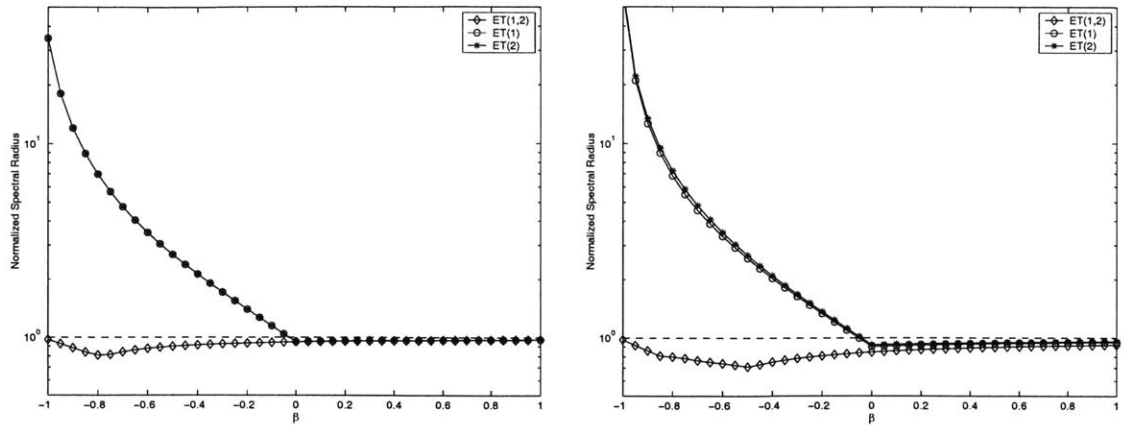
value of the diagonal acceleration parameter which produces the fastest convergence ($\beta \approx -0.70/-0.40$ for homogeneous/disordered potentials). Beyond this point, convergence rate decreases again. For values of $\beta < -1$ (not shown), the iteration is either divergent or very slowly convergent.

In Figure 4-1(b), we show the convergence behavior of the ET iteration for the diagonal acceleration parameter values which led to the fastest two–tree iteration. Interestingly, the single–tree iterations are actually divergent for these β values, implying that each tree must effectively cancel the complementary tree's worst error modes. Figure 4-1(c) plots the diagonally accelerated two–tree iteration against the results presented for these models in §3.5.2. In the homogeneous case, acceleration reduces the number of iterations by nearly 40% relative to the zero–diagonal iteration. However, in the mixed potential case, both ET iterations converge too rapidly for the asymptotic gains predicted by Figure 4-1(a) to be observed.

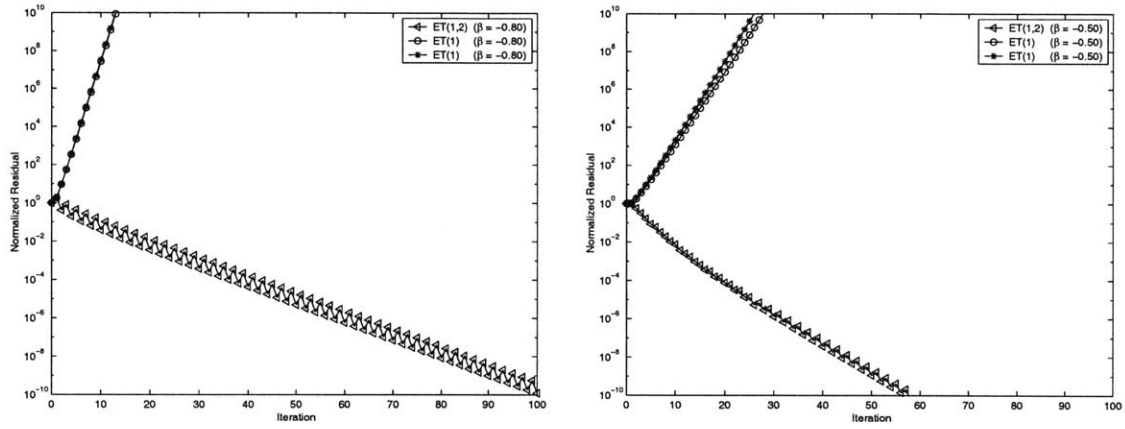
In Figure 4-2, we examine the effects of diagonal acceleration on the 25×25 grids of §3.5.3 with dense measurements and high noise levels. Note that the qualitative relationship between β and the ET algorithm's convergence rate, as shown in Figure 4-2(a), is very similar to that observed for the multiscale models. In particular, for β values between -1 and 0, we again observe two divergent single–tree iterations combining to create a two–tree iteration which converges more rapidly than the zero diagonal case. In Figure 4-2(b), we explicitly see the way in which each tree cancels the other's error modes, as at every other iteration the overall residual actually increases. Although it would seem that such oscillatory behavior would be inefficient, Figure 4-2(c) shows that it actually leads to an overall convergence rate which is significantly faster than the zero–diagonal ET iteration. Note that the diagonally accelerated two–tree iteration provides the best performance for the disordered potential case, and is nearly as fast as CG in the homogeneous case.

4.1.2 Discussion

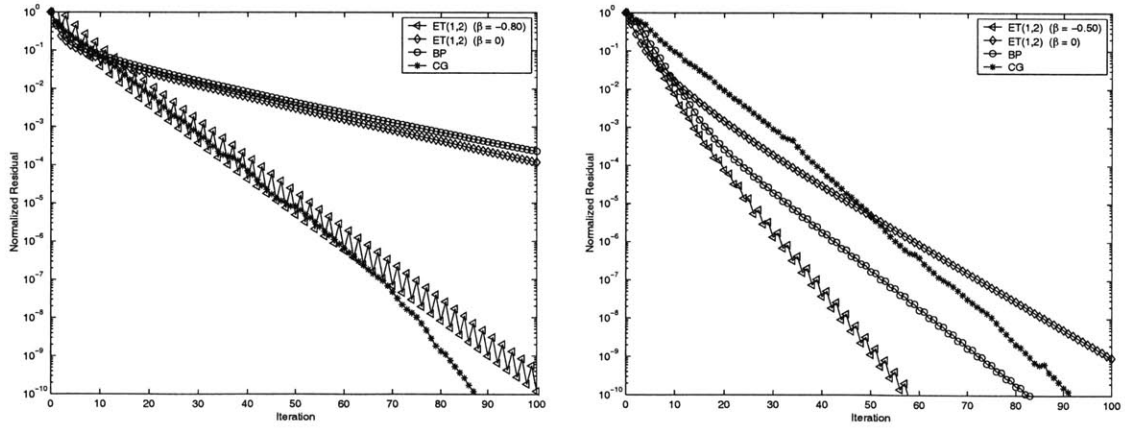
For all of the examples in the previous section, the convergence rate of the ET iteration became progressively slower as β was increased above zero. To understand this



(a)



(b)



(c)

Figure 4-2: Diagonal acceleration of the ET algorithm for a 25×25 nearest–neighbor grid ($\sigma^2 = 10$) with homogeneous potentials (left column) or disordered potentials (right column). (a) Convergence rates for different diagonal acceleration parameters β. (b) Divergent single–tree iterations lead to rapidly convergent two–tree iterations. (c) Comparison of the diagonally accelerated ET iteration to the zero–diagonal iteration, BP, and CG.

behavior, consider the single–tree ET iteration equation (3.7) when a multiple δI of the identity matrix is added to the cutting matrix:

$$\widehat{x}^n = \left(\widehat{J} + K_{\mathcal{T}} + \delta I\right)^{-1} \left(\delta \widehat{x}^{n-1} + K_{\mathcal{T}} \widehat{x}^{n-1} + C^T R^{-1} y\right) \tag{4.2}$$

We see that as δ becomes large, each iteration will place greater emphasis on the previous iterate $\widehat{x}^{n-1}$, and will therefore effectively take a smaller "step" towards the solution. In fact, Corollary 3.12 guarantees that once δ becomes large enough so that $K_{\mathcal{T}} + \delta I$ is positive semidefinite, the step size will be small enough to ensure a convergent iteration regardless of the numerical structure of $\widehat{J}$. However, such decreases in the step size will also naturally slow the convergence rate.

In contrast, negative β values can be seen as taking a larger step away from the current estimate. As demonstrated by the previous simulations, somewhat larger step sizes can improve the convergence rate, but if the step size becomes too large the iteration may become unstable. Note that in some ways, the diagonally accelerated embedded trees iterations are similar to the successive overrelaxation (SOR) method, which adjusts the step size of the Gauss–Seidel iteration to speed convergence (see §2.4.1). However, the precise mechanisms by which the accelerated ET and SOR methods implement their step size adjustments are *not* equivalent.

Diagonal acceleration methods similar to those considered here have also been examined in the context of the ADI method (see §3.2.5). For homogeneous problems on regular grids, effective techniques have been developed which change the cutting matrix diagonals at each iteration, in the process minimizing a Chebyshev polynomial–based upper bound on the convergence rate [73, 75]. However, as discussed in detail by Birkhoff and Varga [14], these analyses only apply to problems for which all cutting matrices and spanning tree distributions share a common set of eigenvectors, and are thus *very* limited.

Although we have not been able to derive a theoretically optimal method for selecting the diagonal acceleration parameters, the experimental results of the previous section suggest some natural heuristics. In particular, for every model we have

examined, improvements over the zero–diagonal convergence rate are attained for a fairly broad range of $\beta \in (-1, 0)$. Thus, on large–scale problems one could perform a small number of ET iterations with a few different β values in this range, and then complete the calculation with the fastest of the trial parameters. It might also be possible to develop methods which monitor the current convergence rate and adapt the acceleration parameter accordingly, as is done in sophisticated implementations of the SOR method [7]. However, we do not explore such possibilites further here.

4.2 Rank–One Corrections

In this section, we consider the dynamics of the embedded trees algorithm in more detail. Using the geometric properties derived in §3.2, we demonstrate that the sequence of past ET iterates $\{\widehat{x}^k\}_{k=0}^n$ reveals important information about the error $(\widehat{x}^n - \widehat{x})$ in the current estimate. We then present a rank–one correction algorithm which uses this information to accelerate the ET algorithm's convergence. We conclude with a set of numerical examples demonstrating this algorithm, and a discussion of its strengths and limitations.

4.2.1 Geometry and Inference on Single Cycle Graphs

In order to demonstrate the intuition behind the rank–one correction procedure, we begin by examining the evolution of the ET algorithm on a single–cycle graph where each node represents a scalar Gaussian random variable. Suppose that the ET algorithm is implemented by cutting the same edge at every iteration with a positive semidefinite cutting matrix $K_{\mathcal{T}}$ chosen so that $\text{rank}(K_{\mathcal{T}}) = 1$. From equation (3.10), we see that the difference $(\widehat{x}^2 - \widehat{x}^1)$ between the first and second iterates is given by

$$(\widehat{x}^2 - \widehat{x}^1) = \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} (\widehat{x}^1 - \widehat{x}^0) \tag{4.3}$$

Note that $\text{rank}(\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}) \leq \text{rank}(K_{\mathcal{T}}) = 1$. Therefore, $(\widehat{x}^2 - \widehat{x}^1)$ must be the eigenvector corresponding to the single nonzero eigenvalue of $\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}$. Let $v \triangleq (\widehat{x}^2 - \widehat{x}^1)$ denote this

eigenvector, and λ its associated eigenvalue. Then, again applying equation (3.10), we have

$$(\widehat{x}^3 - \widehat{x}^2) = \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} (\widehat{x}^2 - \widehat{x}^1) = \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} v = \lambda v \tag{4.4}$$

We may explicitly calculate λ by dividing any non–zero entry of $(\widehat{x}^3 - \widehat{x}^2)$ by the corresponding entry of v. Alternatively, λ may be determined from the normal equations as

$$\lambda = \frac{v^T (\widehat{x}^3 - \widehat{x}^2)}{v^T v} \tag{4.5}$$

This least squares view arises again in the subsequent analysis.

Because $K_{\mathcal{T}}$ is positive semidefinite, we know from Corollary 3.12 that the ET iteration must be convergent, or equivalently $|\lambda| < 1$. This in turn allows us to immediately calculate the conditional mean $\widehat{x}$ as follows:

$$\begin{aligned} \widehat{x} &= \lim_{n \to \infty} \widehat{x}^n = \widehat{x}^2 + \sum_{n=3}^{\infty} (\widehat{x}^n - \widehat{x}^{n-1}) \\ &= \widehat{x}^2 + \sum_{n=1}^{\infty} \left(\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} \right)^n (\widehat{x}^2 - \widehat{x}^1) = \widehat{x}^2 + \sum_{n=1}^{\infty} \left(\widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}} \right)^n v \\ &= \widehat{x}^2 + \sum_{n=1}^{\infty} \lambda^n v = \widehat{x}^2 + \frac{\lambda}{1 - \lambda} v \end{aligned} \tag{4.6}$$

Therefore, using only the information provided by the first three iterations of the ET algorithm, we can directly compute the exact conditional mean for the single cycle graph. Fundamentally, the reason we are able to perform this calculation is that, as shown by Theorem 3.6, the error $(\widehat{x}^n - \widehat{x})$ is constrained to lie in a subspace whose dimension is upper–bounded by the minimal cutting matrix rank. Thus, when $\text{rank}(K_{\mathcal{T}}) = 1$, we can cancel these errors by adding an appropriately chosen multiple of the single vector v which defines that subspace.

There is an alternate way to perform the previous calculation which leads more naturally to the rank–one algorithm derived in the following section. Using the fact that the conditional mean $\widehat{x}$ satisfies the fixed point relationship of equation (3.5),

we may rewrite equation (4.6) as

$$\begin{aligned}\widehat{x} &= \widehat{J}_{\mathcal{T}}^{-1}\left(K_{\mathcal{T}}\left(\widehat{x}^2+\frac{\lambda}{1-\lambda}v\right)+C^TR^{-1}y\right)\\ &= \widehat{J}_{\mathcal{T}}^{-1}\left(K_{\mathcal{T}}\widehat{x}^2+C^TR^{-1}y\right)+\frac{\lambda}{1-\lambda}\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}v\\ &= \widehat{x}^3+\hat{\delta}\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}v \end{aligned} \tag{4.7}$$

where we have defined $\hat{\delta} \triangleq (\lambda/(1-\lambda))$. Then, because equation (4.7) must equal equation (4.6), we see that $\hat{\delta}$ may be determined directly as

$$\begin{aligned}\hat{\delta} &= \arg\min_{\delta}\left|\left|(\widehat{x}^2+\delta v)-(\widehat{x}^3+\delta\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}v)\right|\right|\\ &= \frac{(v-\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}v)^T(\widehat{x}^3-\widehat{x}^2)}{\left|\left|v-\widehat{J}_{\mathcal{T}}^{-1}K_{\mathcal{T}}v\right|\right|^2}\end{aligned} \tag{4.8}$$

Note that equation (4.8) allows us to compute the optimal perturbation factor $\hat{\delta}$ without explicitly determining the eigenvalue λ.

