Multiscale Gaussian Graphical Models and Algorithms for Large-Scale Inference

by

Myung Jin Choi

B.S. in Electrical Engineering and Computer Science
Seoul National University, 2005

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

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2007

Author . . . .
Department of Electrical Engineering and Computer Science
May 24, 2007

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

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

Multiscale Gaussian Graphical Models and Algorithms for Large-Scale Inference

by

Myung Jin Choi

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2007, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Graphical models provide a powerful framework for stochastic processes by representing dependencies among random variables compactly with graphs. In particular, multiscale tree-structured graphs have attracted much attention for their computational efficiency as well as their ability to capture long-range correlations. However, tree models have limited modeling power that may lead to blocky artifacts. Previous works on extending trees to pyramidal structures resorted to computationally expensive methods to get solutions due to the resulting model complexity.

In this thesis, we propose a pyramidal graphical model with rich modeling power for Gaussian processes, and develop efficient inference algorithms to solve large-scale estimation problems. The pyramidal graph has statistical links between pairs of neighboring nodes within each scale as well as between adjacent scales. Although the graph has many cycles, its hierarchical structure enables us to develop a class of fast algorithms in the spirit of multipole methods. The algorithms operate by guiding far-apart nodes to communicate through coarser scales and considering only local interactions at finer scales.

The consistent stochastic structure of the pyramidal graph provides great flexibilities in designing and analyzing inference algorithms. Based on emerging techniques for inference on Gaussian graphical models, we propose several different inference algorithms to compute not only the optimal estimates but also approximate error variances as well. In addition, we consider the problem of rapidly updating the estimates based on some new local information, and develop a re-estimation algorithm on the pyramidal graph. Simulation results show that this algorithm can be applied to reconstruct discontinuities blurred during the estimation process or to update the estimates to incorporate a new set of measurements introduced in a local region.

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

Acknowledgments

I have made many important choices over the past few years; indeed, one of the best choices I have made is to work with Professor Alan Willsky. I am so grateful that he allowed me much freedom in pursuing my own ideas, while giving me enlightening comments whenever I was lost. His enthusiasm and passion deeply intrigued my curiosity, and motivated the work in this thesis. I would especially like to thank him for kindly revising multiple drafts of this thesis and providing prompt and helpful feedback. Without his guidance and encouragement, this thesis would never have been possible.

The first semester at MIT can be scary from changes in both the academic and social environment. Yet, I could get through it without a research home thanks to Professor Al Oppenheim's guidance and care. I'd like to thank him for being much more than an academic advisor.

SSG is an excellent research home, and especially, working with the 'graphniks' grouplet members has been a great experience. I am very much indebted to Venkat Chandrasekaran, Jason Johnson and Dmitry Malioutov for interesting discussions and many useful MATLAB codes. I'd like to thank Venkat for listening to my partially cooked ideas patiently and giving me valuable suggestions. Jason has been a tremendous help for me in understanding graphical models and developing ideas. I'd also like to acknowledge that without Dmitry's cute idea, I would have had much trouble computing error variances. Other members at SSG have also been a source of inspiration. I'd like to send out special thanks to Emily Fox, who has been a good friend in both academic and non-academic occasions. Kush Varshney, Pat Kreidl, Ayres Fan and Sujay Sanghavi, thank you all for creating such a harmonious environment. I am also looking forward to have many (but not too many!) exciting years together with my new officemates Vincent Tan and Michael Chen.

Not many parents would encourage their daughter to leave to study at the other side of the Earth or to go to Africa to teach programming. I'd like to deeply thank my parents for giving me opportunity to freely pursue my interests. All my endeavors and accomplishments are the fruits of persistent support and love I have received from them.

I have been so happy here at MIT for the past two years, and most of the credits should be given to Taeg Sang Cho. He is my best friend to whom I can lean on any time and talk about anything, my dearest colleague who listened to my practice talks many times and provided encouraging feedback, and much more than that. Not to mention that he cooks me delicious meals every day! Thanks, Tim, for everything.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Gaussian processes are widely used in modeling various natural phenomena not only because they have attractive properties that enable a mathematical analysis of algorithms, but also because a random process in a large-scale system often can be approximated well by a Gaussian distribution. Therefore, the estimation of Gaussian processes in a large-scale system arises in a variety of applications including image processing [34], machine learning [44], oceanography [13], and communication systems [19].

Throughout this thesis, we use the term *estimation* to indicate the process of computing both Bayes' least square estimates and error covariances of the estimates of a Gaussian process given noisy measurements. This estimation problem can be formulated as a system of linear equations, and if the number of variables is small, it can be easily solved by matrix inversion. However, matrix inversion has computation complexity that scales cubically with the number of variables, whereas for large-scale problems with millions or billions variables, we need algorithms with linear computational complexity.

Graphical models provide a powerful framework for stochastic processes by representing random variables and their dependency structures with a graph [24, 25]. Markov random fields (MRFs) are undirected graphical models in which nodes represent random variables and edges capture conditional dependencies among the variables. When the random variables are jointly Gaussian, the graphical model is called a Gauss-Markov random field (GMRF). Gaussian processes defined on graphs provide both intuitive interpretation of existing estimation techniques and also highly efficient algorithms that utilize the graph structure.

Given a distribution on a graphical model, the problem of computing marginal statistics at each node is often called *inference* [55]. Many iterative inference algorithms on a graphical model can be interpreted as passing 'messages' along the edges of the graph. When the MRF of interest has long-range correlations, variables (or nodes) located far apart in the graph need to communicate with each other before an iterative algorithm converges. Instead of passing messages through neighboring nodes, we get significant computational gain by communicating through coarser resolutions. This motivates us to construct graphical models with multiple scales, in which the original model is placed at the finest scale (at the bottom of the hierarchy) and auxiliary variables are added at coarser scales to represent the field of interest at coarser resolutions.

This multiscale, or multiresolution modeling framework has attracted much attention in the signal and image processing community for its modeling power as well as computational efficiency (for a list of references, see [56]). Some researchers focus on *coarse-to-fine philosophy* originated from *multigrid* methods [4] and develop algorithms which consider the stochastic structure of different scales isolated from each other [16, 20]. Others construct statistically consistent *multiscale trees*, a class of multiscale models which allow interaction between nodes at adjacent scales but not within each scale, and develop extremely efficient and powerful algorithms [7, 13]. Many researchers [8, 27–29, 32, 48] consider models which incorporate both intra- and inter- scale interactions, but due to the resulting complexity, they either allow only limited extensions of multiscale trees or use computationally expensive methods to get solutions.

In recent years, there have been significant advances in understanding and developing efficient inference algorithms for a larger class of Gaussian graphical models [6, 11, 21, 37, 48]. Based on these emerging techniques, we no longer need to limit ourselves to tree-structured graphs in order to obtain tractable algorithms. In this thesis, we propose a pyramidal graph in which we allow consistent statistical links between neighbors at each scale as well as between adjacent scales. Then, we develop highly efficient algorithms in the spirit of *multipole methods* [18] to compute the optimal estimates as well as the uncertainties of the estimates given noisy measurements at some of the nodes. In addition, using the consistent graphical structure of our models, we propose efficient methods to 'update' the estimates rapidly when measurements are added or new knowledge of a local region (for example, existence of discontinuities in the field) is provided. Lastly, the problem of fitting the model to best explain the given data is considered.

In the remainder of the introduction, we provide a high-level description of the pyramidal graph and discuss its rich modeling capability and attractive structure that enables efficient inference algorithms. Then, we introduce the problem of updating the estimates based on local changes and discuss how the hierarchical structure in the pyramidal graph can be utilized.

1.1 Multiscale Modeling

Constructing a graphical model to describe a stochastic process involves trade-offs between model complexity and modeling capability. When a pair of nodes are not connected with an edge, the corresponding probability distribution is required to satisfy some constraints (see Section 2.1). So as we allow more edges, the modeling capability of a graphical model increases, i.e. the graph can represent a broader set of probability densities. However, the complexity of an inference algorithm usually depends on the *sparsity* of the graph (see Section 2.2.2), which means that we tend to make an inference problem more difficult when we add edges to the graph.

At the one end of the spectrum lie *trees*, graphs without cycles. For Gauss-Markov processes defined on tree-structured graphs, there exist highly efficient algorithms that exactly compute the estimates and error covariances with linear computational complexity [7, 56]. However, this efficiency of trees comes at the cost of limited modeling capability.

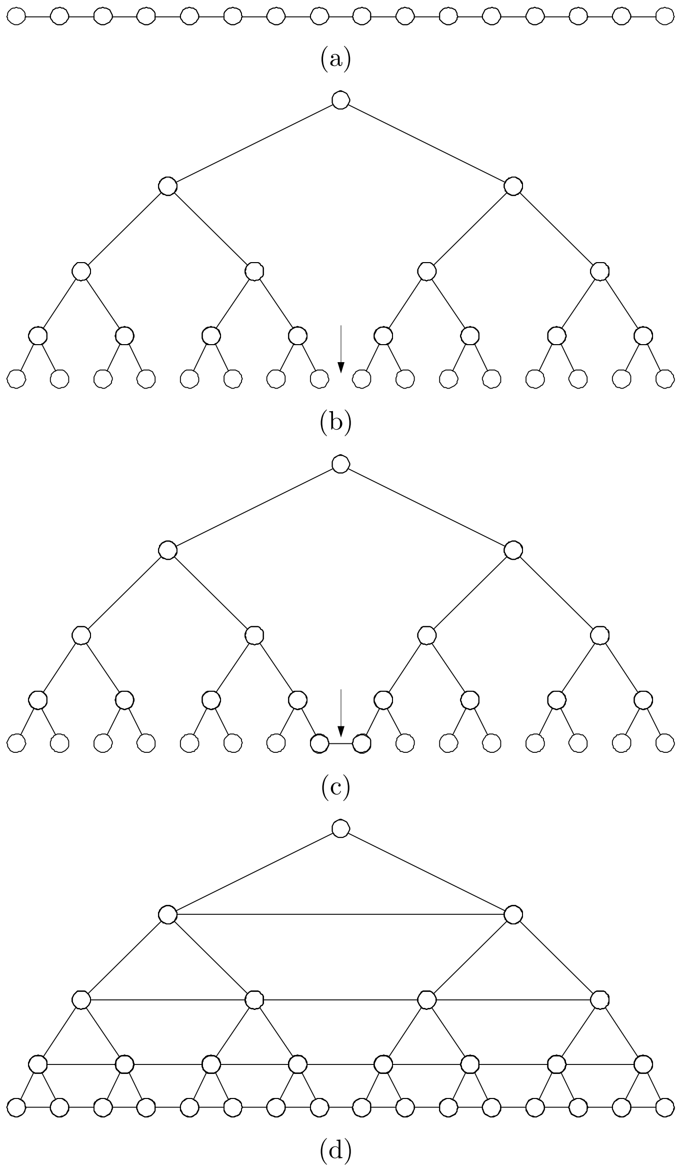


Figure 1-1: Different graphical models for a one-dimensional stochastic process. (a) First-order chain. (b) Multiscale tree. (c) Tree augmented by an edge. (d) Pyramidal graph.

Consider a one-dimensional process, for example, a time series. A simple approach to model such a process is a first-order Markov chain as shown in Figure 1-1(a). However, since each node in the chain is only connected to the nodes next to it, a first-order chain can not capture long-range correlations well. One common way to overcome this limitation while maintaining a tree structure is to construct a multiscale tree model as shown in Figure 1-1(b). Here, the additional nodes correspond to coarser representations of the original nodes at the bottom of the tree. Tree models are better than first-order chains in capturing long-range correlations but they tend to produce blocky artifacts [2, 17, 48]. For example, The neighboring nodes indicated by an arrow in Figure 1-1(b) are located far away in the tree (the shortest path between them consists of many edges), so the correlation between the two nodes cannot be correctly modeled in the tree. Sudderth *et al.* [48] considered an augmented model in Figure 1-1(c) in which a few edges are inserted between the finest scale nodes that are likely to produce most apparent blocky artifacts.

In this thesis, we take a step further and construct a pyramidal graph in Figure 1-1(d) which allows edges between every pair of neighboring nodes at each scale. At a glance, this model seems to reflect an extravagant notion with too many loops compared to trees, but utilizing various emerging techniques that exploit tractable subgraphs (see Section 2.2), we develop highly efficient inference algorithms on the pyramidal graph.

For two-dimensional processes, the motivation to develop multiscale models is even more important. Unlike the one-dimensional case, the most straightforward way of modeling a two-dimensional field is the *nearest-neighbor grid model* shown in Figure 1-2(a) which has many cycles. Iterative algorithms on this grid model tend to converge slowly, and may find only a local minimum of the cost function, which is a serious drawback especially for image classification or segmentation problems.

To overcome these difficulties, multiscale approaches motivated by multigrid methods [4] in computational physics, have been used in image processing [16, 20]. When we construct multiple coarser-resolution versions of the problem, at the coarsest scale, the number of variables may be small enough to perform exact inference and find the global minimum. Once we compute the optimal estimates of a coarser scale, the estimation at the next finer scale can be 'guided' by the result of estimation at coarser scales.

Instead of creating multiple stochastic structures at different scales separated from each other, a multiscale quad-tree model shown in Figure 1-2(b) forms one consistent graphical model structure. Inference algorithms on the tree models are much more

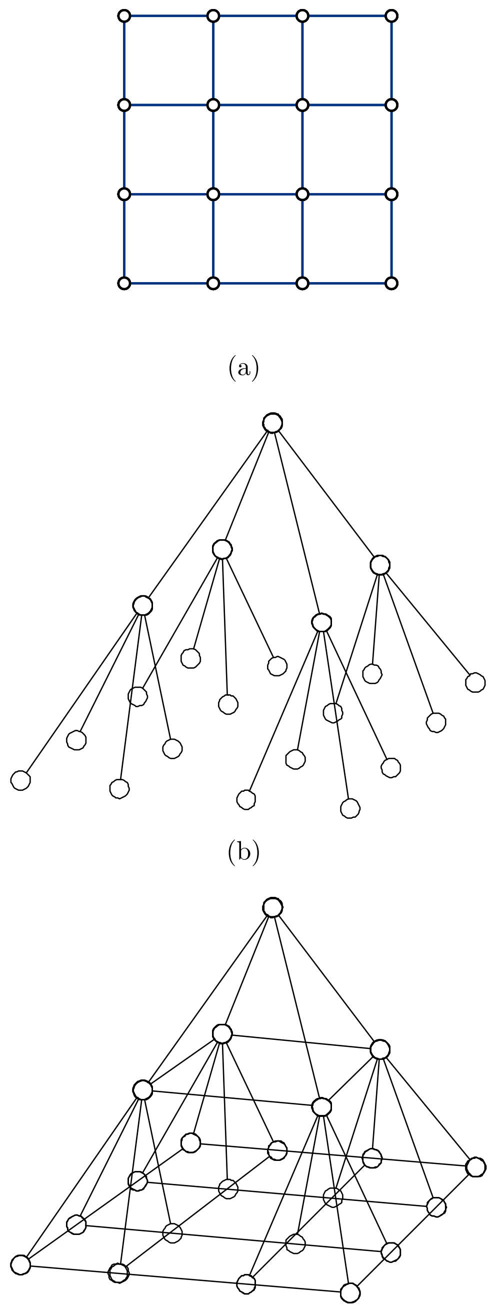


Figure 1-2: Different graphical models for a two-dimensional stochastic process. (a) Nearest-neighbor grid. (b) Multiscale tree. (c) Pyramidal graph.

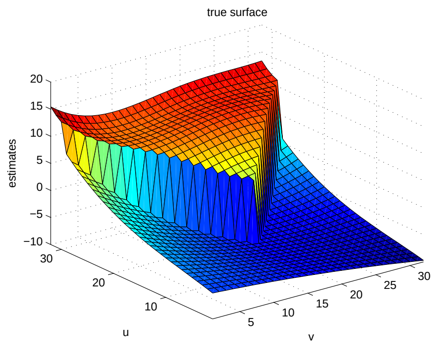
efficient than multigrid-motivated algorithms, but sophisticated modeling is required to avoid blocky artifacts [15, 34].

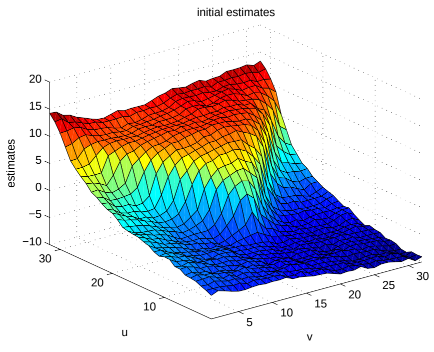
We propose a multiscale pyramidal model in Figure 1-2(c), which incorporates neighbors both within the same scale and between adjacent scales. The pyramidal graph has a consistent statistical dependency structure for the entire graph as with multiscale trees. Similar pyramidal structures have been suggested in [8, 28, 29, 32] for image classification or segmentation applications. However, in those pyramidal models, the original measurements at the finest scale are transformed into coarser scales, either by replicating or by extracting features at multiple resolutions. While it is clear that these multiresolution measurements have dependent errors (as they are all derived from the original fine-scale data), it is implicitly assumed in these approaches that these transformed measurements are conditionally independent. In addition, their approaches use computationally expensive methods such as simulated annealing or Gibbs sampling to obtain solutions.

In spite of the apparent increased complexity compared to a single-scale grid model, the pyramidal graph has many attractive properties that make efficient inference possible. Specifically, we design inference algorithms in the spirit of multipole methods [18], which were originally developed to calculate potentials due to distributions of charges. Instead of calculating every pairwise interaction between the particles, interactions between particle clusters are computed to estimate far-field potentials. This approximation allows us to aggregate far-field effects to reduce computational complexity significantly [15]. In Chapter 4, we use the basic idea of multipole methods to develop efficient inference algorithms, in which variables far-apart communicate through coarser resolutions and nearby variables interact at finer resolutions.

1.2 Re-estimation

Assume that we already have solved an estimation problem based on a large number of measurements, and then wish to modify the estimates to account for new local information. Since variables are correlated with each other, nodes outside the area with new information also need to be updated. Restarting the estimation algorithm would be time-consuming and inefficient. We refer the problem of efficiently updating





(a) (b)

Figure 1-3: Limitation of prior models. (a) Surface with discontinuities. (b) Reconstruction using a smoothness prior model.

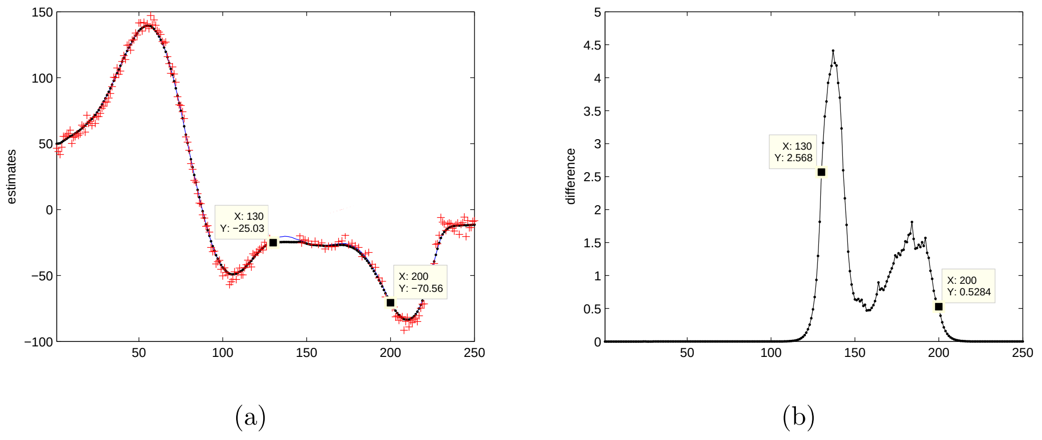
the estimates based on local information as a *re-estimation* problem.