4.2.2 An Automatic Correction Procedure

When the cutting matrix rank is larger than one, as it must be for any graph with multiple cycles, we cannot directly calculate $\widehat{x}$ from the first three ET iterates as in the previous section. However, as we demonstrate below, a very similar method can be used to reduce the dimension of the subspace in which the errors $(\widehat{x}^n-\widehat{x})$ evolve by one. If the direction of reduction is chosen to match a dominant error mode, this can greatly accelerate the convergence of the resulting rank–one corrected ET iteration.[1]

The rank–one procedure we consider here applies to ET iterations which periodically cycle through a fixed set of T cutting matrices $\{K_{\mathcal{T}_j}\}_{j=1}^{T}$. Because multiple–tree iterations are already intended to internally cancel the worst error modes of the corresponding single–tree iterations, we will work directly with the subsampled sequence of estimates $\{\widehat{x}^{nT}\}_{n=0}^{\infty}$ when applying our rank–one perturbations. To simplify the

[1]Multiple rank correction procedures are also possible, but the computational cost is cubic in the dimension of the subspace in which the correction is to be made. See §4.2.4 for more details.

presentation, it will be helpful to define some new notation. Let $\mathcal{F}_T(x)$ be the function which applies the standard ET iteration equation (3.7) T times, using the cutting matrices $\{K_{\mathcal{T}_j}\}_{j=1}^{T}$, from the initial condition $\widehat{x}^0 = x$.[2] Using Theorem 3.3, we may then express every T^{th} iterate of the standard (uncorrected) ET iteration as

$$\widehat{x}^{Tn} = \mathcal{F}_T\left(\widehat{x}^{T(n-1)}\right) = \mathbf{E}_T \widehat{x}^{T(n-1)} + M_T^{-1} C^T R^{-1} y \tag{4.9}$$

where M_T^{-1} is the ET preconditioner matrix of equation (3.19), and $\mathbf{E}_T$ is the matrix whose spectral radius determines the asymptotic convergence rate of the ET iteration (see Theorem 3.2).

The rank–one correction procedure we examine is directly motivated by the geometric single–cycle correction procedure of equations (4.7, 4.8). In particular, given a vector v defining the desired direction of error cancellation, the rank–one corrected ET iteration is given by

$$\begin{aligned} \widehat{x}^{Tn} = \mathcal{H}_T\left(\widehat{x}^{T(n-1)}, v\right) &\triangleq \mathcal{F}_T\left(\widehat{x}^{T(n-1)} + \hat{\delta}_{n-1} v\right) \\ &= \mathcal{F}_T\left(\widehat{x}^{T(n-1)}\right) + \hat{\delta}_{n-1} \mathbf{E}_T v \end{aligned} \tag{4.10}$$

where at each iteration, $\hat{\delta}_{n-1}$ is calculated according to

$$\hat{\delta}_{n-1} = \frac{(v - \mathbf{E}_T v)^T \left(\mathcal{F}_T\left(\widehat{x}^{T(n-1)}\right) - \widehat{x}^{T(n-1)}\right)}{||v - \mathbf{E}_T v||^2} \tag{4.11}$$

Note that for fixed v, $\mathcal{H}_T(x, v)$ is a linear function of x. Also, it is straightforward to show that the accelerated iteration has a fixed point $\mathcal{H}_T(\widehat{x}, v) = \widehat{x}$ at the correct conditional mean. Computationally, the implementation of equations (4.10, 4.11) is little different from the standard ET algorithm. The vector $\mathbf{E}_T v$ can be precomputed at a cost slightly less than T standard ET iterations, and then stored and reused at every acceleration step. Thus, at each accelerated iteration, only a single extra inner

[2]Note that $\mathcal{F}_T(x)$ uses the cutting matrices $\{K_{\mathcal{T}_j}\}_{j=1}^{T}$, inverse error covariance matrix $\widehat{J}$, and observation vector y to perform the ET iterations. However, because only the estimate $\widehat{x}^n$ changes from iteration to iteration, for notational simplicity we do not explicitly denote these other dependencies.

product (to determine $\hat{\delta}_{n-1}$) is required.

Rank–one corrections like those defined in the preceding paragraph have been proposed and studied by Bertsekas [11, 12] as a means of accelerating the value iteration method for solving infinite–horizon optimal control problems.[3] The following theorem, which is adapted from his results, justifies the form of the acceleration procedure.

Theorem 4.1. Consider the modified ET iteration defined by equations (4.10, 4.11), where the correction is performed along the vector v. Let $\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$ be the eigenvalues of the matrix $\mathbf{E}_T$ which governs the convergence of the standard ET iteration (see Theorem 3.2). Then if v is an eigenvector of $\mathbf{E}_T$ corresponding to λ_1, the eigenvalues governing the convergence of the linear operator $\mathcal{H}_T(x, v)$ are equal to $\{0, \lambda_2, \ldots, \lambda_n\}$.

Proof. This result follows directly from Proposition 1 of Bertsekas [12]. □

Thus, if v is chosen as an eigenvector corresponding to the dominant mode of $\mathbf{E}_T$, all errors parallel to v will indeed be cancelled. The convergence rate of the rank–one corrected iteration is then determined by the magnitude of the subdominant eigenvalue.

To apply equations (4.10, 4.11), we must first determine a direction vector v along which to perform the rank–one correction. Although the dominant eigenvector of $\mathbf{E}_T$ is not directly available, we can obtain an estimate of it from the standard ET iterations. From equation (4.9), we see that

$$\left(\widehat{x}^{Tn} - \widehat{x}^{T(n-1)}\right) = \mathbf{E}_T \left(\widehat{x}^{T(n-1)} - \widehat{x}^{T(n-2)}\right) \tag{4.12}$$

Let $\{v_i\}_{i=1}^m$ be the eigenvectors of $\mathbf{E}_T$, and $\{\lambda_i\}_{i=1}^m$ their corresponding eigenvalues. If the initial difference is given by $\left(\widehat{x}^T - \widehat{x}^0\right) = \sum_i \eta_i v_i$, the difference after n subsampled

[3] The classical value iteration procedure [11] is a non–linear version of the Gauss–Jacobi iteration described in §2.4.1.

iterations will be

$$\left(\widehat{x}^{Tn} - \widehat{x}^{T(n-1)}\right) = \sum_{i=1}^{m} \eta_i \lambda_i^{n-1} v_i \tag{4.13}$$

Thus, if $|\lambda_1| > |\lambda_j|$ for all $j \neq 1$, $\left(\widehat{x}^{Tn} - \widehat{x}^{T(n-1)}\right)$ will become increasingly aligned with the dominant eigenvector v_1 as $n \longrightarrow \infty$.

Based on the preceding discussion, we have implemented the rank–one correction procedure using the following two–phase algorithm, as suggested by Bertsekas [12]:

1. From some starting vector $\widehat{x}^0$, iterate the standard ET recursion (4.9). After every T^{th} iteration, calculate the change $\left(\widehat{x}^{Tn} - \widehat{x}^{T(n-1)}\right)$ in the estimate values, and compute the cosine of its angle θ with the previous first difference:

$$\cos\theta = \frac{\left(\widehat{x}^{Tn} - \widehat{x}^{T(n-1)}\right)^T \left(\widehat{x}^{T(n-1)} - \widehat{x}^{T(n-2)}\right)}{||\widehat{x}^{Tn} - \widehat{x}^{T(n-1)}|| \cdot ||\widehat{x}^{T(n-1)} - \widehat{x}^{T(n-2)}||} \tag{4.14}$$

2. On the first iteration where $(1 - \cos\theta)$ becomes smaller than $\epsilon = 10^{-4}$, set $v = \left(\widehat{x}^{Tn} - \widehat{x}^{T(n-1)}\right)$. Subsequent estimates are then calculated using the rank–one corrected ET recursion (4.10), where the same *fixed* error direction vector v is used at every iteration.

Note that the phase one iterations used to estimate the correction direction are not wasted, as they improve the estimate of the conditional mean as well.

The two–phase rank one correction algorithm makes the implicit assumption that $\mathbf{E}_T$ has a real eigenvalue of largest magnitude. When the ET algorithm is implemented with a single tree, the following lemma guarantees that all of the eigenvalues of $\mathbf{E}_1 = \widehat{J}_{\mathcal{T}}^{-1} K_{\mathcal{T}}$ are real:

Lemma 4.2. Let $\widehat{J}$ be symmetric positive definite, and $K_{\mathcal{T}}$ be a symmetric matrix such that $(\widehat{J} + K_{\mathcal{T}})$ is invertible. Then all of the eigenvalues of $(\widehat{J} + K_{\mathcal{T}})^{-1} K_{\mathcal{T}}$ are real.

Proof. Let λ be any eigenvalue of $(\widehat{J} + K_{\mathcal{T}})^{-1} K_{\mathcal{T}}$, and v its corresponding eigenvector. Since $\lambda v = (\widehat{J} + K_{\mathcal{T}})^{-1} K_{\mathcal{T}} v$, we clearly have $\lambda v^T (\widehat{J} + K_{\mathcal{T}}) v = v^T K_{\mathcal{T}} v$. Then, since the invertibility of $(\widehat{J} + K_{\mathcal{T}})$ ensures that $v^T (\widehat{J} + K_{\mathcal{T}}) v \neq 0$, $\lambda = v^T K_{\mathcal{T}} v / v^T (\widehat{J} + K_{\mathcal{T}}) v$. The result then follows from the symmetry of $K_{\mathcal{T}}$ and $(\widehat{J} + K_{\mathcal{T}})$. □

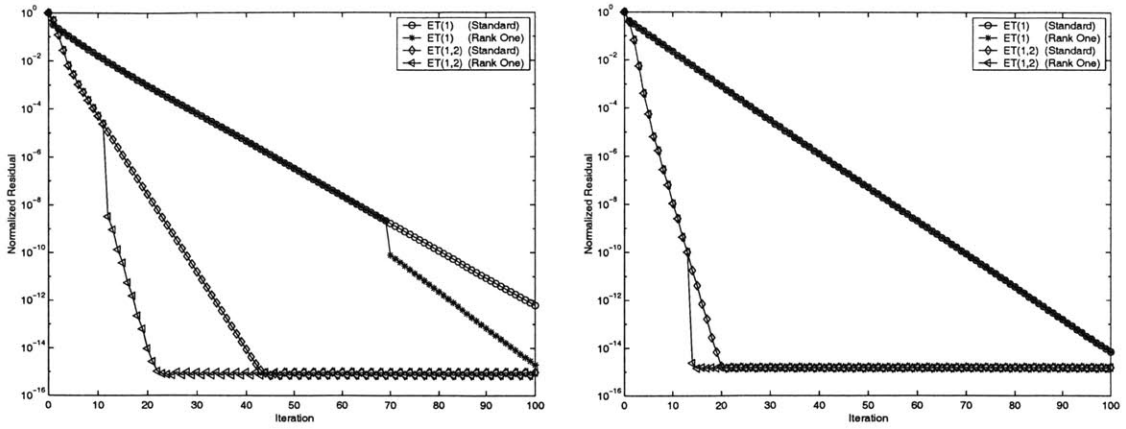


Figure 4-3: Rank one correction of the ET algorithm applied to the augmented multiscale model of Figure 3-6 ($\sigma^2 = 10$) with homogeneous potentials (left) or disordered potentials (right).

Thus, in the single–tree case, the acceleration procedure will only fail when there are two or more real eigenvalues of equal, and largest, magnitude. In the multi–tree case, $\mathbf{E}_T$ will generally have complex eigenvalues. However, for all of the models of §3.5, the eigenvalues of largest magnitude are real. For this reason, in the following section, we examine rank–one acceleration of both the single–tree and two–tree iterations.

4.2.3 Numerical Examples

In this section, we apply the rank–one correction procedure of §4.2.2 to some of the sample models from §3.5. As in §4.1.1, we present results only for the high–noise cases where the standard ET iteration was least effective. For each model, we use the same spanning trees employed in §3.5. In all cases, we employ zero–diagonal cutting matrices rather than the diagonally accelerated iterations of §4.1 in order to separate the effects of the two types of acceleration.

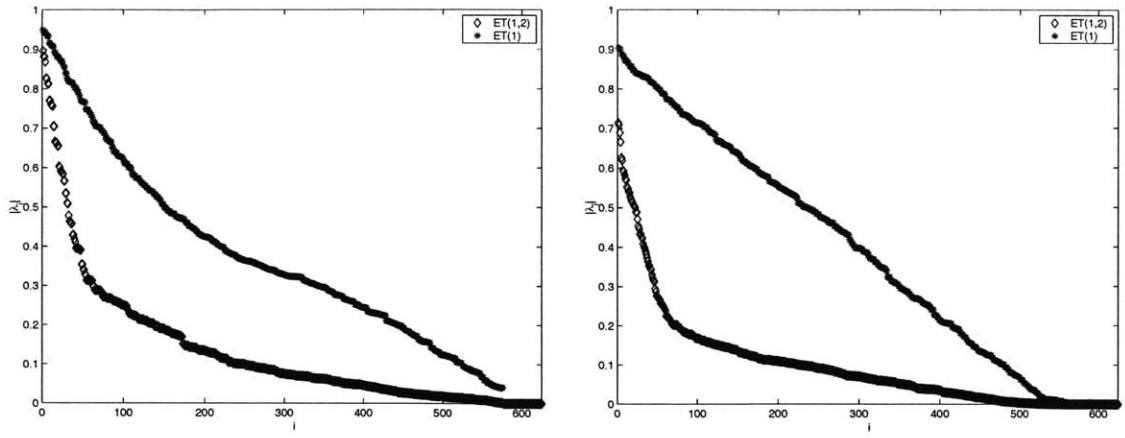
Figure 4-3 presents the simulation results for the augmented multiscale model of Figure 3-6 in both the homogeneous and disordered potential cases. Unsurprisingly, the effectiveness of the rank one correction procedure is strongly dependent on the difference in magnitude between the largest and second–largest eigenvalues of $\mathbf{E}_T$. The most impressive results are seen in the homogeneous case. For the two–tree iteration, the first and second largest eigenvalues are 0.2234 and 0.0410, respectively. After six subsampled iterations, the first–difference vectors are sufficiently aligned (as

measured by equation (4.14)) to begin the rank–one correction procedure. This leads to a sudden drop in the residual as most of the dominant error mode is eliminated, followed by faster convergence in subsequent iterations. In the single–tree case, the dominant and subdominant eigenvalues are less separated ($\lambda = 0.7685, -0.7097$). Thus, more iterations are required to determine the dominant eigenvector, and the acceleration produced by rank one correction is noticeably smaller.

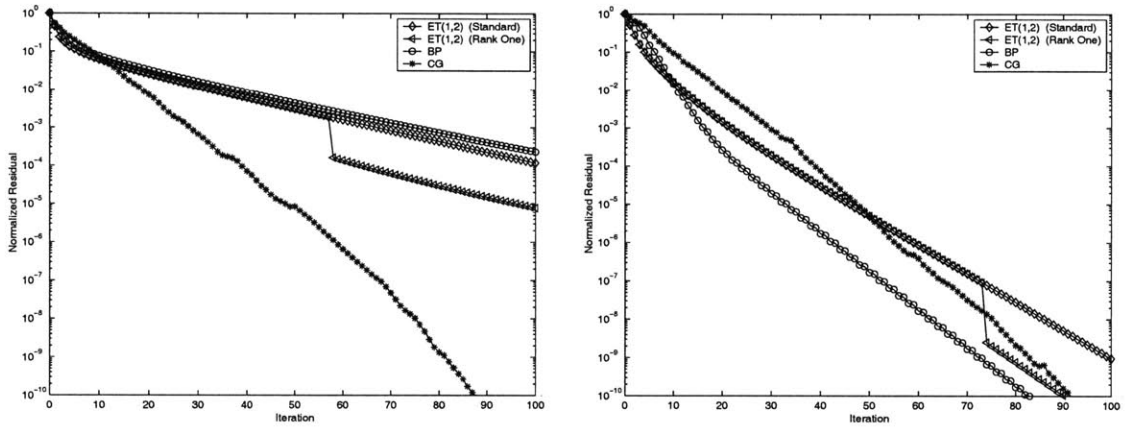
For the disordered multiscale model, acceleration is much less effective. The largest single–tree eigenvalues ($\lambda = -0.7307, 0.7153$) are close enough that a satisfactory estimate of the dominant eigenvector is never attained. In the two–tree case, rank one correction does produce improvements, but the standard ET iteration is already very efficient for this model so the gains are minor. It is likely that on larger augmented multiscale models with mixed potentials, rank one acceleration would be more effective.

Figure 4-4 shows the results of applying the rank one corrections to the 25×25 nearest–neighbor grids of §3.5.3. When applied to the single–tree iteration for any of the grid models, the rank one correction procedure was unable to identify the dominant eigenvector. This behavior is explained by Figure 4-4(a), where we plot the magnitudes of the eigenvalues of the matrix $\mathbf{E}_T$ which determines the ET algorithm's rate of convergence. In the single–tree case, there are many subdominant eigenvalues whose magnitudes nearly equal the spectral radius. Thus, even though the single–tree iterations are quite slow for these problems, they still converge before a single dominant error mode develops.

In the two–tree case, Figure 4-4(a) shows that there are fewer very large subdominant eigenvalues. Thus, as shown in Figure 4-4(b), the rank one correction algorithm does identify a dominant eigenvector. In the homogeneous case, moderate gains are observed, but the resulting convergence rate is still far less than that of CG. For the mixed potential case, the accelerated iteration leads to performance comparable to BP and CG.



(a)



(b)

Figure 4-4: Rank one correction of the ET algorithm applied to a 25×25 nearest–neighbor grid ($\sigma^2 = 10$) with homogeneous potentials (left column) or disordered potentials (right column). (a) Eigenspectrum of the ET convergence matrix $\mathbf{E}_T$ in the single–tree and two–tree cases. (b) Comparison of the rank one correction procedure with other inference algorithms.

4.2.4 Discussion

In the numerical experiments of the previous subsection, the rank one correction procedure proved to be ineffective when applied to single–tree iterations. Even though acceleration was possible in one case, the modified iteration was still far slower than the standard two–tree iteration. In the two–tree case, however, the results were better. In particular, for the multiscale models, noticeable gains were observed, especially in the homogeneous case where the standard two–tree iteration was least effective.

For the nearest–neighbor grids, the gains produced by rank one correction were fairly minor. From Figure 4-4(a), we see that this should be expected, because there

are a few subdominant eigenvalues only slightly smaller than the spectral radius. Note, however, that only a small percentage of the 625 eigenvalues are within even half of the spectral radius. Thus, for these problems, a multiple–rank correction procedure which removed the top 10 or 20 largest error modes would be quite effective. As suggested by Bertsekas [12], if we had available a matrix V whose columns spanned the subspace defining these error modes, we could perform a similar correction using a multidimensional least squares procedure analogous to equation (4.11). The matrix V could be estimated online by successively orthogonalizing blocks of differences between the iterates generated by the ET algorithm. However, the computational cost of such a procedure would grow as the cube of the number of corrected dimensions.

Although a multiple rank correction algorithm would address some of the rank one algorithm's limitations, there is a more fundamental problem with both of these two–phase procedures. Even though information about the principal error direction is available during phase one, it is not used at all to improve the convergence rate. Then, once the switch to phase two is made, the information provided by subsequent iterations is not used to further refine the estimate of the dominant eigenvector. Furthermore, the transition between these two phases is not done in any sort of optimal manner, but is controlled by an arbitrarily chosen threshold parameter.

To improve the rank one algorithm, we would like to incorporate more effectively the information provided by successive iterations of the ET algorithm. From equation (4.12), we see that at the n^{th} subsampled iteration, the set of vectors used by the rank one procedure to estimate the dominant eigenvector is given by

$$\left\{\widehat{x}^{Tk} - \widehat{x}^{T(k-1)}\right\}_{k=1}^{n} = \left\{\mathbf{E}_T^{k-1}(\widehat{x}^T - \widehat{x}^0)\right\}_{k=1}^{n} \tag{4.15}$$

By comparison to equation (2.52), we see that this is precisely the Krylov subspace $\mathcal{K}_n\left(\mathbf{E}_T, \widehat{x}^T - \widehat{x}^0\right)$. Thus, the natural way to make use of this information would be to use a Krylov subspace method (see §2.4.2). In the following section, we show that the embedded trees algorithm can indeed be naturally integrated with Krylov subspace techniques through the concept of preconditioning.

4.3 Spanning Tree Preconditioners

In §3.2.2, we demonstrated that whenever the embedded trees algorithm is implemented by periodically cycling through a fixed set of T cutting matrices, it is equivalent to a preconditioned Richardson iteration. An explicit formula for the implicitly generated preconditioning matrix is given in Theorem 3.3. By simply applying the standard ET iteration equation (3.7) T times, once for each cutting matrix, we can compute the product of the ET preconditioning matrix with any vector in $\mathcal{O}(TNd^3)$ operations. Thus, from the discussion in §2.4.2, we see that the ET iteration can be directly used as a preconditioner for Krylov subspace methods.

In this section, we briefly explore the theoretical and empirical properties of embedded tree–based preconditioners. As discussed in §2.4.2, conjugate gradients (CG) is by far the most widely used Krylov subspace method for positive definite systems. However, for a preconditioning matrix to be used with CG, it must be symmetric. While any single–tree ET iteration leads to a symmetric preconditioner, the preconditioners associated with multiple–tree iterations are in general nonsymmetric. Thus, we begin in §4.3.1 by investigating preconditioners based on a single spanning tree. Then, in §4.3.2, we suggest some methods for creating preconditioners using multiple trees.

4.3.1 Single Tree Preconditioners

In this section, we consider the application of preconditioners based on a single spanning tree, or equivalently on a single iteration of the ET algorithm, to the conjugate gradient method. If the spanning tree is created by cutting matrix $K_\mathcal{T}$, the resulting preconditioned system which CG effectively solves is given by

$$\left(\widehat{J} + K_\mathcal{T}\right)^{-1} \widehat{J}\widehat{x} = \left(\widehat{J} + K_\mathcal{T}\right)^{-1} C^T R^{-1} y \tag{4.16}$$

The convergence rate of the conjugate gradient method is determined by how well the eigenspectrum of the *preconditioned* system $\left(\widehat{J} + K_\mathcal{T}\right)^{-1} \widehat{J}$ can be fit by a polynomial

(see §2.4.2). Effective preconditioners produce smoother, flatter eigenspectra which are accurately fit by a lower order polynomial.

As discussed in §3.1, for graphical models where a relatively small number of edges must be cut to reveal an embedded tree, the cutting matrix $K_{\mathcal{T}}$ will be low rank. The following theorem shows that the rank of $K_{\mathcal{T}}$ has important implications for the convergence of the preconditioned CG method.

Theorem 4.3. Suppose that the conjugate gradient algorithm is used to solve the N–dimensional preconditioned linear system given by equation (4.16). Then if the cutting matrix $K_{\mathcal{T}}$ has $\text{rank}(K_{\mathcal{T}}) = m$, the preconditioned CG method will converge to the exact solution in at most $m + 1$ iterations.

Proof. Let λ be any eigenvalue of $\left(\widehat{J} + K_{\mathcal{T}}\right)^{-1} \widehat{J}$, and v its corresponding eigenvector. We then clearly have

$$\begin{aligned} \left(\widehat{J} + K_{\mathcal{T}}\right)^{-1} \widehat{J}v &= \lambda v \\ \widehat{J}v &= \lambda \left(\widehat{J} + K_{\mathcal{T}}\right) v \\ (1-\lambda)\widehat{J}v &= K_{\mathcal{T}} v \end{aligned} \tag{4.17}$$

Since $\text{rank}(K_{\mathcal{T}}) = m$, there exist $N - m$ linearly independent eigenvectors in the nullspace of $K_{\mathcal{T}}$. Each one of these eigenvectors satisfies equation (4.17) when $\lambda = 1$. Thus, $\lambda = 1$ is an eigenvalue of $\left(\widehat{J} + K_{\mathcal{T}}\right)^{-1} \widehat{J}$, and its multiplicity is at least $N - m$.

Let $\{\lambda_i\}_{i=1}^{m}$ denote the m eigenvalues of $\left(\widehat{J} + K_{\mathcal{T}}\right)^{-1} \widehat{J}$ not constrained to equal one. Consider the $(m+1)^{st}$ order polynomial $p_{m+1}(\lambda)$ defined as

$$p_{m+1}(\lambda) = \frac{(1-\lambda)\prod_{i=1}^{m}(\lambda_i - \lambda)}{\prod_{i=1}^{m}\lambda_i} \tag{4.18}$$

By construction, $p_{m+1}(0) = 1$. Let $A \triangleq \left(\widehat{J} + K_{\mathcal{T}}\right)^{-1} \widehat{J}$. Then, by equation (2.54)

(see [25, p. 313]),

$$\frac{||r^{m+1}||_{A^{-1}}}{||r^0||_{A^{-1}}} \leq \max_{\lambda \in \{\lambda_i(A)\}} |p_{m+1}(\lambda)| \tag{4.19}$$

where r^{m+1} is the residual at iteration $m+1$. Since $p_{m+1}(\bar{\lambda}) = 0$ for all $\bar{\lambda} \in \{\lambda_i(A)\}$, we must have $r^{m+1} = 0$. Therefore, CG must converge by iteration $m+1$. □

Thus, when the cutting matrix rank is smaller than the dimension of $\widehat{J}$, the preconditioned conjugate gradient iteration can be guaranteed to converge in strictly fewer iterations than the unpreconditioned method.

When combined with the results of Chapter 3, Theorem 4.3 has a number of interesting implications. In particular, from §3.3.1 we know that a cutting matrix $K_{\mathcal{T}}$ which cuts E edges from a graph can always be chosen so that $\text{rank}(K_{\mathcal{T}}) \leq \mathcal{O}(Ed)$. Then, if this cutting matrix is used to precondition the CG iteration, we see that the conditional mean can be *exactly* (non–iteratively) calculated in $\mathcal{O}(NEd^4)$ operations. Similarly, if single–tree preconditioned CG is used as the inference method for the fixed point error covariance algorithm of §3.3.4, error covariances can be exactly calculated in $\mathcal{O}(NE^2d^5)$ operations. Note that in problems where E is reasonably large, we typically hope that iterative methods will converge in less than $\mathcal{O}(Ed)$ iterations. Nevertheless, it is useful to be able to provide such guarantees of worst–case computational cost.

To illustrate Theorem 4.3, we applied the single–tree preconditioned conjugate gradient (PCG) iteration to the augmented multiscale model of §3.5.2. The results are shown in Figure 4-5. Recall that only three edges must be cut from this graph to reveal a spanning tree. Thus, a zero–diagonal $K_{\mathcal{T}}$ for this graph will have rank 6, while a positive semidefinite $K_{\mathcal{T}}$ will have rank 3. As predicted by Theorem 4.3, the zero–diagonal and positive semidefinite PCG iterations converge to machine precision after 7 and 4 iterations, respectively. For comparison, we have also plotted the one and two tree Richardson iterations, as well as the unpreconditioned CG method. Note that the single–tree PCG iterations are *dramatically* more effective than either ET(1) or CG are by themselves.

For a 25×25 grid, too many edges must be cut for the bounds of Theorem 4.3 to

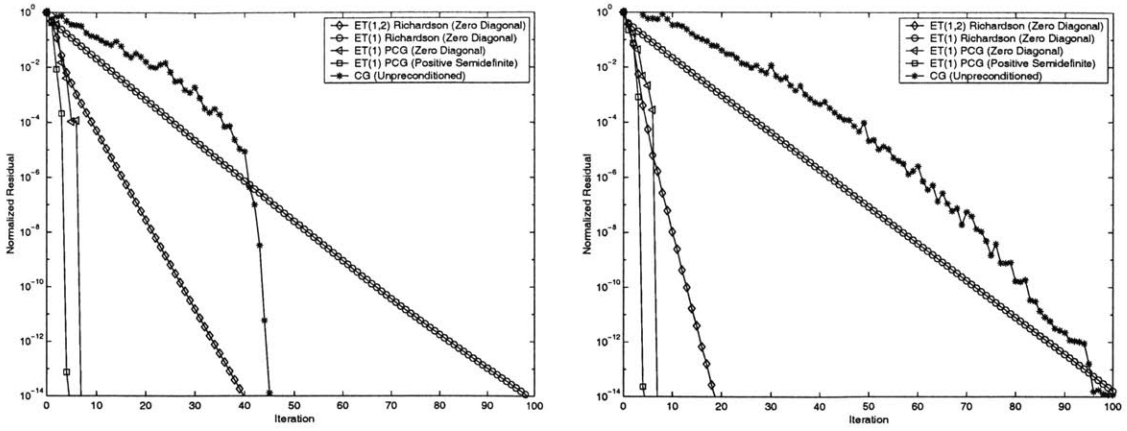
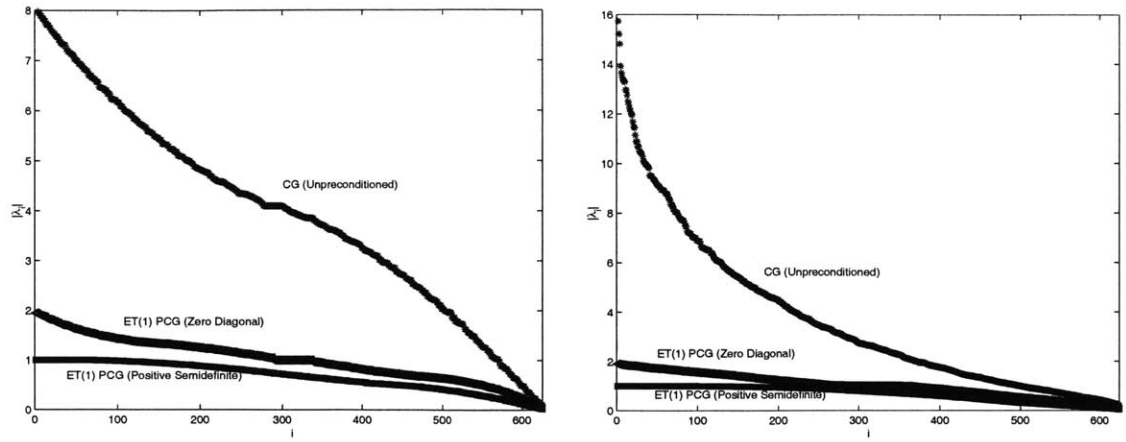


Figure 4-5: Single–tree preconditioning of the CG algorithm applied to the augmented multiscale model of Figure 3-6 ($\sigma^2 = 10$) with homogeneous potentials (left) or disordered potentials (right).