There are two possible scenarios when we need to solve a re-estimation problem. The first case is adding, removing, or changing measurements of a local region. This may happen when measurements are collected over a long period of time or updated continually, of both cases are common in geophysics applications. In addition, we may choose to update the measurements either adaptively or manually if we have unsatisfactory initial estimates at a particular region.

The second issue is detecting and getting accurate estimates of discontinuities. For example, a smoothness prior (see Section 2.2.1) is commonly used to reconstruct surfaces, but as shown in Figure 1-3, a reconstruction based on a smoothness prior results in blurrings across surface discontinuities [14, 56]. For some applications, discontinuities provide more crucial information than smooth regions, so we may wish to post-process the estimates to get more accurate results around the cliffs by relaxing the smoothness prior locally.

In Gaussian graphical models, these two cases can be interpreted in a unified framework of updating nodes when a few model parameters are perturbed from their initial values. The questions arising from this problem is first, what variables should we update and second, how can we update them rapidly.

When the field of interest has long-range correlations, changing the variables in a local region may affect variables far apart. In the spirit of multipole algorithms,



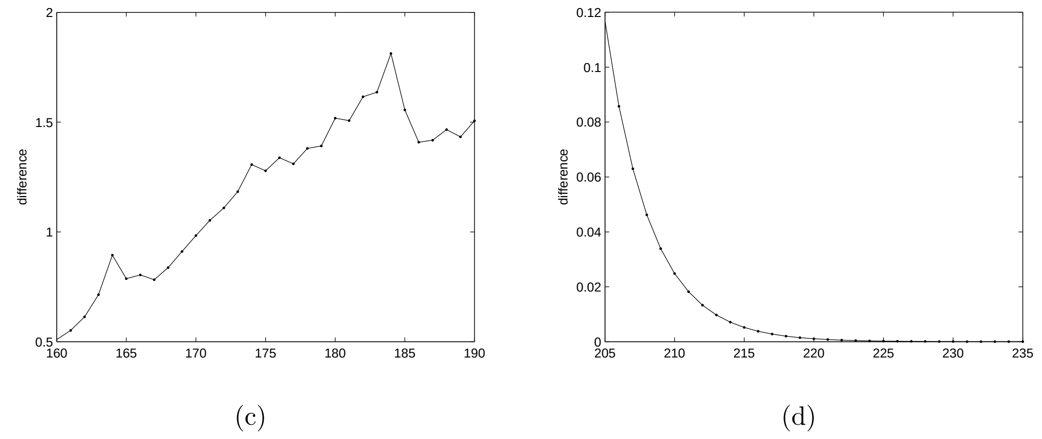


Figure 1-4: Effect of adding new measurements at a local interval.
(a) Measurements (plotted as +) and estimates before (thick solid line) and after (thin solid line) a new set of measurements are added at the interval $x = [130, 200]$, indicated with square boxes in the figure. (b) Difference between the two estimates in a magnified view in y-axis. (c) Difference for the interval $x = [160, 190]$, which is inside the region with added measurements. (d) Difference for the interval $x = [205, 235]$, which is just outside the region with added measurements.

mentioned in the previous section, we model far-field effects as interactions at coarser resolution. Figure 1-4(a) shows two estimates of a one-dimensional process before and after a set of new measurements are added inside the indicated interval $x = [130, 200]$. The difference of the two estimates is magnified in Figure 1-4(b)-(d). Inside the region with added measurements, the difference has high-frequency components as shown in (c), but the difference outside the region in (d) appears smooth and can be well described at a coarser resolution.

This observation suggests that a multiscale representation is an appropriate framework to solve re-estimation problems. When a local region is perturbed, we update the neighboring nodes at a fine scale, but change far away nodes only at coarser scales. In this way, we can update the estimates efficiently without restarting the estimation procedure.

1.3 Thesis Organization

The remainder of the thesis is organized as follows.

Chapter 2. Background

In Chapter 2, we first introduce basic concepts and terminology for graphical models, especially focusing on Gauss-Markov random fields. Then, we discuss estimation of Gaussian processes and a class of iterative algorithms on graphical models based on tractable subgraphs. A *walk-sum* interpretation of inference ensures that for a certain class of graphical models, an iterative algorithm converges regardless of the order of subgraphs it chooses, so we are allowed to choose the subgraphs adaptively to achieve a faster convergence. In addition, we introduce recently developed techniques to approximately compute variances in Gaussian graphical models. Then, we review the existing hierarchical algorithms and models that have been widely used in inference, image segmentation and classification, and solving partial differential equations.

Chapter 3. Multiscale Modeling Using a Pyramidal Graph

We propose a multiscale graphical model with a pyramidal structure in Chapter 3, and define a prior model which is appropriate for smooth fields. Our model is mainly motivated by two-dimensional problems, but we also use one-dimensional problems to illustrate our results. The marginal covariance at the finest scale resulting from this prior model shows that the pyramidal graph can capture long-range correlations better than trees or monoscale grid models. In addition, conditioned on adjacent scales, the conditional covariance of one scale decays quickly since long-range correlations are captured by coarser scale nodes. This implies that despite the complicated appearance of the pyramidal graph, we may obtain highly efficient algorithms exploiting its hierarchical structure.

Chapter 4. Inference on the Pyramidal Graph

In Chapter 4, we describe several efficient inference algorithms on the pyramidal graphical model introduced in Chapter 3. In order to compute the optimal estimates, we design a class of multipole-motivated algorithms consisting of two steps: in the tree-inference step, different scales share information so that we can perform approximate inference at coarser scales. Then, during the in-scale inference step, nearby nodes within each scale pass messages to each other to obtain smooth estimates. Since our pyramidal graph is a GMRF, recently developed techniques for inference in graphs with cycles, such as Embedded Trees (ET) [6, 48] and Lagrangian Relaxation (LR) methods [21] can also be applied. Using the analysis in [6], we show that the multipole-motivated algorithms are guaranteed to converge on the pyramidal graph. Error covariances can be approximately computed using the LR method or the low-rank approximation algorithms [35, 36, 38]. We also consider the re-estimation problems and conclude the chapter with a set of simulations which support the effectiveness of the proposed inference algorithms.

Chapter 5. Multiscale Parameter Estimation

Without the full knowledge of prior models, it is necessary to estimate the model parameters from given data in order to fit the model to best describe the data. We discuss parameter estimation in the pyramidal graph in Chapter 5. Since measurements are only available at the finest scale, it is not easy to estimate the model parameters for the entire pyramidal graph. When we allow a single free parameter to control the prior model, we can apply the standard Expectation Maximization (EM) algorithm which is commonly used for parameter estimation with partially observed data. However, as soon as we increase the number of free parameters, the EM algorithm becomes intractable for our model. We suggest possible directions to perform approximate parameter estimation and leave their investigation as future research topics.

Chapter 6. Conclusions

The main contributions of this thesis are summarized in Chapter 6. We present possible directions to extend the pyramidal graph approach and discuss several open problems in multiscale modeling.

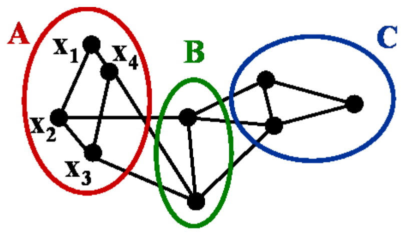
Chapter 2

Background

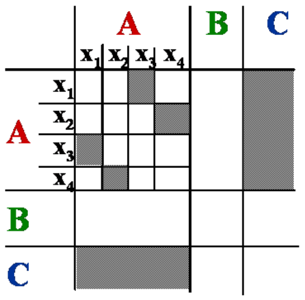
In this chapter, we introduce basic concepts necessary for the subsequent chapters and review the literature on multiscale models and algorithms. We begin Section 2.1 by discussing graphical models and exponential families, and then formulate the problem of estimating Gaussian processes in the graphical model framework. Then, in Section 2.2, we introduce iterative algorithms for efficient inference on graphs with cycles, and describe walk-sum analysis and adaptive iterations which will be utilized in both estimation and re-estimation algorithms in Chapter 4. In addition, low-rank approximation methods to compute variances are introduced. Lastly, in Section 2.3, we review the literature on multiscale models and algorithms, and address the limitations of existing methods which motivate our pyramidal graph in Chapter 3.

2.1 Gaussian Graphical Models

This section provides a brief description of graphical models and exponential families, beginning with general concepts and then specifying the details for the Gaussian case. Then, we discuss how the problem of estimating Gaussian processes can be formulated in the graphical model framework and describe commonly used prior and observation models.



(a)



(b)

Figure 2-1: (a) Graph separation $\Leftrightarrow$ conditional independence. (b) J is sparse with respect to the graph in (a). The shaded areas correspond to zero elements or zero block matrices.

2.1.1 Graphical Models

In graphical models [25, 26], a probability distribution is represented by a graph $\mathcal{G}$ consisting of nodes V and (directed or undirected) edges $\mathcal{E}$. Each node i is associated with a random variable or a random vector x_i, and edges connecting the nodes capture the statistical dependencies among the random variables or random vectors. We focus on undirected graphical models, or Markov random fields, where an edge from node i to node j is equivalent to an edge from node j to node i. For notational simplicity, we assume that x_i is a scalar for every i, but any of the analysis in this thesis can be easily generalized to the case when x_i is a random vector.

Two sets of nodes A and C are said to be *separated* by B if every path between A and C passes through a node in B as shown in Figure 2-1(a). Let x_A be the collection of random variables corresponding to the nodes in set A, and let x denote x_V, where V is the set of all nodes in $\mathcal{G}$. A stochastic process with pdf $p(x)$ is *Markov with respect to* $\mathcal{G}$ if it satisfies the following condition: If A and C are separated by B in graph $\mathcal{G}$, then x_A and x_C are conditionally independent conditioned on x_B, i.e. $p(x_A, x_C|x_B) = p(x_A|x_B)p(x_C|x_B)$.

A *clique* in a graph is defined as a set of nodes that are fully connected to each other (for example, in Figure 2-1(a), B and C are cliques, but A is not). The Hammersely-Clifford theorem [52] states that if a probability distribution can be factorized as a product of functions on each clique, then the underlying process is Markov with respect to the graph. Conversely, a probability distribution $p(x)$ defined on an MRF can be factorized in terms of *clique potentials* if the pdf is strictly positive ($p(x) > 0$ for all $x \in \mathcal{X}$).

If the random variables corresponding to the nodes on the graph are jointly Gaussian, then the MRF is called a Gauss-Markov random field (GMRF). The pdf of a Gaussian process is parameterized by its mean μ and covariance matrix P:

$$p(x) \propto \exp(-\frac{1}{2}(x-\mu)^T P^{-1}(x-\mu)), \tag{2.1}$$

and we denote the process as $x \sim \mathcal{N}(\mu, P)$. In graphical models, it is more convenient to express a Gaussian process in the equivalent *information form* $x \sim \mathcal{N}^{-1}(h, J)$:

$$p(x) \propto \exp(-\frac{1}{2}x^T J x + h^T x) \tag{2.2}$$

where $J = P^{-1}$ is the *information matrix*, and $h = P^{-1}\mu$ is the *potential vector*. Since a covariance matrix is positive definite, it is necessary that J is also positive definite, and we call a graphical model with $J \succ 0$ a *valid* model. If x is Markov with respect to $\mathcal{G}$, then the inverse covariance matrix J is *sparse with respect to* $\mathcal{G}$: A nonzero off-diagonal element in matrix J indicates the presence of an edge linking the corresponding nodes [47]. An example is shown in Figure 2-1(a) and 2-1(b). J_{12} and J_{21} are nonzero since there is an edge between x_1 and x_2, but J_{13} and J_{31} are zero because x_1 and x_3 are not connected with an edge. Similarly, the block matrices J_{AC} and J_{CA} are zero because there is no edge connecting A and C directly.

2.1.2 Exponential Families

An exponential family [55] of probability distributions is defined by a set of sufficient statistics $\phi_a(x)$ (also called potential functions) and associated parameters θ_a:

$$p(x;\theta) = \exp(\sum_a \theta_a \phi_a(x) - \Phi(\theta)) \tag{2.3}$$

The log partition function $\Phi(\theta)$ normalizes the probability distribution so that it integrates to one:

$$\Phi(\theta) = \log \int_{\mathcal{X}} \exp(\sum_a \theta_a \phi_a(x)) dx, \tag{2.4}$$

where $\mathcal{X}$ is the sample space in which x is taking values. The domain of the exponential parameter vector is the set $\Theta = \{\theta | \Phi(\theta) < \infty\}$. By the Hammersley-Clifford theorem, if each potential function $\phi_a(x)$ is a function of random variables in a clique, then the underlying process x is Markov with respect to the graph $\mathcal{G}$.

The log partition function plays an important role in parameter estimation in Chapter 5, and also in inference for Lagrangian relaxation methods (see Section 2.2.3). Specifically, it can be shown that the derivatives of $\Phi(\theta)$ with respect to θ gives the cumulants of $\phi_a(x)$ [55]:

$$\begin{aligned} \frac{\partial \Phi}{\partial \theta_a}(\theta) &= \mathbb{E}[\phi_a] & (2.5) \\ \frac{\partial^2 \Phi}{\partial \theta_a \partial \theta_b}(\theta) &= \mathbb{E}\{(\phi_a - \mathbb{E}[\phi_a])(\phi_b - \mathbb{E}[\phi_b])\} & (2.6) \end{aligned}$$

where the expectation is taken with respect to $p(x;\theta)$. From (2.6), it can be shown that the log partition function is a convex function of θ.

Let $x = (x_1, \ldots x_n)$ be a Gaussian random vector and represent its probability density in the information form:

$$p(x) = \frac{1}{\sqrt{det(2\pi J^{-1})}} \exp(-\frac{1}{2}x^T J x + h^T x - \frac{1}{2} h^T J^{-1} h) \tag{2.7}$$

Comparing the above equation with (2.3), we can see that Gaussian distributions are a class of exponential families with exponential parameters, sufficient statistics, and the log partition function given as:

$$\begin{aligned} \theta_a &= \{h_i\} \cup \{-0.5 * J_{ii}\} \cup \{-J_{ij}, i \neq j\} \\ \phi_a(x) &= \{x_i\} \cup \{x_i^2\} \cup \{x_i x_j, i \neq j\} \\ \Phi(\theta) &= \frac{1}{2}(n \log(2\pi) + h^T J^{-1} h - \log\det(J)). & (2.8) \end{aligned}$$

2.1.3 Prior and Observation Models

We assume that the field we are estimating is smooth overall, with the possible exception of a few discontinuities. Two models have been commonly used as smoothness priors [56]. The *thin-membrane model* penalizes the gradient by minimizing the differences between the neighbors. Each node is modeled to be close to its neighbor. If we denote the neighboring nodes of x_i as $\mathcal{N}(x_i)$,

$$p(x) \propto \exp(-\alpha_1 \sum_{i \in V} \sum_{j \in \mathcal{N}(x_i)} (x_i - x_j)^2) = \exp(-x^T J_{tm} x) \qquad (2.9)$$

The *thin-plate model* penalizes the curvature. Each node is modeled to be close to the average of its neighbors. While the thin-membrane prior prefers a flat surface over a tilted one, the thin-plate model treats a tilted surface and a flat surface equally as long as they have the same curvature.

$$p(x) \propto \exp(-\alpha_2 \sum_{i \in V} (x_i - \frac{1}{|\mathcal{N}(x_i)|} \sum_{j \in \mathcal{N}(x_i)} x_j)^2) = \exp(-x^T J_{tp} x) \qquad (2.10)$$

Based on (2.9) and (2.10), we can define $h_{prior} = 0$, and J_{prior} as either J_{tm} , J_{tp}, or a mixture of them. Then the J_{prior} matrix is sparse (the number of nonzero elements is small compared to the number of total elements of the matrix), so the corresponding graph is sparse (the number of edges is small compared to that of a fully-connected graph).

Suppose we are given noisy observations $y = Cx + v$, where $v \sim \mathcal{N}(0, R)$ is a Gaussian white noise process. If we have one measurement for each node, C would simply be an identity matrix. More generally, if we have measurements at only a subset of the nodes, then C is a selection matrix with only a single nonzero value (equal to 1) in each row. However, if we are modeling a physical phenomenon which is defined over a continuous field, a measurement may be taken at a spatial location between nodes. In this case, we can either map an observation to the closest node or use bilinear interpolation to involve a set of nodes contributing to the observation, so that the resulting C matrix may have more than one nonzero entry in some of the rows. The conditional distribution of x given the observation y is as follows:

$$\begin{aligned} p(x|y) &\propto p(x)p(y|x) \\ &\propto \exp(-\frac{1}{2}x^T J_{prior} x + h_{prior}^T x) \exp(-\frac{1}{2}(y - Cx)^T R^{-1}(y - Cx)) \\ &\propto \exp(-\frac{1}{2}x^T (J_{prior} + C^T R^{-1} C)x + x^T (h_{prior} + C^T R^{-1} y)) \end{aligned} \tag{2.11}$$

If we take the first approach and assign an observation to the closest node, $C^T R^{-1} C$ is a diagonal matrix, so $J = J_{prior} + C^T R^{-1} C$ has the same sparsity structure as J_{prior}. In other words, including the observation model leaves the graph structure unaltered.

2.1.4 Estimation of Gaussian Processes

In Gaussian processes, both maximum a posteriori (MAP) and Bayes' least squares estimates lead to the conditional mean $E[x|y]$, which can be derived from (2.11):

$$\hat{x} = \arg\max p(x|y) = J^{-1}h, \tag{2.12}$$

where $J^{-1} = (J_{prior} + C^T R^{-1} C)^{-1}$, and $h = h_{prior} + C^T R^{-1} y$. The error covariance matrix is the inverse of the J matrix:

$$P = E[(x - \hat{x})(x - \hat{x})^T |y] = J^{-1} \tag{2.13}$$

When the number of variables is small, the optimal estimates and its error covariance can be directly calculated by inverting J. However, inverting a matrix has a cubic computational complexity, so in large-scale systems with millions or billions of variables, this direct computation is intractable.

If a process x can be modeled in a graph with no loops, an efficient algorithm is available for computing both conditional means and error covariances as described in [56]. For graphs with cycles, we may use Gaussian elimination based on *junction trees* [25, 31] to get exact marginal probabilities, but the complexity is cubic in the order of the *tree-width* of the graph. For example, for a nearest-neighbor grid model shown in Figure 1-2(a), the tree-width is equal to the width of the graph, so for a square grid with N nodes, the junction tree algorithm results in $\mathcal{O}(N^{3/2})$ computations. When the number of variables N is large, we need an algorithm with computational complexity $\mathcal{O}(N)$, so we turn to iterative algorithms introduced in the next section.

2.2 Inference Algorithms on Graphs with Cycles

In the recent few years, there have been significant advances in understanding and developing inference algorithms on graphs with cycles. Embedded subgraph algorithms [6, 11, 48] and Lagrangian relaxation methods [21] exploit tractable subgraphs to solve (2.12) iteratively. These algorithms have linear complexity for each iteration and usually converge in a few iterations compared to the number of variables. Using the walk-sum analysis [37], we can choose the order of subgraphs for Embedded subgraph algorithms adaptively to reduce estimation errors quickly as possible. Although these iterative algorithms converge to the correct mean for a large class of graphical models, estimating error covariances is a more challenging problem. In the last part of this section, we introduce low-rank variance approximation methods [35, 36, 38].