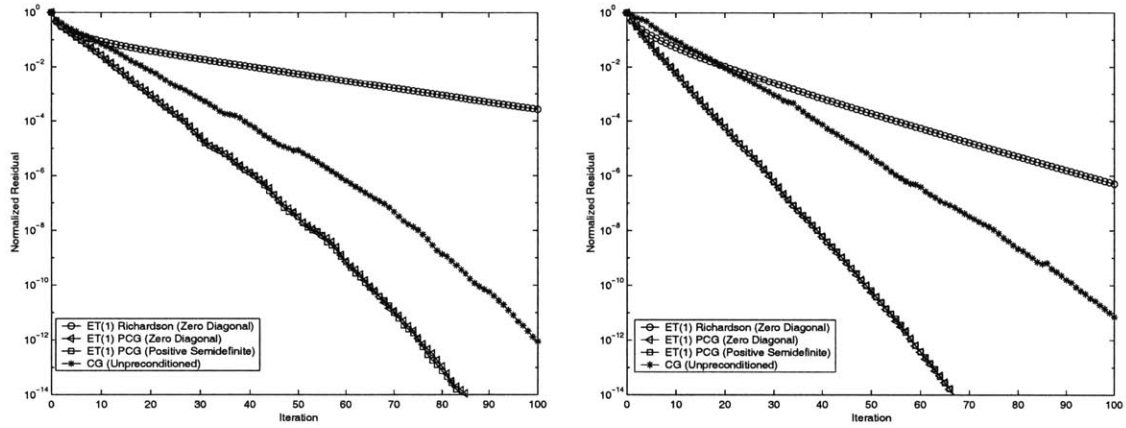
be directly useful. Nevertheless, as shown in Figure 4-6, single–tree preconditioners still perform very well. We again consider single–tree preconditioning using both zero–diagonal and positive semidefinite cutting matrices. Figure 4-6(a) compares the eigenvalues of the preconditioned and unpreconditioned systems. We see that both the tree–based preconditioners significantly flatten the original eigenspectrum. Although the positive semidefinite preconditioner leads to a smaller condition number, the spectrum is no smoother than the zero–diagonal case. This is reflected by their nearly identical performance in Figure 4-6(b). Note that preconditioning is most effective when applied to the more irregular disordered model. For both of these models, the single–tree PCG iterations converge more rapidly than any of the methods considered earlier in this thesis, including BP and the accelerated two–tree ET iterations of §4.1 and §4.2.

4.3.2 Multiple Tree Preconditioners

In this section, we briefly explore the properties of multiple–tree preconditioners. As discussed earlier, the preconditioning matrix corresponding to a multiple–tree ET iteration will in general by nonsymmetric. However, as shown by the following proposition, if the cutting matrices are appropriately chosen, symmetric multi–tree preconditioners can indeed be constructed.



(a)



(b)

Figure 4-6: Single–tree preconditioning of the CG algorithm applied to a 25×25 nearest–neighbor grid ($\sigma^2 = 10$) with homogeneous potentials (left column) or disordered potentials (right column). (a) Eigenspectra of the unpreconditioned and preconditioned linear systems. (b) Comparison of single–tree preconditioning with other inference algorithms.

Proposition 4.4. Suppose that the ET iteration (3.7) is implemented by periodically cycling through a fixed set of T cutting matrices $\{K_{\mathcal{T}_n}\}_{n=1}^{T}$. Assume that the reversal of this sequence of cutting matrices equals the original sequence, i.e. that $K_{\mathcal{T}_n} = K_{\mathcal{T}_{T+1-n}}$ for $n = 1, \ldots, T$. Then the corresponding preconditioning matrix M_T^{-1}, as defined by Theorem 3.3, is symmetric.

Proof. From Theorem 3.3, it is straightforward to show that reversing the cutting matrix order generates a new preconditioning matrix which equals the transpose of the original preconditioner. Thus, if the reversed cutting matrix sequences are equal, the resulting preconditioner must be symmetric. □

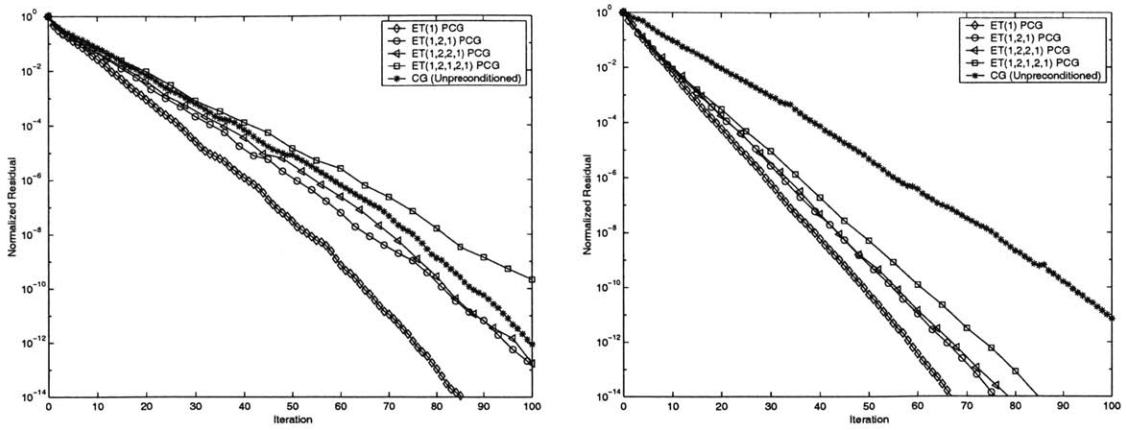
For example, the preconditioners corresponding to the cutting matrix sequences $\{K_{\mathcal{T}_1}, K_{\mathcal{T}_2}, K_{\mathcal{T}_1}\}$, $\{K_{\mathcal{T}_1}, K_{\mathcal{T}_2}, K_{\mathcal{T}_2}, K_{\mathcal{T}_1}\}$, and $\{K_{\mathcal{T}_1}, K_{\mathcal{T}_2}, K_{\mathcal{T}_3}, K_{\mathcal{T}_2}, K_{\mathcal{T}_1}\}$ are all symmetric.

In general, multiple–tree preconditioned systems generated by low rank cutting matrices are *not* guaranteed to converge in a small number of iterations as ensured by Theorem 4.3 in the single–tree case. Thus, for very sparse problems, any convergence rate gains attained through the use of multiple trees are typically more than offset by the dramatic superlinear convergence demonstrated in the previous subsection. In particular, when applied to the augmented multiscale models of §3.5.2, multiple–tree preconditioners converged more slowly than the single–tree case shown in Figure 4-5.

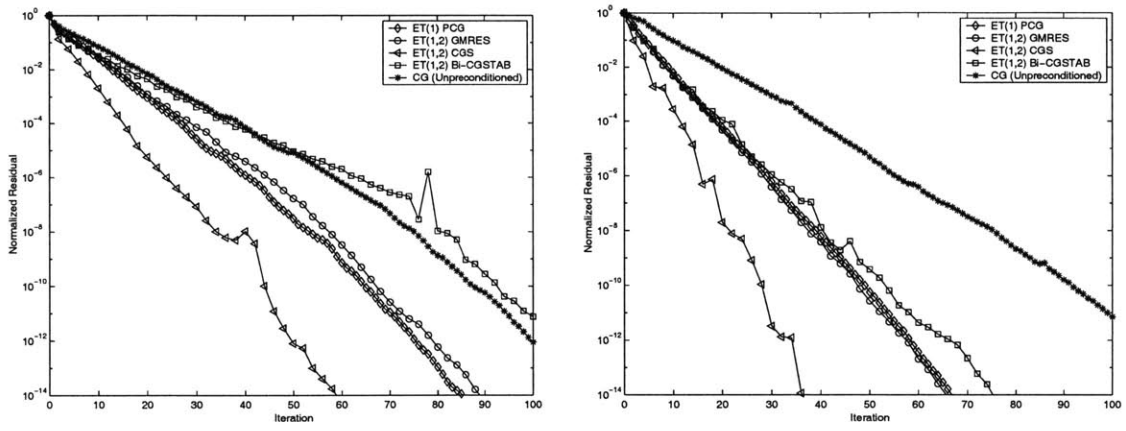
For more densely connected models such as nearest–neighbor grids, the situation is more complicated. Figure 4-7(a) investigates the performance of the symmetric multiple–tree preconditioners suggested by Proposition 4.4 when used to precondition the CG algorithm. The tests were performed on the same high–noise 25×25 grids considered in the previous sections, and zero diagonal cutting matrices were used to generate the preconditioners. The computational cost of applying a multiple–tree preconditioner based on T cutting matrices is $\mathcal{O}(TNd^3)$. Therefore, in order to make the comparison fair, we plot the residuals generated by each multiple–tree preconditioned iteration versus the total number of single–tree iterations (T) which could be performed for the same computational cost.

From the plots in Figure 4-7(a), we see that when scaled for computational cost, none of the symmetric multi–tree preconditioners are as effective as the single–tree preconditioner examined in the previous subsection. In addition, as the number of preconditioning matrices increases, performance generally degrades. This is not surprising, because as the preconditioner length grows the convergence rate must approach that of the two–tree Richardson iteration, which was less effective than single–tree preconditioned CG for these problems.

For the embedded trees Richardson iteration, symmetric preconditioners like those investigated above are much less effective than the alternating two–tree iteration considered in earlier sections of this thesis. Intuitively, after performing an iteration



(a)



(b)

Figure 4-7: Multiple–tree preconditioning applied to a 25×25 nearest–neighbor grid ($\sigma^2 = 10$) with homogeneous potentials (left column) or disordered potentials (right column). (a) CG preconditioned by different symmetric multiple–tree preconditioners. (b) Krylov subspace methods for nonsymmetric matrices preconditioned by two trees.

with some tree, the best error cancellation occurs when the subsequent iteration uses a different tree. Thus, it seems likely that we could obtain better performance if we could find a way to use the nonsymmetric two–tree preconditioner which provided the most effective Richardson iteration.

Although CG cannot be preconditioned with a nonsymmetric matrix, there exist other Krylov subspace methods which can be. Unfortunately, for reasons beyond the scope of this discussion, there does not exist a nonsymmetric Krylov subspace method which shares CG's combination of performance and efficiency. Some methods, like generalized minimum residual (GMRES), minimize an explicit error metric at each

iteration, but the cost of the k^{th} iteration is $\mathcal{O}(Nk)$, which is much worse than CG's $\mathcal{O}(N)$. Others, such as conjugate gradient squared (CGS) and BiConjugate Gradient Stabilized (Bi-CGSTAB), retain CG's $\mathcal{O}(N)$ cost, but do not explicitly minimize any error metric and may diverge from the correct solution. For more information on these and other nonsymmetric Krylov subspace methods, see [7, 37].

Figure 4-7(b) shows the results of applying the three non–symmetric methods mentioned in the previous paragraph to the two–tree preconditioned system. Bi-CGSTAB and GMRES are both inferior to the single–tree preconditioner. However, the two–tree preconditioned CGS iteration does converge noticeably faster than the single–tree preconditioned CG, albeit somewhat more irregularly. Certainly, far more extensive studies are needed to determine whether such nonsymmetric methods can be stable and effective enough for general use with symmetric linear systems like the estimation problems considered in this thesis.

4.3.3 Discussion

For very sparsely connected graphical models like the augmented multiscale model, the single–tree preconditioned conjugate gradient iteration is by far the most effective inference procedure presented in this thesis. For such small models, the benefits of the regular eigenstructure guaranteed by Theorem 4.3 outweigh the gains normally provided by multiple–tree iterations. However, it is not clear which techniques will be most effective for graphs where E is much smaller than N, but still quite large. If a multiple–tree based method can consistently converge in less than $\mathcal{O}(E)$ iterations, it may still provide an attractive alternative to the guarantees of the single–tree preconditioned conjugate gradient iteration.

The experimental results concerning multiple tree preconditioners raise a number of interesting questions. In particular, given the huge gains which multiple–tree ET iterations consistently provide relative to single–tree ET iterations, it is somewhat surprising that multiple–tree preconditioners were generally slightly less effective than their single–tree counterparts. One possibility is that since CG is much more sophisticated than a basic Richardson iteration, the extra spectral smoothing provided by

multiple tree preconditioners is not justified by their additional computational cost. However, a more interesting alternative is that we are simply not making a sufficiently intelligent choice for the spanning trees. The spanning trees used in our experiments were chosen heuristically by attempting to separate cut edges widely within each tree, while at the same time minimizing overlap of cut edges between successive trees. However, they were not at all optimized for the potentials of the underlying graph. For very inhomogeneous models in particular, such optimizations could potentially prove quite effective.

In the search for principled ways to construct good multiple–tree preconditioners, single–tree preconditioning theory may provide a useful starting point. In particular, several authors have recently rediscovered [8], analyzed [19], and extended [16, 17] a technique called support graph theory which provides powerful methods for constructing effective preconditioners from maximum–weight spanning trees. Support graph theory is especially interesting because it provides a set of results guaranteeing the effectiveness of the resulting preconditioners. However, extending support theory to the multiple–tree case would almost certainly require non–trivial graph theoretic developments, and is beyond the scope of this thesis.

Chapter 5

Recommendations and Conclusions

This thesis has proposed and analyzed a family of tree–based algorithms for iteratively solving Gaussian inference problems defined by graphs with cycles. The previous chapters have presented a variety of theoretical and empirical results which demonstrate the effectiveness of these methods. In this chapter, we begin by highlighting the most significant of these contributions. We then discuss a variety of open research problems naturally motivated by our results. These open questions involve not only the design and analysis of more efficient inference algorithms, but also the investigation of an interesting new class of graphical models.

5.1 Contributions

In the Introduction to this thesis, we presented a multiscale modeling example which suggested that very sparsely connected graphs with cycles may offer significant modeling advantages relative to their tree–structured counterparts. The primary goal of the remainder of this thesis has been to develop inference algorithms which could efficiently and accurately solve inference problems defined on such graphs. In the process, we have developed several techniques which yield good results when applied to densely connected graphical models. However, the proposed algorithms are most effective when a spanning tree may be revealed by removing a relatively small number of edges.

5.1.1 Embedded Trees Estimation Algorithm

The basis for most of the results in this thesis is an embedded trees (ET) algorithm which iteratively calculates the conditional mean of a Gaussian inference problem defined by a graph with cycles. Each step of the ET algorithm involves the removal of a subset of the full graph's edges to reveal a spanning tree. An exact inference procedure is performed on this spanning tree, the results of which provide a starting point for the subsequent iteration's calculations. Intuitively, the ET algorithm is most effective when different spanning trees are used at each iteration, so that constraints neglected in one iteration are directly enforced in the following iteration's calculation.

We have presented a variety of theoretical results analyzing the convergence behavior of this algorithm. These results provide a set of easily verifiable sufficient conditions which can often be used to verify the algorithm's convergence. In addition, they partially explain the dramatic empirical performance gains obtained through the use of multiple spanning trees. The convergence results are complemented by a geometric analysis which reveals the structure underlying the ET iterations. This structure is particularly interesting in the context of the sparsely connected models motivated in the introduction, as it shows that this sparsity places very specific constraints on the errors made at each iteration.

In addition to these theoretical results, the ET algorithm's performance was tested on an representative set of synthetic problems. These experiments showed that on sparsely connected graphs with cycles, the ET algorithm is extremely effective in comparison with other standard techniques such as loopy belief propagation and conjugate gradient. On more densely connected graphs, its performance is typically comparable to other methods.

5.1.2 Embedded Trees Error Covariance Algorithm

In addition to the ET algorithm for calculating conditional means, this thesis has proposed and analyzed a tree–based algorithm for exact calculation of error variances. This algorithm depends on the fact that if a graphical model is nearly tree–structured,

its error variances will be closely related to a set of easily calculated tree–based error variances. To calculate the appropriate perturbation of these tree variances, the algorithm iteratively solves a set of carefully chosen synthetic estimation problems. These problems may be solved using the ET algorithm for the calculation of conditional means. This in turn allows most of the ET algorithm's theoretical analyses to be applied directly to the error covariance method.

The error covariance method has been evaluated on the same problems used to test the ET algorithm. For sparsely connected graphs, the error covariance method provides significant gains over other techniques in terms of both accuracy and computational cost. For more densely connected models, however, the computational cost of calculating the resulting high–dimensional perturbation is in general intractable.

5.1.3 Accelerated Tree–Based Iterations

Although the ET iteration is very effective for many problems, its convergence rate is not always competitive with other techniques. Thus, this thesis has proposed and analyzed three methods for constructing more rapidly convergent tree–based iterations. The first method is a diagonal acceleration technique which is based on a simple modification of the numerical structure underlying the tree–based inference problems solved at each iteration. This acceleration procedure depends on a single parameter. Although optimally setting this parameter is difficult, our simulations indicate that it is relatively simple to numerically determine a value which leads to good performance. For the most difficult test problems, diagonal acceleration was shown to provide modest performance gains with no additional computational cost.

The second acceleration procedure perturbs each iteration of the standard ET algorithm by a rank one correction designed to cancel the most significant errors in the current estimate. The derivation and analysis of this algorithm relies heavily on earlier results in this thesis concerning the geometry of the ET iteration. Unfortunately, because this algorithm can only cancel a one–dimensional subspace of the current errors, it is not effective for many test problems. Although higher dimensional variants are possible, the computational cost grows quickly with the dimension of the subspace

in which errors are to be cancelled.

Motivated by an analysis of the rank one correction procedure's weaknesses, we conclude the thesis with a study of tree–based preconditioners for Krylov subspace methods. When a single tree is used as a preconditioner for a sparsely connected graph, we provide a result which guarantees that the preconditioned iteration will quickly converge to the exact answer. We present a set of simulations which show that even for more densely connected graphs, single–tree preconditioners often perform extremely well. In contrast with the standard ET algorithm, however, preconditioners based on multiple trees do not appear to provide significant advantages.

5.2 Extensions

There are a wide variety of open research problems naturally suggested by the results of this thesis. Some involve the development of new and more efficient inference algorithms, while others explore the class of augmented multiscale models suggested by the introductory example. We present some of the most interesting questions below, organized by topic.

5.2.1 Inference Algorithms for Graphs with Cycles

Structural Optimization of Embedded Trees

For all of the experimental results in this thesis, the spanning trees were chosen heuristically by attempting to separate cut edges widely within each tree, while at the same time minimizing overlap of cut edges between successive trees. As demonstrated by the effectiveness of many of the resulting ET iterations, these heuristics have some merit. However, for inhomogeneous problems in particular, it seems certain that improved performance could be attained by matching the spanning trees to the potentials underlying the graph.

One natural idea for finding a good tree is to use a maximum weight spanning tree algorithm. However, in this case it is not entirely clear what weight should be assigned

to each edge. One option is the magnitude of the off–diagonal entry being cut; another is the conditional correlation coefficient associated with that edge (equation (2.12)). More generally, however, the experimental results show that the interactions between multiple trees are extremely important, and would need to be handled by any effective tree selection procedure. This could potentially be accomplished by greedily selecting one max–weight tree, and then penalizing the second tree for discarding the same edges. However, the development of a reasonable means for assigning such penalties is an open problem.

In understanding the relationships between multiple spanning trees, the decomposition provided by Theorem 3.6 may be of some use. Letting $K_{\mathcal{T}_n} = U_n D_n U_n^T$ be the eigendecomposition of the n^{th} cutting matrix, the ET algorithm's convergence rate is determined by the eigenvalues of

$$\mathbf{E}_r = \prod_{n=1}^{T} U_{n+1}^T \widehat{P} U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n \right)^{-1}$$

The terms of this equation have intuitive interpretations. The $\left(D_n^{-1} + U_n^T \widehat{P} U_n \right)^{-1}$ term seems to measure the strength of the discarded edges relative to the loopy graph distribution $\widehat{P}$. To a first approximation, this term will be made small when D_n is small, or equivalently when the cut edges are weak. However, the $U_n^T \widehat{P} U_n$ term shows that there may be important interactions between discarded edges. Similarly, the $U_{n+1}^T \widehat{P} U_n$ term measures the interactions between subsequent spanning trees. It will be smallest when $K_{\mathcal{T}_n}$ and $K_{\mathcal{T}_{n+1}}$ cut edges which are connected to weakly correlated nodes.

It seems likely that the terms in the preceding discussion could be adapted to determine good edge weights for use in the tree optimization procedure. The main problem is in determining how to deal with the entries of $\widehat{P}$, which are not explicitly available, and are often the variables we would most like to calculate.

Adaptive Error Covariance Calculation

In the examples employing our error covariance method in this thesis, we assumed that a spanning tree was used to provide the base structure from which perturbations are calculated. However, exact error covariance calculation is also possible for more general graphs, as long as the maximal cliques in the triangulated graph are not overly large. One interesting exact inference algorithm would be an adaptive triangulation procedure which throws away the "worst" edges which lead to the largest growth in clique size. The effects of these worst edges could then be exactly calculated using the techniques of this thesis, potentially with much lower total cost than if the graph was pruned all the way to a tree. However, the means by which these problem edges could be identified is an open problem.

Inference Algorithms for Non–Gaussian Graphical Models

While this thesis has focused on Gaussian graphical models, it seems likely that tree–based procedures could also lead to efficient inference algorithms for non–Gaussian models. In related work inspired by the ET algorithm, a set of powerful tree–based reparameterization algorithms [76, 77] for the inference of discrete–valued processes have been proposed and analyzed. In addition, the Gaussian inference procedures of this thesis may provide a starting point for solving continuous, non–Gaussian inference problems on graphs with cycles.

5.2.2 Multiscale Modeling using Graphs with Cycles

The example in the Introduction of this thesis suggested that adding small numbers of edges to multiscale tree–structured models may greatly increase their effectiveness, in particular alleviating the commonly encountered boundary artifact problem. The inference algorithms developed in this thesis demonstrate that it is indeed possible to perform efficient, exact inference on such augmented multiscale models. However, these inference algorithms have also raised some new concerns about this class of models. In particular, our results demonstrate that it is critical to determine how

quickly the number of "extra" edges E must grow as a function of the number of fine scale nodes N. If effective models can be constructed with $E = \mathcal{O}(\log N)$ edges, this may be an extremely powerful model class. However, if $E = \mathcal{O}(N)$, exact inference, in particular the calculation of error variances, may be intractable.

More generally, there are a number of interesting questions associated with the realization of graphical models with cycles. For example, existing multiscale model stochastic realization theory [31] explicitly depends on the fact that the primary probabilistic function of each coarse scale node is to decorrelate the subtrees it separates. For graphs with cycles, however, coarse scale nodes may not decorrelate anything by themselves, but need to fulfill several different decorrelation roles in combination with their neighbors. Determining the hidden graphical structure which best balances the goals of model accuracy and computational efficiency is a quite interesting, and quite poorly understood, open problem.

Appendix A

Linear Algebraic Identities

A.1 Inversion of Partitioned Matrices

When working with Gaussian random variables, it is often necessary to calculate the inverse of a partitioned matrix. Consider the 2×2 block matrix

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{A.1}$$

which is assumed to be invertible. Assuming that the matrices A and D are square and invertible, the following equations can be verified by direct calculation:

$$M^{-1} = \begin{bmatrix} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1} BD^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1} + D^{-1}C(A - BD^{-1}C)^{-1} BD^{-1} \end{bmatrix} \tag{A.2}$$

$$= \begin{bmatrix} A^{-1} + A^{-1}B(D - CA^{-1}B)^{-1} CA^{-1} & -A^{-1}B(D - CA^{-1}B)^{-1} \\ -(D - CA^{-1}B)^{-1} CA^{-1} & (D - CA^{-1}B)^{-1} \end{bmatrix} \tag{A.3}$$

By equating entries in equations (A.2) and (A.3), we may also derive the following expressions:

$$\left(A - BD^{-1}C\right)^{-1} = A^{-1} + A^{-1}B\left(D - CA^{-1}B\right)^{-1} CA^{-1} \tag{A.4}$$

$$\left(A - BD^{-1}C\right)^{-1} BD^{-1} = A^{-1}B\left(D - CA^{-1}B\right)^{-1} \tag{A.5}$$

Equations (A.4) and (A.5) are often useful because they allow expressions involving the inverse of A to be converted into expressions involving the inverse of D, and vice versa. Equation (A.4) is sometimes called the *matrix inversion lemma* [38, 44].

A.2 Eigenvalue Identities

When analyzing the convergence of many iterative algorithms, it is necessary to determine whether the spectral radius of a particular matrix is less than one. The following theorem provides a set of equivalent, necessary and sufficient conditions which are often easier to evaluate.

Theorem A.1. Let A be a real, square matrix. Then $\rho(A) < 1$ if and only if any one of the following conditions holds:

1. There exists a real positive definite matrix G such that

$$G - A^T G A > 0 \tag{A.6}$$

2. There exists a real positive definite matrix R such that

$$\sigma_{\max}(RAR^{-1}) < 1 \tag{A.7}$$

3. $(I - A)$ is nonsingular, and the matrix H defined by

$$H = (I - A)^{-1}(I + A) \tag{A.8}$$

has eigenvalues with positive real parts. This will hold if and only if there exists a real positive definite matrix Q such that

$$HQ + QH^T > 0 \tag{A.9}$$

Proof. The first condition is originally due to Stein [70]. Proofs of the remaining

conditions can be found in the text by Young [85, pp. 80–84]. □

Note that the conditions of Theorem A.1 are very closely related to techniques which arise in the analysis of Riccati equations associated with Kalman filtering algorithms for linear state space models [44].

Appendix B

Gaussian Random Vectors

B.1 Information Parameters

Let $x \sim \mathcal{N}(\mu, \mathcal{P})$ be an N-dimensional Gaussian random vector with mean $\mu \triangleq E[x]$ and positive definite covariance $\mathcal{P} \triangleq E\left[(x-\mu)(x-\mu)^T\right]$. The probability distribution $p(x)$ of x, as a function of the *moment* parameters μ and $\mathcal{P}$, is given by

$$p(x) = \frac{1}{\sqrt{(2\pi)^N \det \mathcal{P}}} \exp\left\{-\frac{1}{2}(x-\mu)^T \mathcal{P}^{-1}(x-\mu)\right\} \tag{B.1}$$

Although the mean μ and covariance $\mathcal{P}$ provide the most common parameterization of a Gaussian random vector, they are not the only possibility. In particular, one can define a set of *information* parameters (ϑ, Λ) by

$$\vartheta = \mathcal{P}^{-1}\mu \qquad \Lambda = \mathcal{P}^{-1} \tag{B.2}$$

Note that the Gaussian information parameterization is a special case of the exponential parameterization commonly used in the information geometry literature [4]. Using these information parameters, $p(x)$ may be rewritten as

$$p(x) = \alpha \exp\left\{\vartheta^T x - \frac{1}{2}x^T \Lambda x\right\} \tag{B.3}$$

where α is a normalization constant. We use the notation $x \sim \mathcal{N}^{-1}(\vartheta, \Lambda)$ to indicate that x is a Gaussian random vector with information parameters ϑ and Λ.

In principle, equation (B.2) may be used to convert between the moment and information parameterizations at will. In many situations, however, the information parameters provide a more natural form for performing calculations. In the following subsections, we provide a set of formulas for performing various common statistical operations on Gaussian densities parameterized in the information form (B.3).

Conditional Densities

Let x_1 and x_2 be two jointly Gaussian random vectors with distribution $p(x_1, x_2) = \mathcal{N}^{-1}\left(\begin{bmatrix}\vartheta_1 \\ \vartheta_2\end{bmatrix}, \begin{bmatrix}\Lambda_{11} & \Lambda_{12} \\ \Lambda_{21} & \Lambda_{22}\end{bmatrix}\right)$. Then the conditional distribution $p(x_1 \mid x_2) = \mathcal{N}^{-1}(\vartheta_{1|2}, \Lambda_{1|2})$ is also Gaussian, with information parameters given by

$$\vartheta_{1|2} = \vartheta_1 - \Lambda_{12} x_2 \tag{B.4}$$

$$\Lambda_{1|2} = \Lambda_{11} \tag{B.5}$$

These equations may be derived by transforming the moment parameter conditional distribution equations (2.1, 2.2) using the formulas for the inversion of partitioned matrices given in Appendix A.1.

Marginal Densities

Let x_1 and x_2 be two jointly Gaussian random vectors with distribution $p(x_1, x_2) = \mathcal{N}^{-1}\left(\begin{bmatrix}\vartheta_1 \\ \vartheta_2\end{bmatrix}, \begin{bmatrix}\Lambda_{11} & \Lambda_{12} \\ \Lambda_{21} & \Lambda_{22}\end{bmatrix}\right)$. Then the marginal distribution $p(x_1) = \mathcal{N}^{-1}(\vartheta_1^m, \Lambda_1^m)$ is also Gaussian, with information parameters given by

$$\vartheta_1^m = \vartheta_1 - \Lambda_{12}\Lambda_{22}^{-1}\vartheta_2 \tag{B.6}$$

$$\Lambda_1^m = \Lambda_{11} - \Lambda_{12}\Lambda_{22}^{-1}\Lambda_{21} \tag{B.7}$$

These equations may be derived by using the partitioned matrix inversion formulas (A.2, A.3) to express μ and $\mathcal{P}$ in terms of the joint information parameters.

Product Densities

Let $p_1(x) = \mathcal{N}^{-1}(\vartheta_1, \Lambda_1)$ and $p_2(x) = \mathcal{N}^{-1}(\vartheta_2, \Lambda_2)$ be two different distributions on the same random Gaussian random vector x, and consider the product density

$$p_{12}(x) = \alpha p_1(x) p_2(x)$$

A product density of this form arises, for example, when conditional distributions representing two independent observations of x are combined. From equation (B.3), we see immediately that $p_{12}(x) = \mathcal{N}^{-1}(\vartheta_{12}, \Lambda_{12})$ is also Gaussian, with information parameters given by

$$\vartheta_{12} = \vartheta_1 + \vartheta_2 \tag{B.8}$$

$$\Lambda_{12} = \Lambda_1 + \Lambda_2 \tag{B.9}$$

Similarly, the quotient $\frac{p_1(x)}{p_2(x)}$ produces an exponentiated quadratic form with parameters $(\vartheta_1 - \vartheta_2, \Lambda_1 - \Lambda_2)$. However, this quotient will define a valid (normalizable) probability density only if $(\Lambda_1 - \Lambda_2)$ is positive definite.

B.2 Canonical Correlations

In this section, we briefly introduce *canonical correlations* analysis as a general method for understanding the joint statistics of a pair of Gaussian random vectors. In the case of scalar random variables, canonical correlations coincides with the standard correlation coefficient. In the following three sections, we first show how the canonical correlation coefficients may be calculated using the singular value decomposition. Then, we discuss some of the properties that make them useful, including their connection to mutual information. Finally, we provide a proof of Proposition 2.1, which shows the important role that canonical correlations plays in the interpretation of the parameters of Gaussian Markov random fields.

B.2.1 Calculation of Canonical Correlation Coefficients

Canonical correlations is a general multivariate statistical analysis technique originally developed by Hotelling [39] in 1936. However, its important connections to the singular value decomposition, which provides a numerically stable method for calculating the canonical coefficients and vectors, were not made until 1973 by Björck and Golub [15, 37]. For an interesting tutorial introduction to canonical correlations, emphasizing connections to Wiener filtering and communications theory, see [63, 64].

Let $x = [x_1 \dots x_m]^T \in \mathbb{R}^m$ and $y = [y_1 \dots y_n]^T \in \mathbb{R}^n$ be two jointly Gaussian random vectors with distribution $p(x,y) = \mathcal{N}\left(\begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}\mathcal{P}_x & \mathcal{P}_{xy}\\ \mathcal{P}_{xy}^T & \mathcal{P}_y\end{bmatrix}\right)$, where we have assumed for convenience that x and y are zero mean.[1] We may view the random variables x_i and y_j as elements of a vector space $\mathcal{H}$ with associated inner product $\langle x_i, y_j \rangle \triangleq E[x_i y_j]$. The random vectors x and y then define subspaces $\mathcal{H}_x$ and $\mathcal{H}_y$ of $\mathcal{H}$, each spanned by linear combinations of the corresponding scalar random variables.