2.2.1 Embedded Subgraph Algorithms

Computing conditional means of Gaussian processes is essentially solving the linear system equation $J\hat{x} = h$. Let $\mathcal{G} = (V, \mathcal{E})$ be the corresponding graph of the random process x. The Embedded Trees (ET) algorithm [6, 48] selects a subset of edges $\mathcal{E}_n \subseteq \mathcal{E}$ at each iteration and forms a spanning tree $\mathcal{G}_n = (V, \mathcal{E}_n)$. Let J_n be the matrix defined as follows:

$$(J_n)_{ij} = \begin{cases} (J)_{ij} & \text{if}(i,j) \in \mathcal{E}_n \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

Then, J_n is sparse with respect to $\mathcal{G}_n$. Let $K_n = J - J_n$, then

$$\hat{x} = J_n^{-1}(h - K_n \hat{x}). \tag{2.15}$$

If we assume that $\hat{x}$ in the right side of the above equation is a fixed vector, this equation can be interpreted as an inference problem in the tree defined by J_n, which can be solved in linear time. This leads to the recursive equation to compute $\hat{x}_n$:

$$\hat{x}^{(n)} = J_n^{-1}(h - K_n \hat{x}^{(n-1)}) \tag{2.16}$$

Instead of selecting a subset of edges, we can also consider the block Gauss-Seidel algorithm [11], which updates a subset of nodes $V_n \subseteq V$ at each iteration. Let $x_{V_n} = \{x_i | i \in V_n\}$ be the variables to be updated at n^{th} iteration and let $x_{V_n^c} = \{x_i | i \notin V_n\}$ be the variables to be unchanged. By reordering the variables, the equation $J\hat{x} = h$ can be decomposed as follows:

$$\begin{pmatrix} J_{V_n} & J_{V_n,V_n^c} \\ J_{V_n^c,V_n} & J_{V_n^c} \end{pmatrix} \begin{pmatrix} \hat{x}_{V_n} \\ \hat{x}_{V_n^c} \end{pmatrix} = \begin{pmatrix} h_{V_n} \\ h_{V_n^c} \end{pmatrix} \tag{2.17}$$

From the upper part of the equation, it follows that

$$\hat{x}_{V_n} = J_{V_n}^{-1} \left(h_{V_n} - J_{V_n,V_n^c} \cdot \hat{x}_{V_n^c} \right) \tag{2.18}$$

If $|V_n|$ is small, (2.18) can be solved by inverting J_{V_n}. When $|V_n|$ is large and inverting the matrix is intractable, we can apply the ET algorithm within the subgraph $\mathcal{G}_n = (V_n, \mathcal{E}_{V_n})$, where $\mathcal{E}_{V_n} = \{(i,j) | (i,j) \in \mathcal{E}, i, j \in V_n\}$. This leads to the hybrid of ET and block Gauss-Seidel algorithms: At n^{th} iteration, choose a subset of variables $V_n \subseteq V$ and a subset of edges $\mathcal{E}_n \subseteq \mathcal{E}_{V_n}$. Let $S_n = (V_n, \mathcal{E}_n)$ be the embedded subgraph of $G_n = (V_n, \mathcal{E}_{V_n})$. A node $i \in V_n$ first gets *messages* from all its neighboring nodes except those in $j \in S_n$. Then we perform local estimation within S_n. A node $i \in V_n^c$

remains unchanged at this iteration.

$$\begin{aligned}\hat{x}_{V_n}^{(n)} &= J_{S_n}^{-1}\left(h_{V_n} - K_{S_n}\cdot\hat{x}_{V_n}^{(n-1)} - J_{V_n,V_n^c}\cdot\hat{x}_{V_n^c}^{(n-1)}\right)\\ \hat{x}_{V_n^c}^{(n)} &= \hat{x}_{V_n^c}^{(n-1)}\end{aligned}\tag{2.19}$$

Using the walk-sum analysis in the next section, it can be shown that this iterative algorithm is guaranteed to converge for a certain class of graphical models.

The marginal error variance of each node corresponds to the diagonal element of the inverse of J. Let e_i be the N-dimensional vector of zeros with a one in the i^{th} position, then

$$(J^{-1})_{ii} = (J^{-1}e_i)_i. \tag{2.20}$$

So the error variance of node i can be computed by setting h in (2.12) to e_i and computing the resulting conditional means. Since conditional means can be computed in $\mathcal{O}(N)$ operations per iteration, it takes $\mathcal{O}(N^2)$ operations at each iteration to compute error variances for all nodes. Sudderth *et al.* [48] developed an algorithm which has linear complexity for each iteration when the graph of interest has cycles but is sparsely connected. Delouille *et al.* [11] focus on sensor network applications and approximately compute the variance of a node by considering only a subset of necessary messages. However, both of these methods are not appropriate to compute error variances of all nodes in a general graphical model, for example, a two-dimensional grid.

2.2.2 Walk-sum Analysis and Adaptive Iterations

Inference in Gaussian graphical models can be interpreted as computing walk-sums on the graph as described in [37]. Let us first define the edge weight of an edge in graph $\mathcal{G} = (V, \mathcal{E})$. The partial correlation coefficient between variable x_i and x_j is defined as the conditional correlation coefficient of x_i and x_j conditioned on all other variables $x_{V\backslash ij} \triangleq \{x_i | i \in V\backslash\{i,j\}\}$:

$$r_{ij} \triangleq \frac{cov(x_i, x_j | x_{V\setminus ij})}{\sqrt{var(x_i|x_{V\setminus ij}) var(x_j|x_{V\setminus ij})}} = -\frac{J_{ij}}{\sqrt{J_{ii}J_{jj}}} \tag{2.21}$$

The edge weight of an edge $(i, j) \in \mathcal{E}$ is defined as the partial correlation coefficient between x_i and x_j and can be computed as follows: let $D = diag(J)$ be a diagonal matrix with diagonal entries of J and $\tilde{J} = D^{-1/2}JD^{-1/2}$ be a normalized J matrix in which all diagonal entries are one. Then, an edge weight r_{ij} of an edge $(i, j) \in \mathcal{E}$ is the (i, j) entry of the matrix $R \triangleq I - \tilde{J}$.

A *walk* of length l in $\mathcal{G}$ is defined as a sequence $w = (i_0, i_1, \ldots, i_l)$ where $i_k \in V$ for all $k = 0, 1, \ldots, l$ and $(i_{k-1}, i_k) \in \mathcal{E}$ for all $k = 1, 2, \ldots, l$. The weight of a walk is defined as the product of all edge weights along the walk:

$$\phi(w) = \prod_{k=1}^{l} r_{i_{k-1}i_k}, \tag{2.22}$$

Then, the (i, j) entry of the matrix R^l is equivalent to the sum of all length-l walks from node i to node j.

Let us denote $\phi(j \to i)$ as the sum of weights of all possible walks from node j to node i.

$$\phi(j \to i) = \sum_{w:j\to i} \phi(w) \tag{2.23}$$

A GMRF is called *walk-summable* if for every $i, j \in V$, the sum in (2.23) converges to the same value for every summation order. For walk-summable models, the inverse of normalized J matrix can be computed by walk-sums:

$$(\tilde{J}^{-1})_{ij} = ((I - R)^{-1})_{ij} = (I + R + R^2 + \cdots)_{ij} = \phi(j \to i). \tag{2.24}$$

Since $\tilde{J}^{-1} = D^{1/2}J^{-1}D^{1/2}$, we can easily recover the covariance matrix $P = J^{-1}$ from the walk-sums.

The normalized conditional means $\mu = \tilde{J}^{-1}h$ can be interpreted as reweighted walk-sums in which each walk is weighted by h_j at the start node j of the walk:

$$\mu_i = \sum_{j \in V} (\tilde{J}^{-1})_{ij} h_j = \sum_{j \in V} h_j \phi(j \to i) \tag{2.25}$$

Chandrasekaran *et al.* [6] analyzed the embedded subgraph algorithms using walk-sums and showed that in *walk-summable models*, as long as every edge is updated infinitely often, the convergence of (2.19) is guaranteed for any order of subgraphs we choose. Taking advantage of this flexibility in choosing the order of subgraphs, they developed techniques for choosing trees and subsets of variables adaptively to reduce the error quickly as possible. These techniques will prove to be useful both for inference and for re-estimation in Chapter 4.

2.2.3 Lagrangian Relaxation Methods

The inference algorithms presented in Section 2.2.1 exploit tractable subgraphs embedded in an intractable graph. In this section, we introduce another method that explicitly decomposes a graph into tractable subgraphs and uses the result of inference in each subgraph to perform approximate inference for the entire graph.

As presented in Section 2.1.2, the derivatives of the log partition function with respect to an exponential parameter gives the expected value of the corresponding potential function. For Gaussian graphical models, we can recover the conditional means, variances of each node, and covariance between neighboring nodes by taking derivatives of the log partition function with respect to the elements in h and J, defined in Section 2.1.4. Therefore, the log partition function is useful not only for parameter estimation but also for inference as well [55]. For tree-structured graphs, the log partition function can be computed in linear computational complexity using a dynamic programming approach [40]. Unfortunately, for intractable graphs, computing the log partition function is at least as difficult as performing inference, so we are interested in finding a surrogate log partition function which is tractable to compute.