Let $\tilde{x} = \mathcal{P}_x^{-\frac{1}{2}} x$ and $\tilde{y} = \mathcal{P}_y^{-\frac{1}{2}} y$ be the *whitened* random vectors corresponding to x and y, so that $E\left[\tilde{x}\tilde{x}^T\right] = I_m$ and $E\left[\tilde{y}\tilde{y}^T\right] = I_n$. Geometrically, $\tilde{x}$ and $\tilde{y}$ define *orthonormal* bases for the subspaces $\mathcal{H}_x$ and $\mathcal{H}_y$. The covariance of $\tilde{x}$ and $\tilde{y}$ is given by $\mathcal{P}_{\tilde{x}\tilde{y}} = \mathcal{P}_x^{-\frac{1}{2}} \mathcal{P}_{xy} \mathcal{P}_y^{-\frac{T}{2}}$. Let the *singular value decomposition* [37] of $\mathcal{P}_{\tilde{x}\tilde{y}}$ be given by

$$\mathcal{P}_{\tilde{x}\tilde{y}} = \mathcal{P}_x^{-\frac{1}{2}} \mathcal{P}_{xy} \mathcal{P}_y^{-\frac{T}{2}} = URV^T \tag{B.10}$$

where $U^T U = I_m$, $V^T V = I_n$, and R is an $m \times n$ matrix whose entries are all zero except for the principal diagonal, which contains the $p = \min(m,n)$ singular values $\{\sigma_i(\mathcal{P}_{\tilde{x}\tilde{y}})\} \triangleq \{r_i\}_{i=1}^p$. We assume the singular values are ordered so that $r_1 \geq r_2 \geq \cdots \geq r_p \geq 0$, where $r_1 \leq 1$ as we demonstrate in the following section. The singular vectors U and V may then be used to define a set of *canonical* random variables $\bar{x} = U^T \tilde{x}$ and $\bar{y} = V^T \tilde{y}$ with covariance

$$\mathcal{P}_{\bar{x}\bar{y}} = U^T \mathcal{P}_{\tilde{x}\tilde{y}} V = U^T U R V^T V = R \tag{B.11}$$

[1] The canonical correlation coefficients of two random vectors x and y depend only on their covariance $\begin{bmatrix}\mathcal{P}_x & \mathcal{P}_{xy}\\ \mathcal{P}_{xy}^T & \mathcal{P}_y\end{bmatrix}$, not on their means $E[x]$ and $E[y]$.

We see that the canonical vectors $\bar{x}$ and $\bar{y}$ are aligned so that $E[\bar{x}_i \bar{y}_j] = 0$ for $i \neq j$. The singular values $r_i = E[\bar{x}_i \bar{y}_i]$ are known as the *canonical correlation coefficients* of the random vectors x and y.

The preceding construction has also implicitly defined a canonical decomposition of the covariance matrix $\begin{bmatrix} \mathcal{P}_x & \mathcal{P}_{xy} \\ \mathcal{P}_{xy}^T & \mathcal{P}_y \end{bmatrix}$. In particular, if we let $T_x = U^T \mathcal{P}_x^{-\frac{1}{2}}$ and $T_y = V^T \mathcal{P}_y^{-\frac{1}{2}}$, we have

$$\begin{bmatrix} T_x & 0 \\ 0 & T_y \end{bmatrix} \begin{bmatrix} \mathcal{P}_x & \mathcal{P}_{xy} \\ \mathcal{P}_{xy}^T & \mathcal{P}_y \end{bmatrix} \begin{bmatrix} T_x & 0 \\ 0 & T_y \end{bmatrix}^T = \begin{bmatrix} I_m & R \\ R^T & I_n \end{bmatrix} \tag{B.12}$$

The matrices T_x, T_y, and R are sometimes called the *canonical correlation matrices.* The purpose of the singular value decomposition step (B.10) is to choose, out of the many possible square root matrices $\mathcal{P}_x^{\frac{1}{2}}$ and $\mathcal{P}_y^{\frac{1}{2}}$, a pair of square roots T_x and T_y which produce the canonical alignment (B.11). Interestingly, such a pair always exists.

In many applications of canonical correlations analysis, one is interested in the properties of conditional probability distributions. Suppose that x, y, and z are three jointly Gaussian random vectors, and that $\mathrm{cov}(x, y | z) = \begin{bmatrix} \mathcal{P}_{x|z} & \mathcal{P}_{xy|z} \\ \mathcal{P}_{xy|z}^T & \mathcal{P}_{y|z} \end{bmatrix}$. Then the *conditional* canonical correlation coefficients of x and y, conditioned on z, are given by $\{r_i\} = \left\{\sigma_i \left(\mathcal{P}_{x|z}^{-\frac{1}{2}} \mathcal{P}_{xy|z} \mathcal{P}_{y|z}^{-\frac{T}{2}}\right)\right\}$. All of the properties discussed in the following section apply equally well to both conditional and unconditional canonical correlation coefficients.

B.2.2 Interpretation of Canonical Correlation Coefficients

The canonical correlation coefficients may be interpreted both statistically and geometrically. Both interpretations depend on a variational characterization [37] of the singular value decomposition. Let $U = [u_1 \cdots u_m]$ and $V = [v_1 \cdots v_n]$ denote the singular vectors associated with the decomposition (B.10). The singular values $\{r_i\}_{i=1}^p$

may then be found using the following sequential optimization procedure:

$$\begin{aligned} r_i &= \max_{\substack{u_i^T u_j = 0,\ v_i^T v_j = 0 \\ 1 \le j \le i-1}} \frac{u_i^T \mathcal{P}_x^{-\frac{1}{2}} \mathcal{P}_{xy} \mathcal{P}_y^{-\frac{T}{2}} v_i}{\left(u_i^T u_i\right)^{1/2} \left(v_i^T v_i\right)^{1/2}} \\ &= \max_{\substack{u_i^T \mathcal{P}_x u_j = 0,\ v_i^T \mathcal{P}_y v_j = 0 \\ 1 \le j \le i-1}} \frac{u_i^T \mathcal{P}_{xy} v_i}{\left(u_i^T \mathcal{P}_x u_i\right)^{1/2} \left(v_i^T \mathcal{P}_y v_i\right)^{1/2}} \end{aligned} \tag{B.13}$$

Using the definition of the expectation operator, we may rewrite equation (B.13) as

$$\begin{aligned} r_i &= \max_{\substack{E\left[(u_i^T x)(u_j^T x)\right] = 0,\ E\left[(v_i^T y)(v_j^T y)\right] = 0 \\ 1 \le j \le i-1}} \frac{E\left[(u_i^T x)(v_i^T y)\right]}{E\left[(u_i^T x)^2\right]^{1/2} E\left[(v_i^T y)^2\right]^{1/2}} \\ &= \max_{\substack{E\left[(u_i^T x)(u_j^T x)\right] = 0,\ E\left[(v_i^T y)(v_j^T y)\right] = 0 \\ 1 \le j \le i-1}} r\left(u_i^T x, v_i^T y\right) \end{aligned} \tag{B.14}$$

where $r\left(u_i^T x, v_i^T y\right)$ is the standard correlation coefficient between the scalar random variables $u_i^T x$ and $v_i^T y$. Thus, the first canonical correlation coefficient r_1 is equal to the largest possible correlation coefficient between two scalar random variables ($u_1^T x$ and $v_1^T y$) formed from linear combinations of the random vectors x and y. The next canonical correlation coefficient r_2 is the largest possible correlation coefficient between scalar random variables $u_2^T x$ and $v_2^T y$, subject to the additional constraint that $u_2^T x$ and $v_2^T y$ are independent of $u_1^T x$ and $v_1^T y$, respectively. Subsequent canonical correlation coefficients are defined similarly.

The canonical correlation coefficients have an important geometric interpretation which complements their statistical association with the scalar correlation coefficient. In particular, using the inner product $\langle x_i, y_j \rangle = E\left[x_i y_j\right]$ defined earlier, we may rewrite equation (B.14) as

$$r_i = \max_{\substack{\langle u_i^T x, u_j^T x \rangle = 0,\ \langle v_i^T y, v_j^T y \rangle = 0 \\ 1 \le j \le i-1}} \frac{\langle u_i^T x, v_i^T y \rangle}{\langle u_i^T x, u_i^T x \rangle^{1/2} \langle v_i^T y, v_i^T y \rangle^{1/2}} = \cos \alpha_i \tag{B.15}$$

Here, α_i is the angle between $u_i^T x$ and $v_i^T y$, when viewed as elements of an inner product space. The angles $\{\alpha_i\}_{i=1}^p$ are known as the *principal angles* [15, 37] between

the subspaces $\mathcal{H}_x$ and $\mathcal{H}_y$. From the variational characterization, it follows that α_1 is the smallest possible angle between one-dimensional subspaces of $\mathcal{H}_x$ and $\mathcal{H}_y$. Similarly, α_2 is the angle between the most closely aligned subspaces which are also orthogonal to those subspaces defining α_1. Thus, the canonical correlation coefficients depend only on the relative orientation of the subspaces defined by the random vectors x and y, not on the magnitude of their variation within those subspaces.

In addition to their important geometric properties, canonical correlation coefficients are also deeply connected to the *mutual information* [22] between the random vectors x and y, defined as

$$I(x;y) = \int\int p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)}\, dx\, dy \tag{B.16}$$

For Gaussian probability densities (B.1), it is straightforward to show that the mutual information is given by [35]

$$I(x;y) = -\frac{1}{2} \log \left[\frac{\det \begin{bmatrix} \mathcal{P}_x & \mathcal{P}_{xy} \\ \mathcal{P}_{xy}^T & \mathcal{P}_y \end{bmatrix}}{\det \mathcal{P}_x \det \mathcal{P}_y} \right] \tag{B.17}$$

If x and y are scalar Gaussian random variables, equation (B.17) may be simply expressed using the correlation coefficient $r(x,y)$:

$$I(x;y) = -\frac{1}{2} \log \left[1 - r^2(x,y)\right] \qquad r(x,y) = \frac{\operatorname{cov}(x,y)}{\sqrt{\operatorname{var}(x) \operatorname{var}(y)}} \tag{B.18}$$

Using the basic properties of mutual information [22], it can be shown that because the transformations T_x and T_y defined in §B.2.1 are invertible, $I(x;y) = I(T_x x; T_y y)$. In addition, from equation (B.11) we know that $T_x x$ and $T_y y$ define p independent pairs of random variables such that $r((T_x x)_i, (T_y y)_i) = r_i$. The mutual information of independent pairs of variables is simply the sum of their individual mutual informations:

$$I(x;y) = -\frac{1}{2} \sum_{i=1}^{p} \log \left[1 - r_i^2\right] \tag{B.19}$$

Thus, the canonical correlation coefficients $\{r_i\}_{i=1}^p$ completely characterize the mutual information between the random vectors x and y. For more details, including extensions to infinite–dimensional random processes, see [35].

In many application areas, canonical correlations analysis is used as an approximation tool. For example, one would often like to find a matrix $L \in \mathbb{R}^{m \times k}$, $k < m$, such that $I\left(L^T x; y\right)$ is as large as possible. The optimal solution to this problem chooses the columns of L to span the subspace of $\mathcal{H}_x$ corresponding to the k largest canonical correlation coefficients between x and y. Similarly, such an L defines the k–dimensional subspace of $\mathcal{H}_x$ which minimizes the conditional mutual information $I\left(x; y | L^T x\right)$. This latter property has led to the extensive use of canonical correlations in algorithms for the approximate stochastic realization of state space models for time series [3, 5, 27]. These stochastic realization algorithms have been extended to the design of multiscale tree–structured graphical models [41, 42]. In addition, canonical correlations has played an integral role in the design of subspace methods for system identification [49, 71, 72].

B.2.3 Proof of Proposition 2.1

Proposition. For any $s \in \mathcal{V}$, the *conditional* covariance matrix $\mathrm{var}\left(x_s \left| x_{N(s)}\right.\right)$ may be directly calculated from the corresponding block diagonal entry of $J = P^{-1}$:

$$\mathrm{var}\left(x_s \left| x_{N(s)}\right.\right) = \left(J_{s,s}\right)^{-1} \tag{B.20}$$

In addition, for any $s, t \in \mathcal{V}$, the *conditional canonical correlation coefficients* of x_s and x_t, conditioned on their local neighborhood $x_{N(s,t)}$, may be calculated from

$$\left\{\sigma_i\left(\mathrm{var}\left(x_s \left| x_{N(s,t)}\right.\right)^{-\frac{1}{2}} \mathrm{cov}\left(x_s, x_t \left| x_{N(s,t)}\right.\right) \mathrm{var}\left(x_t \left| x_{N(s,t)}\right.\right)^{-\frac{1}{2}}\right)\right\}$$
$$= \left\{\sigma_i\left(\left(J_{s,s}\right)^{-\frac{1}{2}} J_{s,t} \left(J_{t,t}\right)^{-\frac{1}{2}}\right)\right\} \tag{B.21}$$

Proof. Suppose a Gaussian random vector x with inverse covariance matrix J is partitioned into two sets $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. If the inverse covariance is partitioned in the

same way as $J = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix}$, then from equation (B.5) we see that $\mathrm{var}\left(x_1 \left| x_2 \right.\right) = J_{11}^{-1}$. If we choose $x_1 = x_s$, this implies that $\mathrm{var}\left(x_s \left| x_{\mathcal{V}\backslash s} \right.\right) = (J_{s,s})^{-1}$. Equation (B.20) then follows directly from the Markov properties of the graph.

Alternatively, if we choose $x_1 = \begin{bmatrix} x_s \\ x_t \end{bmatrix}$ for some $s, t \in \mathcal{V}$, we have

$$\begin{aligned} \begin{bmatrix} J_{s,s} & J_{s,t} \\ J_{t,s} & J_{t,t} \end{bmatrix}^{-1} &= \begin{bmatrix} \mathrm{var}\left(x_s \left| x_{\mathcal{V}\backslash\{s,t\}} \right.\right) & \mathrm{cov}\left(x_s, x_t \left| x_{\mathcal{V}\backslash\{s,t\}} \right.\right) \\ \mathrm{cov}\left(x_t, x_s \left| x_{\mathcal{V}\backslash\{s,t\}} \right.\right) & \mathrm{var}\left(x_t \left| x_{\mathcal{V}\backslash\{s,t\}} \right.\right) \end{bmatrix} \\ &= \begin{bmatrix} \mathrm{var}\left(x_s \left| x_{N(s,t)} \right.\right) & \mathrm{cov}\left(x_s, x_t \left| x_{N(s,t)} \right.\right) \\ \mathrm{cov}\left(x_t, x_s \left| x_{N(s,t)} \right.\right) & \mathrm{var}\left(x_t \left| x_{N(s,t)} \right.\right) \end{bmatrix} \end{aligned} \tag{B.22}$$

where the second line follows from the Markov properties of the graph. Using the block matrix inversion formulas (A.2, A.3) to evaluate the left-hand side of equation (B.22), we have

$$\mathrm{var}\left(x_s \left| x_{N(s,t)} \right.\right)^{-\frac{1}{2}} = \left(J_{s,s} - J_{s,t}(J_{t,t})^{-1} J_{t,s}\right)^{\frac{1}{2}} \tag{B.23a}$$

$$\mathrm{var}\left(x_t \left| x_{N(s,t)} \right.\right)^{-\frac{1}{2}} = \left(J_{t,t} - J_{t,s}(J_{s,s})^{-1} J_{s,t}\right)^{\frac{1}{2}} \tag{B.23b}$$

$$\begin{aligned} \mathrm{cov}\left(x_s, x_t \left| x_{\mathcal{V}\backslash\{s,t\}} \right.\right) &= -\left(J_{s,s} - J_{s,t}(J_{t,t})^{-1} J_{t,s}\right)^{-1} J_{s,t}(J_{t,t})^{-1} && \text{(B.23c)} \\ &= -(J_{s,s})^{-1} J_{s,t} \left(J_{t,t} - J_{t,s}(J_{s,s})^{-1} J_{s,t}\right)^{-1} && \text{(B.23d)} \end{aligned}$$

Combining these four equations, and using the fact that $\{\sigma_i(A)\} = \left\{\lambda_i\left(AA^T\right)\right\}$ for any matrix A, we have

$$\begin{aligned} &\left\{\sigma_i\left(\mathrm{var}\left(x_s \left| x_{N(s,t)} \right.\right)^{-\frac{1}{2}} \mathrm{cov}\left(x_s, x_t \left| x_{N(s,t)} \right.\right) \mathrm{var}\left(x_t \left| x_{N(s,t)} \right.\right)^{-\frac{1}{2}}\right)\right\} \\ &= \left\{\lambda_i\left(\left(J_{s,s} - J_{s,t}(J_{t,t})^{-1} J_{t,s}\right)^{-\frac{1}{2}} J_{s,t}(J_{t,t})^{-1} J_{t,s}(J_{s,s})^{-1} \left(J_{s,s} - J_{s,t}(J_{t,t})^{-1} J_{t,s}\right)^{\frac{T}{2}}\right)\right\} \\ &= \left\{\lambda_i\left((J_{s,s})^{-\frac{1}{2}} J_{s,t}(J_{t,t})^{-1} J_{t,s}(J_{s,s})^{-\frac{T}{2}}\right)\right\} = \left\{\sigma_i\left((J_{s,s})^{-\frac{1}{2}} J_{s,t}\left(J_{t,t}\right)^{-\frac{1}{2}}\right)\right\} \end{aligned} \tag{B.24}$$

where the second equality follows from the identity $\{\lambda_i(AB)\} = \{\lambda_i(BA)\}$. This proves the desired result. □

Appendix C

Proofs for Chapter 3

In this appendix, we provide proofs for two results from §3.2 which establish basic properties of the embedded trees algorithm, as well as theorems in §3.4 which examine the conditions under which the embedded trees algorithm converges.

C.1 Proof of Theorem 3.2

Theorem. Suppose the embedded trees mean recursion (3.7) is implemented by periodically cycling through T embedded trees, as in equation (3.15). Then the error $e^n \triangleq \widehat{x}^n - \widehat{x}$ and first difference $d^n \triangleq \widehat{x}^n - \widehat{x}^{n-1}$ evolve according to

$$e^{Tn+T} = \left[\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right] e^{Tn} \triangleq \mathbf{E} e^{Tn} \tag{C.1}$$

$$d^{Tn+T+1} = \left[\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_{j+1}}^{-1} K_{\mathcal{T}_j}\right] d^{Tn+1} \triangleq \mathbf{D} d^{Tn+1} \tag{C.2}$$

where $\widehat{J}_{\mathcal{T}_{T+1}} = \widehat{J}_{\mathcal{T}_1}$, and the matrices $\mathbf{E}$ and $\mathbf{D}$ have the same eigenvalues. If $\rho(\mathbf{E}) < 1$, then for arbitrary $\widehat{x}^0$, $e^n \stackrel{n\to\infty}{\longrightarrow} 0$ at an asymptotic rate of at most $\gamma \triangleq \rho(\mathbf{E})^{\frac{1}{T}}$. Alternatively, if $\rho(\mathbf{E}) > 1$, then $|e^n| \stackrel{n\to\infty}{\longrightarrow} \infty$ for almost all $\widehat{x}^0$.

Proof. Equations (C.1, C.2) are direct specializations of equations (3.13, 3.14) for the case of periodically varying cutting matrices. The statements connecting the convergence of e^n to the spectral radius $\rho(\mathbf{E})$ then follow from basic linear algebra.

To show that $\{\lambda_i(\mathbf{E})\} = \{\lambda_i(\mathbf{D})\}$, we note from Lemma 3.4 (see §3.2.4) that $\widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} = \widehat{P} G_{\mathcal{T}_n}$ for some symmetric matrix $G_{\mathcal{T}_n}$, where $\widehat{P} = \widehat{J}^{-1}$. Recall that for arbitrary square matrices A and B, $\{\lambda_i(A)\} = \{\lambda_i(A^T)\}$ and $\{\lambda_i(AB)\} = \{\lambda_i(BA)\}$. Therefore, if A is symmetric, $\{\lambda_i(AB)\} = \{\lambda_i(AB^T)\}$. Applying these basic properties, we have

$$\begin{aligned}
\{\lambda_i(\mathbf{D})\} &= \left\{\lambda_i\left(\widehat{J}_{\mathcal{T}_1}^{-1} K_{\mathcal{T}_T} \widehat{J}_{\mathcal{T}_T}^{-1} K_{\mathcal{T}_{T-1}} \cdots \widehat{J}_{\mathcal{T}_3}^{-1} K_{\mathcal{T}_2} \widehat{J}_{\mathcal{T}_2}^{-1} K_{\mathcal{T}_1}\right)\right\} \\
&= \left\{\lambda_i\left(\widehat{J}_{\mathcal{T}_1}^{-1} K_{\mathcal{T}_1} \widehat{J}_{\mathcal{T}_2}^{-1} K_{\mathcal{T}_2} \cdots \widehat{J}_{\mathcal{T}_{T-1}}^{-1} K_{\mathcal{T}_{T-1}} \widehat{J}_{\mathcal{T}_T}^{-1} K_{\mathcal{T}_T}\right)\right\} \\
&= \left\{\lambda_i\left(\widehat{J}_{\mathcal{T}_2}^{-1} K_{\mathcal{T}_2} \widehat{J}_{\mathcal{T}_3}^{-1} K_{\mathcal{T}_3} \cdots \widehat{J}_{\mathcal{T}_T}^{-1} K_{\mathcal{T}_T} \widehat{J}_{\mathcal{T}_1}^{-1} K_{\mathcal{T}_1}\right)\right\} \\
&= \left\{\lambda_i\left(\widehat{P} G_{\mathcal{T}_2} \widehat{P} G_{\mathcal{T}_3} \cdots \widehat{P} G_{\mathcal{T}_T} \widehat{P} G_{\mathcal{T}_1}\right)\right\} \\
&= \left\{\lambda_i\left(\widehat{P} G_{\mathcal{T}_T} \widehat{P} G_{\mathcal{T}_{T-1}} \cdots \widehat{P} G_{\mathcal{T}_2} \widehat{P} G_{\mathcal{T}_1}\right)\right\} \\
&= \left\{\lambda_i\left(\widehat{J}_{\mathcal{T}_T}^{-1} K_{\mathcal{T}_T} \widehat{J}_{\mathcal{T}_{T-1}}^{-1} K_{\mathcal{T}_{T-1}} \cdots \widehat{J}_{\mathcal{T}_2}^{-1} K_{\mathcal{T}_2} \widehat{J}_{\mathcal{T}_1}^{-1} K_{\mathcal{T}_1}\right)\right\} = \{\lambda_i(\mathbf{E})\} \qquad \square
\end{aligned}$$

C.2 Proof of Lemma 3.4

Lemma. Given two inverse covariance matrices $\widehat{J}$ and $\widehat{J}_{\mathcal{T}_n}$ of dimension Nd, let $K_{\mathcal{T}_n} = (\widehat{J}_{\mathcal{T}_n} - \widehat{J})$ be the associated symmetric cutting matrix. Defining $r \triangleq \text{rank}(K_{\mathcal{T}_n})$, the reduced–rank diagonalization of $K_{\mathcal{T}_n}$ is given by $K_{\mathcal{T}_n} = U_n D_n U_n^T$, where $D_n \in \mathbb{R}^{r \times r}$ is a diagonal matrix containing the nonzero eigenvalues of $K_{\mathcal{T}_n}$ and the columns of $U_n \in \mathbb{R}^{Nd \times r}$ are the corresponding eigenvectors ($U_n^T U_n = I_r$). We then have

$$\widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} = \left(\widehat{P}^{-1} + K_{\mathcal{T}_n}\right)^{-1} K_{\mathcal{T}_n} = \widehat{P} U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n\right)^{-1} U_n^T$$

Proof. Using the matrix inversion lemma (A.4), we have

$$\widehat{J}_{\mathcal{T}_n}^{-1} = \left(\widehat{P}^{-1} + U_n D_n U_n^T\right)^{-1} = \widehat{P} - \widehat{P} U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n\right)^{-1} U_n^T \widehat{P}$$

The desired result then follows from algebraic manipulation:

$$\begin{aligned}
\widehat{J}_{\mathcal{T}_n}^{-1} K_{\mathcal{T}_n} &= \left[\widehat{P} - \widehat{P}U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n\right)^{-1} U_n^T \widehat{P}\right] U_n D_n U_n^T \\
&= \widehat{P}U_n \left[I - \left(D_n^{-1} + U_n^T \widehat{P} U_n\right)^{-1} U_n^T \widehat{P} U_n\right] D_n U_n^T \\
&= \widehat{P}U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n\right)^{-1} \left[D_n^{-1} + U_n^T \widehat{P} U_n - U_n^T \widehat{P} U_n\right] D_n U_n^T \\
&= \widehat{P}U_n \left(D_n^{-1} + U_n^T \widehat{P} U_n\right)^{-1} U_n^T
\end{aligned}$$

□

C.3 Proof of Theorem 3.11

Theorem. Let A be a symmetric positive definite matrix, and K be a symmetric cutting matrix such that $(A+K)$ is nonsingular. Then

$$\rho\left((A+K)^{-1}K\right) < 1 \qquad \text{if and only if} \qquad A + 2K > 0$$

Proof. Suppose that $\rho\left((A+K)^{-1}K\right) < 1$. By Theorem A.1, the matrix H defined below must have positive eigenvalues:

$$\begin{aligned}
H &= \left(I - (A+K)^{-1}K\right)^{-1} \left(I + (A+K)^{-1}K\right) \\
&= \left((A+K)^{-1}A\right)^{-1} \left((A+K)^{-1}(A+K+K)\right) \\
&= A^{-1}(A+K)(A+K)^{-1}(A+2K) = A^{-1}(A+2K)
\end{aligned}$$

Algebraic manipulation of this equation then gives

$$A^{-\frac{1}{2}}(A+2K)A^{\frac{1}{2}} = A^{\frac{1}{2}} H A^{\frac{1}{2}}$$

$A^{-\frac{1}{2}}(A+2K)A^{\frac{1}{2}}$ is a similarity transform of $(A+2K)$, and therefore has the same eigenvalues. $A^{\frac{1}{2}}HA^{\frac{1}{2}}$ is congruent to H, and therefore must have positive eigenvalues. Since $(A+2K)$ is symmetric, it follows that it must have positive real eigenvalues, and hence be positive definite.

Alternatively, suppose that $(A+2K) > 0$. By Theorem A.1, to verify that

$\rho\left((A+K)^{-1}K\right) < 1$ it is sufficient to show that

$$Q = A - \left((A+K)^{-1}K\right)^T A \left((A+K)^{-1}K\right) > 0$$

Using the identity $(A+K)^{-1}K = I - (A+K)^{-1}A$, we have

$$\begin{aligned} Q &= A - \left(I - A(A+K)^{-1}\right) A \left(I - (A+K)^{-1}A\right) \\ &= A - \left[A - 2A(A+K)^{-1}A + A(A+K)^{-1}A(A+K)^{-1}A\right] \\ &= A(A+K)^{-1}\left[2(A+K) - A\right](A+K)^{-1}A \\ &= \left((A+K)^{-1}K\right)^T (A+2K)\left((A+K)^{-1}K\right) \end{aligned}$$

Because $(A+2K) > 0$, it follows that $Q > 0$, and hence $\rho\left((A+K)^{-1}K\right) < 1$. □

C.4 Proof of Theorem 3.15

Theorem. Consider the embedded trees iteration generated by a pair of cutting matrices $\{K_{\mathcal{T}_1}, K_{\mathcal{T}_2}\}$, as in equation (3.57). Suppose that the following three matrices are positive definite:

$$\widehat{J} + K_{\mathcal{T}_1} + K_{\mathcal{T}_2} > 0 \qquad \widehat{J} + K_{\mathcal{T}_1} - K_{\mathcal{T}_2} > 0 \qquad \widehat{J} - K_{\mathcal{T}_1} + K_{\mathcal{T}_2} > 0$$

Then the resulting iteration is convergent ($\rho(\mathbf{E}) < 1$).

Proof. For simplicity in notation, we let $K_{\mathcal{T}_1} = K_1$ and $K_{\mathcal{T}_2} = K_2$. From Theorem 3.2, we know that $\mathbf{E}$ has the same eigenvalues as the first–difference matrix $\mathbf{D}$ given by

$$\mathbf{D} = \left(\widehat{J} + K_1\right)^{-1} K_2 \left(\widehat{J} + K_2\right)^{-1} K_1$$

Because $(\widehat{J} + K_1 + K_2)$ is positive definite, it has a symmetric square root matrix $(\widehat{J} + K_1 + K_2)^{\frac{1}{2}}$. Using this square root to form a similarity transform, we see that

$\{\lambda_i(\mathbf{D})\} = \{\lambda_i(\tilde{\mathbf{D}})\}$, where $\tilde{\mathbf{D}}$ is given by

$$\tilde{\mathbf{D}} = \left[(\widehat{J} + K_1 + K_2)^{\frac{1}{2}}(\widehat{J} + K_1)^{-1}K_2(\widehat{J} + K_1 + K_2)^{-\frac{1}{2}}\right] \times \left[(\widehat{J} + K_1 + K_2)^{\frac{1}{2}}(\widehat{J} + K_2)^{-1}K_1(\widehat{J} + K_1 + K_2)^{-\frac{1}{2}}\right]$$

Through straightforward algebraic manipulation, it follows that $\tilde{\mathbf{D}} = Q_1Q_2$, where

$$Q_i = I - (\widehat{J} + K_1 + K_2)^{\frac{1}{2}}(\widehat{J} + K_i)^{-1}(\widehat{J} + K_1 + K_2)^{\frac{1}{2}}$$

From basic properties of eigenvalues and singular values, we know that

$$\rho(Q_1Q_2) \leq \sigma_{max}(Q_1Q_2) \leq \sigma_{\max}(Q_1)\sigma_{\max}(Q_2)$$

Therefore, in order to verify that $\rho(\mathbf{E}) < 1$, it is sufficient to show that $\sigma_{\max}(Q_i) < 1$ for $i = 1, 2$. Recall that $\{\sigma_i(Q_i)\} = \left\{\lambda_i\left(Q_iQ_i^T\right)\right\}$. Through algebraic manipulation of the expression for Q_1, it can be shown that

$$\begin{aligned} Q_1Q_1^T &= I - (\widehat{J} + K_1 + K_2)^{\frac{1}{2}}(\widehat{J} + K_1)^{-1}(\widehat{J} + K_1 - K_2)(\widehat{J} + K_1)^{-1}(\widehat{J} + K_1 + K_2)^{\frac{1}{2}} \\ &= I - S_1(\widehat{J} + K_1 - K_2)S_1^T \end{aligned}$$

where $S_1 \triangleq (\widehat{J} + K_1 + K_2)^{\frac{1}{2}}(\widehat{J} + K_1)^{-1}$. From this formula, we see that the eigenvalues of $Q_1Q_1^T$, which must all be positive, are equal to one minus the eigenvalues of $S_1(\widehat{J} + K_1 - K_2)S_1^T$. Thus, $\rho\left(Q_1Q_1^T\right) < 1$ if and only if $S_1(\widehat{J} + K_1 - K_2)S_1^T$ is positive definite, or equivalently if and only if $(\widehat{J} + K_1 - K_2) > 0$. A similar argument for $\rho\left(Q_2Q_2^T\right)$ establishes the $(\widehat{J} - K_1 + K_2) > 0$ condition. □