Let's consider splitting an intractable graph $\mathcal{G}$ defined by J into subgraphs $\mathcal{G}^k$ and associated J^k such that $J = \sum_k J^k$. Here, for notational simplicity, we consider zero-mean Gaussian processes and assume that $h = 0$, but the analysis can be extended to general cases with an arbitrary mean vector. Then, J determines the exponential parameters, so we denote the log partition function as $\Phi(J)$. Since the log partition function is a convex function of exponential parameters, for any ρ_k's such that $\rho_k > 0, \quad \sum_k \rho_k = 1$, we get an upper bound of the log partition function as follows:

$$\Phi(J) = \Phi(\sum_k \rho_k \frac{J^k}{\rho_k}) \leq \sum_k \rho_k \Phi(\frac{J^k}{\rho_k}) \tag{2.26}$$

Now, a surrogate log partition function can be obtained by minimizing the upper bound. Johnson [21] proved that for a fixed decomposition $\{J^k\}$, the optimal weight ρ^* can be explicitly represented in terms of J^k's as follows:

$$\rho_k = \frac{1}{Z} \exp^{\frac{1}{N} \log \det J^k}, \tag{2.27}$$

where N is the number of nodes in the original graph and Z is the normalizing factor

$$Z = \sum_k \exp^{\frac{1}{N} \log \det J^k} \tag{2.28}$$

In [21], it is shown that for a given set of subgraphs $G^k = (V^k, \mathcal{E}^k)$, minimizing the upper-bound is equivalent to identifying a valid decomposition ($J^k \succ 0$) that satisfies the re-weighted moment-matching conditions:

$$\begin{aligned} \rho_k^* P_i^k &= K_i, \quad \forall k, \quad i \in V^k \\ \rho_k^* P_e^k &= K_e, \quad \forall k, \quad e \in \mathcal{E}^k \end{aligned} \tag{2.29}$$

where ρ_k^* is the optimal weight for J^k and $P^k = (J^k)^{-1}$. K_i and K_e's are Lagrange multipliers and can be interpreted as pseudo-moments of the original graph.

A similar set of conditions is also derived by Wainwright *et al.* [54] for models in which each node is a discrete random variable (or vector). They consider a convex decomposition of exponential parameters to find an upper bound on the log partition function, and developed the *tree-reweighted message passing algorithm* to identify the optimal weights as well as optimal decomposition that minimizes the upper bound. However, instead of passing messages in the original graph, the Lagrangian relaxation algorithm [21] performs inference in each subgraph and exchange potentials among the subgraphs that share the same node or same edge. The pseudo-moments computed by this algorithm converge to the correct conditional means and provides an upper bound on true variances. In Chapter 4, we describe the algorithm in detail, and apply it to our pyramidal graph.

2.2.4 Low-Rank Variance Approximation Algorithms

As shown in the previous sections, it is more challenging to compute exact variances in linear operations per iteration than to compute conditional means. Malioutov *et al.* [36] describe a simple idea to use low-rank approximation to estimate variances for models in which correlations decay exponentially in distance.

Let J be the inverse covariance matrix of $x \in \mathbb{R}^N$. Remember that iterative inference algorithms approach the variance estimation problem as solving $(J^{(-1)}e_i)$ N times, once for each node i. Since this is too costly, consider a matrix $B \in \mathbb{R}^{N \times M}$ with $M \ll N$ and $B^T B = I$, and let us use (BB^T) as a low-rank approximation of I. Let b_i denote the i^{th} row of B and assume that $b_i^T b_i = 1$ for all i. Then,

$$\hat{P}_{ii} \triangleq (J^{-1}(BB^T))_{ii} = P_{ii} + \sum_{i \neq j} P_{ij} {b_i}^T b_j. \tag{2.30}$$

When the model of interest has short-range correlations, P_{ij} becomes negligible compared to P_{ii} when the distance from node i to node j becomes far. Therefore, by designing the matrix B such that b_i and b_j becomes orthogonal only when i and j are close, an unbiased estimator of the variances is developed in [36].

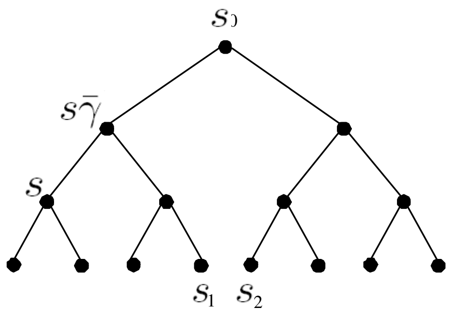
In [38], this idea is extended to an elegant wavelet-based approach to apply the method to models with longer correlation lengths. This approach is based on the observation that when a covariance matrix P is filtered with wavelet, the correlation lengths at finest scale become much shorter. At coarser scales, the correlation still decays slowly, but since coarser scales are low-pass filtered, we are allowed to decimate the output to have a fewer number of variables. The computational complexity can be reduced significantly by this multiscale approach.

2.3 Hierarchical Models

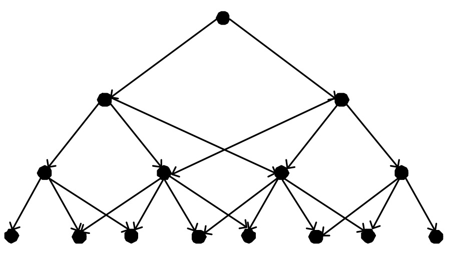
For some cases, multiscale stochastic models are natural framework to describe the physical phenomenon of interest or to assimilate data from distinct sources. In addition, for large-scale problems, it is often desirable to provide estimates at different resolutions depending on the need of users. However, even when multiscale modeling is not required by the physical phenomenon, the data, or the user's interest, multiscale algorithms may provide significant computational gains over the monoscale counterpart. In this section, we review the existing hierarchical models and algorithms. For a comprehensive overview of multiscale, or multiresolution models arising in a wide variety of disciplines, see the survey paper [56].

The common weakness of iterative relaxation methods, such as Jacobi and Gauss-Seidel algorithms [6], is that they tend to eliminate high-frequency components of the error rapidly, but require many iterations to remove low-frequency components of the error. In order to overcome this weakness, multigrid methods [4], commonly used to solve partial differential equations, create multiple grids at different resolutions and replicate measurements at each scale. Low-frequency components of the error are transformed to higher frequencies at coarser resolutions, so they can be eliminated rapidly at those scales. In addition, even if the original problem has a large number of nodes, the size of the problem may be small enough at a coarser scale to get estimates easily. Therefore, multigrid algorithms start by solving a problem at the coarsest scale, and then proceed to the next finer scale and use the estimates of the coarser scale as the initial guess. In turn, the estimates at a finer scale can be used to reduce aliasing effects at a coarser scale.

There has been considerable work to incorporate multigrid concepts in signal or image processing by modeling coarse-to-fine mechanism as stochastic relationships [20, 50]. However, many of these models have limited relationships between different



(a)



(b)

Figure 2-2: (a) A tree-structured graph. (b) Augmented hierarchical graph structure used in [2].

scales. The finer grids are averaged to produce a coarser scale, and the coarser grids are interpolated to create a finer scale. Moreover, the coarser scale variables are not hidden variables because measurements are replicated at every scale. In other words, given a scale, the finer scale and the coarser scales are not conditionally independent since they share the same measurements.

The renormalization group (RG) method [16] generates coarser scales by a non-linear transformation called the RG transformation. The iterations at finer scales are accelerated by searching in the subspace of configurations constrained by coarser scale estimates. However, Markovianity is not usually satisfied at coarser scales generated by RG transformations, and although for certain cases, one can make an approximation as in [16], the computation is not straightforward in general.

Instead of isolating the statistical structure from scale to scale, we can build a coherent graphical model by linking random variables at different resolutions. When a graph does not contain a loop as shown in Figure 2-2(a), both conditional means and error covariances can be efficiently calculated in $\mathcal{O}(d^3N)$ time complexity [56], where d is the state dimension of the nodes and N is the number of nodes. Therefore, we may introduce auxiliary variables at coarser scales and construct a tree-structured graph to approximate the fine scale stochastic process of interest. The multiscale autoregressive (MAR) model specifies the tree model in the following recursive way:

$$x(s) = A(s)x(s\bar{\gamma}) + w(s) \tag{2.31}$$

where $s\bar{\gamma}$ is the parent node of node s and $w(s)$ is a Gaussian white noise process. Therefore, we are assuming that given the parent node, its children are independent of each other.

However, this is a rather severe assumption, especially in the regions where neighboring nodes in the finest scale are far apart in the tree. For example, in Figure 2-2(a), s_1 and s_2 should be independent given s_0. As a result, a tree-structured graph can have boundary artifacts as pointed out in [48]. In order to reduce the blockiness, ones needs to use a sophisticated modeling such as overlapping trees [14], or increase the state dimensions of nodes at coarser scales.

In order to overcome the limitation of tree-structured models, hierarchical graphs with extra edges augmented to trees have been suggested. Bouman and Shapiro [2] proposed a multiscale random field in which a sequence of random fields from coarse to fine scale form a Markov chain. The artifacts of tree-based algorithms are reduced by adding extra edges between adjacent scales as shown in Figure 2-2(b), and a non-iterative upward-downward sweep algorithm for image segmentation is developed. However, in order to circumvent the complexity arising from introducing cycles in the graph, they use tree models in the upward-sweep and only consider extra edges in the downward-sweep.

Sudderth *et al.* [48] introduced a few edges at the finest scale between the neighboring nodes modeled to be far on a tree and reduced the blockiness artifact significantly. Li *et al.* [32] designed a causal quadtree model for image classification application and allowed intrascale interactions only between the nodes that share the same parent node, to incorporate high-frequency information useful for distinguishing classes.

In order to capture both inter- and intra- scale interactions, a pyramidal graph shown in Figure 1-2(c) is a natural extension to the quad-tree models. Kato *et al.* [27–29] constructed such pyramidal graph by introducing a quad-tree neighboring system between two neighboring grids in multigrid models. By partitioning the pyramidal graph into disjoint sets so that the nodes in the same set are conditionally independent given all other sets, they developed a massively parallel relaxation algorithms that updates different scales at the same time. However, these interactions between scales make the model more complicated, and although the algorithm converges in fewer iterations, each iteration becomes computationally much more expensive.

Comer *et al.* [8] also proposed a full pyramidal graph to segment textured images, and considered the neighborhood system that consists of a parent, four children, and four adjacent nodes within the same scale. They used Gibbs sampler to compute the marginal statistics.

The pyramidal graph we are proposing in this thesis essentially has the same graphical structure as considered in [8, 27–29]. However, there are several fundamental differences. First of all, in the previous approaches, the data are either observed at multiple resolutions [8] or replicated for inference at coarser resolutions [27–29]. In our model, the measurements stay in the original resolution (finest scale) so the coarser scale variables are truly hidden variables. Secondly, we use recently developed efficient algorithms for Gaussian graphical models (in particular, Embedded subgraph algorithms, Lagrange Relaxation... etc.) and develop algorithms much faster than simulated annealing or sampling approaches. Thirdly, we utilize the fact that our pyramidal graph is a coherent graphical model with consistent statistical dependencies between intra- and inter- scale variables, which provides great flexibility in designing inference algorithms.

Chapter 3

Multiscale Modeling Using a Pyramidal Graph

In this chapter, we propose a class of multiscale graphical models with a pyramidal structure and demonstrate its rich modeling power. We begin Section 3.1 with the basic notation of pyramidal graphs and extend the thin-membrane model introduced in Section 2.1.3 to define a prior model. In Section 3.2, we observe the resulting covariance structure and compare the correlation decays at the finest scale of the pyramidal graph with the tree and monoscale thin-membrane counterparts. The pyramidal graph can capture long-range correlations better than monoscale thin-membrane models and do not produce blockiness as in tree models. In addition, the conditional covariance of each scale conditioned on other scales can be approximated as a banded covariance matrix. This suggest that despite the complicated appearance of the pyramidal graph, we may obtain highly efficient algorithms utilizing its hierarchical structure.

3.1 Prior Models

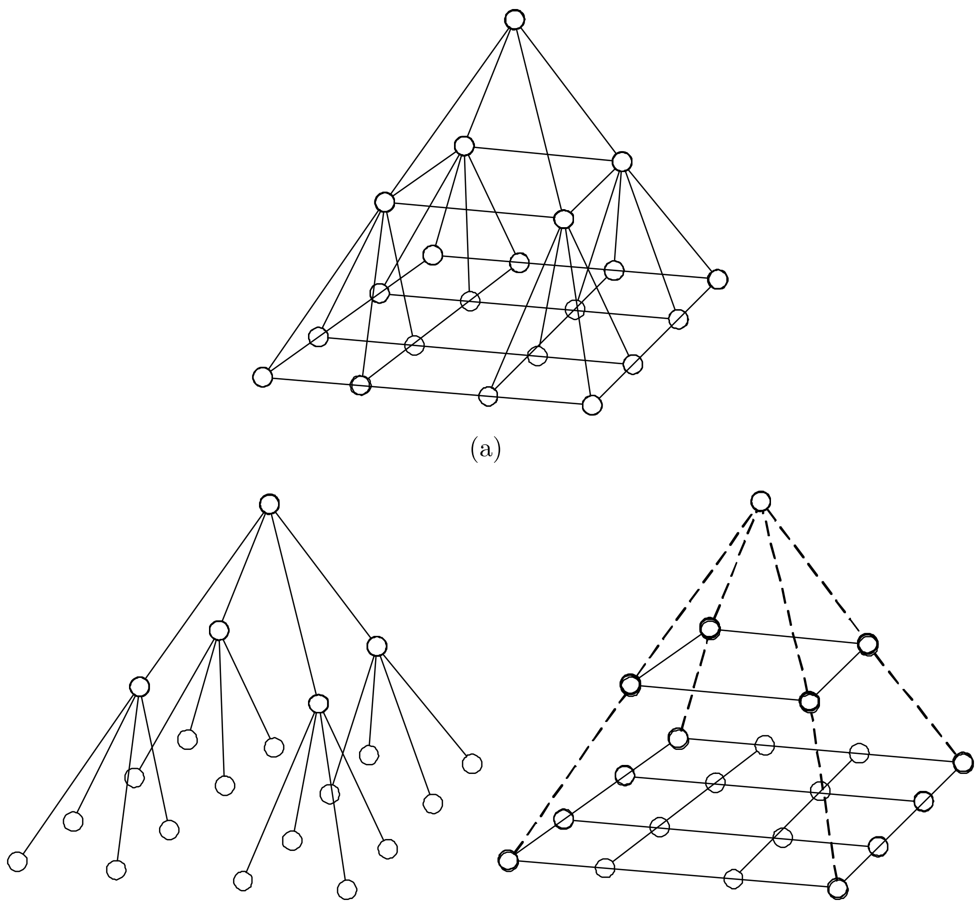
In multiscale modeling, it is common to consider the original resolution as the finest resolution and construct approximate, coarser versions of the problem. Although the pyramidal graph we are proposing in this thesis can easily incorporate data or user objectives at multiple resolutions, we focus on the case in which the coarser scales are merely acting to help the inference at the finest scale. Let's assume that the field of interest is two-dimensional and originally can be described at a single resolution. Even though the iterative algorithms introduced in Section 2.2 provide tractable methods of inference, they may take many iterations to converge for single-resolution models with large numbers of variables and with complex stochastic dependencies. The convergence rate can be significantly improved by introducing auxiliary variables which represent the field of interest at coarser resolutions.

We construct a pyramidal graphical model shown in Figure 3-1(a) by placing the original field at the bottom of the hierarchy and introducing hidden variables at coarser scales. Let M be the number of different levels of resolution in the hierarchy. We denote the coarsest resolution as *scale 1* and place it at the top of the hierarchy. The scale number increases as we go downward and the field of interest is placed at the bottom of the hierarchy and denoted as the *finest scale* or *scale* M. For $1 < m < M$, each scale m has its coarser, or parent scale $m-1$ and the finer, or child scale $m+1$. The i^{th} random variable at scale m is denoted as $x_{(m,i)}$, and the collection of all random variables at scale m is denoted as x_m.

The structure of a Gaussian graphical model can be represented by the corresponding information matrix $J = P^{-1}$. The J matrix for the prior that we use consists of two components:

$$J_{prior} = J_t + J_s. \tag{3.1}$$

where J_t encodes statistical links between different scales, and J_s represents edges within each scale. For a pyramidal graph for two-dimensional processes shown in Figure 3-1(a), J_t corresponds to a quadtree in Figure 3-1(b) in which each parent-child pair is connected by an edge, and J_s corresponds to nearest-neighbor grid models within each scale as shown in Figure 3-1(c). There are many ways to define these priors, but we extend the thin-membrane model introduced in Section 2.1.3 to construct the prior for our pyramidal graph.


(a)

(b)

(c)

Figure 3-1: A pyramidal graphical model and its decomposition. (a) A pyramidal graph for two-dimensional processes. (b) A quadtree (c) Nearest-neighbor grids at multiple scales.

Quadtree structure

A parent node in the quadtree is the coarse representation of its four children. Therefore, we simply let J_t impose the constraint that a parent node is close to its children. If we use $\mathcal{C}(i) \subset V_{m+1}$ to denote the children of node $i \in V_m$, J_t is defined as follows:

$$\exp(-x' J_t x) = \exp(-\sum_{m=1}^{M-1} \beta_m \sum_{i \in V_m} \sum_{j \in \mathcal{C}(i)} (x_{(m,i)} - x_{(m+1,j)})^2), \tag{3.2}$$

where the parameter β_m determines how severely we penalize the difference between the value at a node at scale m and the value at each of its children at scale $m+1$. J_t

is a block tri-diagonal matrix and can be decomposed by scale as follows:

$$J_t = \begin{pmatrix} c\beta_1 I_{N_1} & \beta_1 J_{T_{12}} & 0 & 0 \\ \beta_1 J_{T_{21}} & (\beta_1 + c\beta_2) I_{N_2} & \beta_2 J_{T_{23}} & 0 \\ 0 & \ddots & \ddots & \ddots \\ 0 & 0 & \beta_{M-1} J_{T_{M,M-1}} & \beta_{M-1} I_{N_M} \end{pmatrix} \tag{3.3}$$

Here, N_m indicates the number of nodes at scale m, and I_{N_m} is the $N_m \times N_m$ identity matrix. The constant c is the number of children each parent has, so in our pyramidal graph, c equals 4. $J_{T_{m,m+1}}$ is a sparse $N_m \times N_{m+1}$ matrix in which each entry corresponding to a parent-child pair equals -1, and all other entries are zero. We denote the collection of β_m's as $\beta = [\beta_1, \beta_2, \ldots \beta_{M-1}]$.

Grid structure

The nearest-neighbor grid model J_s imposes smoothness within each scale. Since the edges between different scales are captured by J_t, every element of J_s that corresponds to an inter-scale entry is zero, so it can be decomposed by scale as follows:

$$J_s = \begin{pmatrix} \alpha_1 J_{s1} & 0 & 0 & 0 \\ 0 & \alpha_2 J_{s2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \alpha_M J_{sM} \end{pmatrix} \tag{3.4}$$

where J_{sm} represents a thin-membrane prior at scale m. Therefore, if we let $\mathcal{N}(i) \subset V_m$ be the neighboring nodes of node $i \in V_m$ within the same scale,

$$\exp(-x'_m J_{sm} x_m) = \exp(-\sum_{i \in V_m} \sum_{j \in \mathcal{N}(i)} (x_{(m,i)} - x_{(m,j)})^2). \tag{3.5}$$

Notice that an off-diagonal entry of J_{sm} is $(J_{sm})_{ij} = -1$ if $(i,j) \in \mathcal{E}$ and 0 otherwise. The diagonal elements of J_{sm} are equal to the number of neighbors each node has within scale m. The parameter α_m determines how severely we penalize the gradient of the field at scale m. If we want a smoother field, we can increase the value of α_m. Coarser scale nodes represent spatial regions in which the center points are located farther apart, so it is natural to decrease α_m as we go to a coarser scale. We use vector $\alpha = [\alpha_1, \alpha_2, \dots \alpha_M]$ to denote the collection of α_m's.