C.5 Proof of Theorem 3.16

Theorem. Suppose that the ET iteration (3.7) is implemented by periodically cycling through a set of T spanning trees defined by the cutting matrices $\{K_{\mathcal{T}_j}\}_{j=1}^T$. Define $K_{\mathcal{T}_{T+1}} \triangleq K_{\mathcal{T}_1}$. Then if any of the following three sets of *non-equivalent* conditions

holds for $j = 1, \ldots, T$, we are guaranteed that $\rho(\mathbf{E}) < 1$:

$$\begin{aligned}
\sigma_{\max}\left((\widehat{J} + K_{\mathcal{T}_j})^{-1} K_{\mathcal{T}_j}\right) < 1 \qquad &\Longleftrightarrow \qquad (\widehat{J} + K_{\mathcal{T}_j})^2 > K_{\mathcal{T}_j}^2 \\
\sigma_{\max}\left((\widehat{J} + K_{\mathcal{T}_{j+1}})^{-1} K_{\mathcal{T}_j}\right) < 1 \qquad &\Longleftrightarrow \qquad (\widehat{J} + K_{\mathcal{T}_{j+1}})^2 > K_{\mathcal{T}_j}^2 \\
\sigma_{\max}\left((\widehat{J} + K_{\mathcal{T}_j})^{-1} K_{\mathcal{T}_{j+1}}\right) < 1 \qquad &\Longleftrightarrow \qquad (\widehat{J} + K_{\mathcal{T}_j})^2 > K_{\mathcal{T}_{j+1}}^2
\end{aligned}$$

Proof. We begin by proving the first set of conditions. Recall that $\widehat{J}_{\mathcal{T}_j} \triangleq (\widehat{J} + K_{\mathcal{T}_j})$. Using standard properties of eigenvalues and singular values, the spectral radius of the ET convergence matrix $\mathbf{E}$ (see Theorem 3.2) may be bounded as

$$\rho(\mathbf{E}) = \rho\left(\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right) \leq \sigma_{\max}\left(\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right) \leq \prod_{j=1}^{T} \sigma_{\max}\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right)$$

Thus, if $\sigma_{\max}\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right) < 1$ for every cutting matrix, we must have $\rho(\mathbf{E}) < 1$.

To prove the alternate statement of the first condition set, we note that

$$\left\{\sigma_i\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right)\right\} = \left\{\lambda_i\left(\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right)\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right)^T\right)\right\} = \left\{\lambda_i\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}^2 \widehat{J}_{\mathcal{T}_j}^{-1}\right)\right\}$$

Thus, $\sigma_{\max}\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}\right) < 1$ if and only if $\rho\left(\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}^2 \widehat{J}_{\mathcal{T}_j}^{-1}\right) < 1$, or equivalently

$$\widehat{J}_{\mathcal{T}_j}^{-1} K_{\mathcal{T}_j}^2 \widehat{J}_{\mathcal{T}_j}^{-1} < I \qquad \Longleftrightarrow \qquad K_{\mathcal{T}_j}^2 < \widehat{J}_{\mathcal{T}_j}^2$$

which is the desired condition.

To prove the second set of conditions, we employ the first difference matrix $\mathbf{D}$ defined in Theorem 3.2:

$$\rho(\mathbf{E}) = \rho(\mathbf{D}) = \rho\left(\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_{j+1}}^{-1} K_{\mathcal{T}_j}\right) \leq \sigma_{\max}\left(\prod_{j=1}^{T} \widehat{J}_{\mathcal{T}_{j+1}}^{-1} K_{\mathcal{T}_j}\right) \leq \prod_{j=1}^{T} \sigma_{\max}\left(\widehat{J}_{\mathcal{T}_{j+1}}^{-1} K_{\mathcal{T}_j}\right)$$

The alternate statement follows from arguments identical to those above. Finally, the third set of conditions may be obtained by using Proposition 3.5 to reverse the order of the cutting matrices, and then again applying the same bounds. □

Bibliography

[1] L. Adams. m–Step preconditioned conjugate gradient methods. *SIAM Journal on Scientific and Statistical Computing*, 6(2):452–463, April 1985.

[2] S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46(2):325–343, March 2000.

[3] H. Akaike. Markovian representation of stochastic processes by canonical variables. *SIAM Journal on Control*, 13(1):162–173, January 1975.

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

[5] K. S. Arun and S. Y. Kung. Balanced approximation of stochastic systems. *SIAM Journal on Matrix Analysis and Applications*, 11(1):42–68, January 1990.

[6] O. Axelsson. Bounds of eigenvalues of preconditioned matrices. *SIAM Journal on Matrix Analysis and Applications*, 13(3):847–862, July 1992.

[7] R. Barrett et al. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM, Philadelphia, 1994.

[8] M. Bern, J. R. Gilbert, B. Hendrickson, N. Nguyen, and S. Toledo. Support–graph preconditioners. *SIAM Journal on Matrix Analysis and Applications*, January 2001. Submitted.

[9] J. G. Berryman. Analysis of approximate inverses in tomography II: Iterative inverses. *Optimization and Engineering*, 1:437–473, 2000.

[10] D. P. Bertsekas. *Dynamic Programming and Optimal Control: Volume I.* Athena Scientific, Belmont, Massachusetts, 1995.

[11] D. P. Bertsekas. *Dynamic Programming and Optimal Control: Volume II.* Athena Scientific, Belmont, Massachusetts, 1995.

[12] D. P. Bertsekas. Generic rank-one corrections for value iteration in Markovian decision problems. *Operations Research Letters*, 17:111–119, 1995.

[13] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of Royal Statistical Society, Series B*, 36:192–223, 1974.

[14] G. Birkhoff and R. S. Varga. Implicit alternating direction methods. *Transactions of the AMS*, 92(1):13–24, July 1959.

[15] A. Björck and G. H. Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of Computation*, 27(123):579–594, July 1973.

[16] E. Boman, D. Chen, B. Hendrickson, and S. Toledo. Maximum–weight–basis preconditioners. *Journal on Numerical Linear Algebra*, June 2001. Submitted.

[17] E. Boman and B. Hendrickson. Support theory for preconditioning. *SIAM Journal on Matrix Analysis and Applications*, 2001. Submitted.

[18] P. Brémaud. *Markov Chains: Gibbs Fields, Monte Carlo Simulation, and Queues.* Springer-Verlag, New York, 1999.

[19] D. Chen and S. Toledo. Vaidya's preconditioners: Implementation and experimental study. *Electronic Transactions on Numerical Analysis*, August 2001. Submitted.

[20] K. C. Chou, A. S. Willsky, and A. Benveniste. Multiscale recursive estimation, data fusion, and regularization. *IEEE Transactions on Automatic Control*, 39(3):464–478, March 1994.

[21] P. Clifford. Markov random fields in statistics. In G. R. Grimmett and D. J. A. Welsh, editors, *Disorder in Physical Systems*, pages 19–32. Oxford University Press, Oxford, 1990.

[22] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley, New York, 1991.

[23] M. M. Daniel. *Multiresolution Statistical Modeling with Application to Modeling Groundwater Flow.* PhD thesis, Massachusetts Institute of Technology, February 1997.

[24] M. M. Daniel and A. S. Willsky. A multiresolution methodology for signal–level fusion and data assimilation with applications to remote sensing. *Proceedings of the IEEE*, 85(1):164–180, January 1997.

[25] J. W. Demmel. *Applied Numerical Linear Algebra.* SIAM, Philadelphia, 1997.

[26] A. P. Dempster. Covariance selection. *Biometrics*, 28:157–175, March 1972.

[27] U. B. Desai, D. Pal, and R. D. Kirkpatrick. A realization approach to stochastic model reduction. *International Journal of Control*, 42(4):821–838, 1985.

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

[29] 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(2):280–292, March 1995.

[30] A. B. Frakt. *Internal Multiscale Autoregressive Processes, Stochastic Realization, and Covariance Extension.* PhD thesis, Massachusetts Institute of Technology, August 1999.

[31] A. B. Frakt and A. S. Willsky. Computationally efficient stochastic realization for internal multiscale autoregressive models. *Multidimensional Systems and Signal Processing*, 12(2):109–142, April 2001.

[32] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low–level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[33] B. J. Frey and D. J. C. MacKay. A revolution: Belief propagation in graphs with cycles. In *Neural Information Processing Systems 10.* MIT Press, 1998.

[34] R. G. Gallager. *Low-Density Parity-Check Codes.* MIT Press, Cambridge, 1963.

[35] I. M. Gel'fand and A. M. Yaglom. Calculation of the amount of information about a random function contained in another such function. In *American Mathematical Society Translations, series 2*, volume 12, pages 199–250. 1959.

[36] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, November 1984.

[37] G. H. Golub and C. F. Van Loan. *Matrix Computations.* John Hopkins University Press, Baltimore, 1996.

[38] H. V. Henderson and S. R. Searle. On deriving the inverse of a sum of matrices. *SIAM Review*, 23(1):53–60, January 1981.

[39] H. Hotelling. Relations between two sets of variates. *Biometrika*, 28(3):321–377, December 1936.

[40] C. Huang and D. P. O'Leary. A Krylov multisplitting algorithm for solving linear systems of equations. *Linear Algebra and its Applications*, 194:9–29, 1993.

[41] W. W. Irving. *Multiscale Stochastic Realization and Model Identification with Applications to Large–Scale Estimation Problems.* PhD thesis, Massachusetts Institute of Technology, September 1995.

[42] W. W. Irving and A. S. Willsky. A canonical correlations approach to multiscale stochastic realization. *IEEE Transactions on Automatic Control*, 46(10):1514–1528, October 2001.

[43] M. I. Jordan, editor. *Learning in Graphical Models.* MIT Press, Cambridge, 1999.

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

[45] S. M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory.* Prentice Hall, New Jersey, 1993.

[46] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations.* SIAM, Philadelphia, 1995.

[47] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum–product algorithm. *IEEE Transactions on Information Theory*, 7(2):498–519, February 2001.

[48] S. L. Lauritzen. *Graphical Models.* Oxford University Press, Oxford, 1996.

[49] A. Lindquist and G. Picci. Canonical correlation analysis, approximate covariance extension, and identification of stationary time series. *Automatica*, 32(5):709–733, 1996.

[50] M. R. Luettgen, W. C. Karl, and A. S. Willsky. Efficient multiscale regularization with applications to the computation of optical flow. *IEEE Transactions on Image Processing*, 3(1):41–64, January 1994.

[51] R. J. McEliece, D. J. C. MacKay, and J. Cheng. Turbo decoding as an instance of Pearl's "Belief Propagation" algorithm. *IEEE Journal on Selected Areas in Communications*, 16(2):140–152, February 1998.

[52] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence 15*, pages 467–475. Morgan Kaufmann, 1999.

[53] R. Nikoukhah, A. S. Willsky, and B. C. Levy. Kalman filtering and Riccati equations for descriptor systems. *IEEE Transactions on Automatic Control*, 37(9):1325–1342, September 1992.

[54] D. P. O'Leary and R. E. White. Multi–splittings of matrices and parallel solution of linear systems. *SIAM Journal on Algebraic and Discrete Methods*, 6(4):630–640, October 1985.

[55] J. M. Ortega. *Numerical Analysis: A Second Course.* Academic Press, New York, 1972.

[56] J. M. Ortega and R. J. Plemmons. Extensions of the Ostrowski–Reich theorem for SOR iterations. *Linear Algebra and its Applications*, 28:177–191, 1979.

[57] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, March 1982.

[58] D. W. Peaceman and H. H. Rachford, Jr. The numerical solution of parabolic and elliptic differential equations. *Journal of the SIAM*, 3(1):28–41, March 1955.

[59] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufman, San Mateo, 1988.

[60] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[61] S. Roweis and Z. Ghahramani. A unifying review of linear Gaussian models. *Neural Computation*, 11:305–345, 1999.

[62] P. Rusmevichientong and B. Van Roy. An analysis of belief propagation on the turbo decoding graph with Gaussian densities. *IEEE Transactions on Information Theory*, 47(2):745–765, February 2001.

[63] L. L. Scharf and C. T. Mullis. Canonical coordinates and the geometry of inference, rate, and capacity. *IEEE Transactions on Signal Processing*, 48(3):824–831, March 2000.

[64] L. L. Scharf and J. K. Thomas. Wiener filters in canonical coordinates for transform coding, filtering, and quantizing. *IEEE Transactions on Signal Processing*, 46(3):647–654, March 1998.

[65] M. K. Schneider. *Krylov Subspace Estimation.* PhD thesis, Massachusetts Institute of Technology, January 2001.

[66] M. K. Schneider, P. W. Fieguth, W. C. Karl, and A. S. Willsky. Multiscale methods for the segmentation and reconstruction of signals and images. *IEEE Transactions on Image Processing*, 9(3):456–468, March 2000.

[67] M. K. Schneider and A. S. Willsky. Krylov subspace estimation. *SIAM Journal on Scientific Computing*, 22(5):1840–1864, 2001.

[68] J. A. Sjogren. Iterative methods in the solution of dependability models. In *Linear Algebra in Signals, Systems, and Control (Boston, MA, 1986)*, pages 220–243. SIAM, Philadelphia, 1988.

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

[70] P. Stein. Some general theorems on iterants. *Journal of Research of the National Bureau of Standards*, 48(1):82–83, January 1952.

[71] Peter Van Overschee and Bart De Moor. N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1):75–93, 1994.

[72] Peter Van Overschee and Bart De Moor. A unifying theorem for three subspace system identification algorithms. *Automatica*, 31(12):1853–1864, 1995.

[73] E. L. Wachspress. Optimum Alternating–Direction–Implicit iteration parameters for a model problem. *Journal of the SIAM*, 10(2):339–350, June 1962.

[74] E. L. Wachspress. Extended application of Alternating–Direction–Implicit iteration model problem theory. *Journal of the SIAM*, 11(4):994–1016, December 1963.

[75] E. L. Wachspress and G. J. Habetler. An Alternating–Direction–Implicit iteration technique. *Journal of the SIAM*, 8(2):403–424, June 1960.

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

[77] M. J. Wainwright, T. Jaakkola, and A. S. Willsky. Tree–based reparameterization for approximate estimation on loopy graphs. In *Neural Information Processing Systems 14*. MIT Press, 2002.

[78] M. J. Wainwright, E. B. Sudderth, and A. S. Willsky. Tree–based modeling and estimation of Gaussian processes on graphs with cycles. In *Neural Information Processing Systems 13*, pages 661–667. MIT Press, 2001.

[79] Y. Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12:1–41, 2000.

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

[81] R. E. White. Multisplitting of a symmetric positive definite matrix. *SIAM Journal on Matrix Analysis and Applications*, 11(1):69–82, January 1990.

[82] J. S. Yedidia. An idiosyncratic journey beyond mean field theory. In D. Saad and M. Opper, editors, *Advanced Mean Field Methods*. MIT Press, 2001.

[83] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In *Neural Information Processing Systems 13*, pages 689–695. MIT Press, 2001.

[84] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *International Joint Conference on Artificial Intelligence*, August 2001.

[85] D. M. Young. *Iterative Solution of Large Linear Systems.* Academic Press, New York, 1971.