Note that the thin-membrane model, as well as its extension to a quadtree and multiple grids, yields positive semidefinite J matrices. Therefore, in order to make J_{prior} a valid prior model, we add a small regularization term ϵI to J_{prior} to make it positive definite. Unless otherwise stated, it is assumed that all prior models in the rest of this chapter are valid models.

Walk-summability

Notice that as long as all parameters α and β are nonnegative, the diagonal elements of $J_{prior} = J_t + J_s$ are positive, and the off-diagonal elements are negative. Therefore, the partial correlation coefficient between any pair of nodes is nonnegative, and the prior of the pyramidal graph is an *attractive model* [37]. As mentioned in Section 2.1.3, if irregular measurements are mapped to the closest nodes, the observation model $C^T R^{-1} C$ is a diagonal matrix with positive elements, so the posterior model $J = J_{prior} + C^T R^{-1} C$ is also an attractive model. It is proven in [37] that all valid and attractive models are walk-summable. Without the regularization term ϵI, J_{prior} is positive semidefinite, but if we have at least one measurement, the observation model makes J a positive definite matrix. Therefore, the posterior model J is walk-summable, and J_{prior} is also walk-summable if we add the regularization term ϵI to make it a valid model.

We may use other variants of prior models for either J_t or J_s. For example, a parent node may be modeled as the scaling coefficient of a wavelet transform [39] of its children, or the thin-plate model introduced in Section 2.1.3 can be used to model intra-scale smoothness in J_s. However, these priors may produce a more complicated graph structure with more edges, and for many cases, the walk-summability of such models is not guaranteed (for example, the thin-plate model is not walk-summable). Therefore, in this thesis, we focus on the multiresolution extension of the

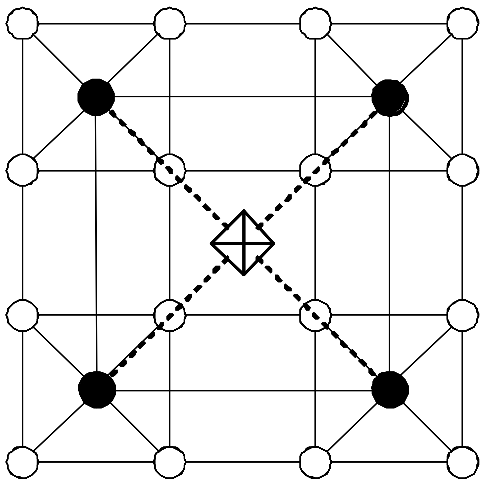


Figure 3-2: An illustration of the pyramidal graph with all nodes projected downward. ○ : nodes at scale $m+1$; ● : nodes at scale m; ⊠ : a node at scale $m-1$.

thin-membrane model to define the prior on pyramidal graphs.

3.2 Covariance Realized by Pyramidal Graphs

In this section, we observe the covariance structure of the pyramidal graph with the prior model defined in the previous section. Since we are primarily interested in modeling the finest scale of pyramidal graphs, we compare the correlation decay at the finest scale with the tree and monoscale thin-membrane counterparts. For illustration purposes, we use one-dimensional processes in this section to plot the decay of correlations with distances. The covariance structure of two-dimensional processes can be described similarly.

Let us decompose the J matrix of the pyramidal graph into block matrices for each scale as follows:

$$J = \begin{pmatrix} & | & | & | & \\ \cdots & J_{[m-1,m-1]} & J_{[m-1,m]} & 0 & 0 \\ 0 & J_{[m,m-1]} & J_{[m,m]} & J_{[m,m+1]} & 0 \\ 0 & 0 & J_{[m+1,m]} & J_{[m+1,m+1]} & \cdots \\ & | & | & | & \end{pmatrix} \tag{3.6}$$

where $J_{[i,j]}$ refers to the $N_i \times N_j$ sub-matrix of J, corresponding to scale i and scale j. Notice that scale m does not have edges from the scales other than its parent scale $m-1$ and its child scale $m+1$. Therefore, $J_{i,m}$ is a zero matrix except $i = m-1, m, m+1$.

The thin-membrane model as well as its extension to trees and pyramidal graphs is nearly singular even with the added regularization term. So, in order to observe how correlations decay with distances, we use *posterior covariance* with a stronger regularization term which corresponds to measurements at the finest scale:

$$P \triangleq J^{-1} = (J_{prior} + C^T R^{-1} C)^{-1} \tag{3.7}$$

For the pyramidal graph, we also consider *posterior conditional covariance* at scale m conditioned on adjacent scales $m-1$ and $m+1$:

$$\bar{P}_{[m,m]} \triangleq \left(J_{[m,m]}\right)^{-1} \tag{3.8}$$

This posterior conditional covariance conditioned on other scales plays an important role in developing efficient inference algorithms, so we use the shortened term *conditional covariance* throughout the thesis to denote $\bar{P}_{[m,m]}$. When we wish to emphasize the contrast between P and $\bar{P}_{[m,m]}$, we use the term *marginal covariance* to denote P. For a mathematical analysis of conditional covariances later in this chapter, we remove the regularization term and analyze *prior conditional covariance*:

$$(\bar{P}_{prior})_{[m,m]} \triangleq \left((J_{prior})_{[m,m]}\right)^{-1} \tag{3.9}$$

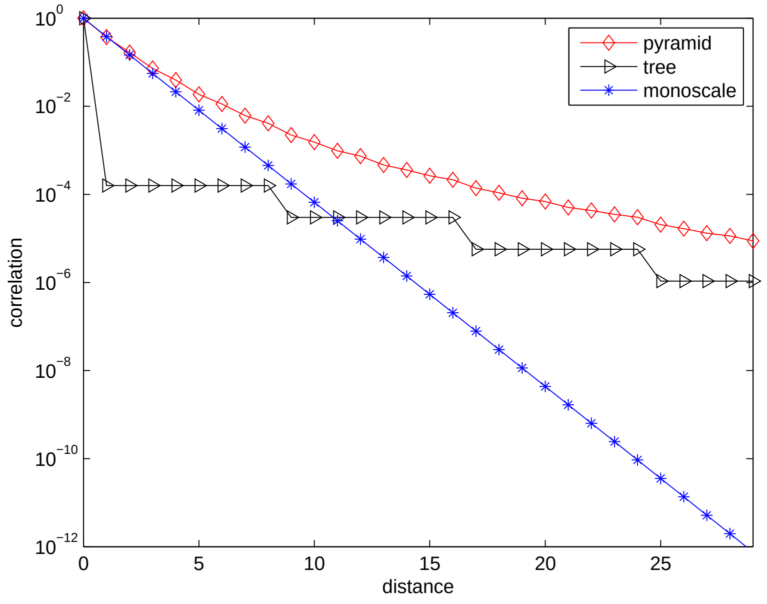
Once we fix the structure of a Gaussian graphical model, its posterior covariance matrix P is parameterized by the noise variance R and the parameters of the prior. In the pyramidal graph, the ratio of α and β of different scales can be adjusted to get the desired covariance structure at the finest scale. We first set the ratio of the parameters based on the physical distance between the corresponding pair of nodes when projected downward as shown in Figure 3-2. The distance between a pair of neighboring nodes at scale m is twice the distance of a pair of neighboring nodes at scale $m+1$. Since we are putting constraints on the square of the differences (see (3.2) and (3.5)), it is appealing to set α_{m-1} as one quarter of α_m, to impose weaker constraints on nodes that are farther apart. Similarly, β_m also decreases by a factor of four as we go to coarser scales. A parent node is located at the center of its four children, so the physical distance between a child and a parent is $1/\sqrt{2}$ of the distance between a pair of siblings (nodes which share the same parent). So, $\beta_{m-1} = \frac{1}{2}\alpha_m$. Therefore, we let $\alpha_M = \varphi$ and set the rest of the parameters as follows:

$$\begin{aligned} \alpha_m &= \frac{\varphi}{4^{M-m}} \qquad m = 1, 2, \ldots M \\ \beta_m &= \frac{1}{2}\frac{\varphi}{4^{M-1-m}} \qquad m = 1, 2, \ldots M-1 \end{aligned} \tag{3.10}$$

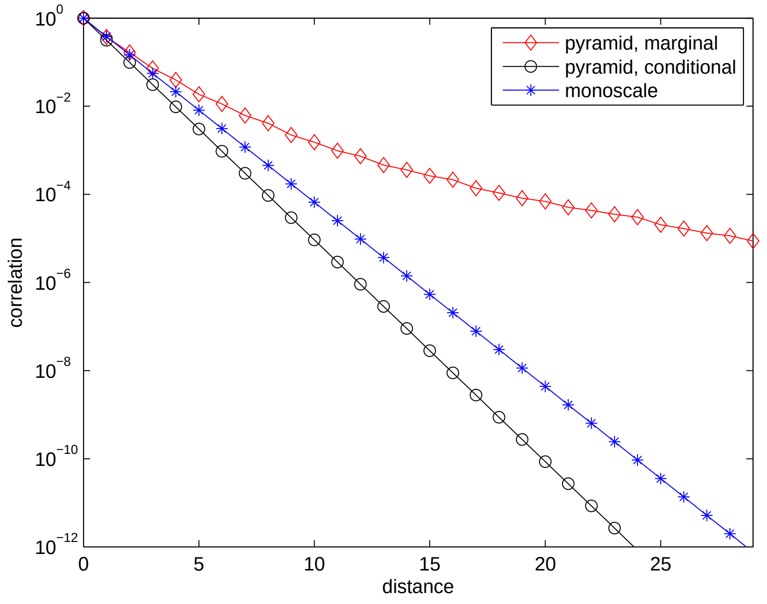
We use a one-dimensional (1D) process with 64 variables and set both φ and the noise variance equal to one ($R = \sigma^2 I$, $\sigma = 1$). The thin-membrane model in 1D is a first-order chain model shown in Figure 1-1(a), and we construct four scales for both the pyramidal graph and the tree. For the tree model, we use the same parameter β but remove all edges within each scale (equivalent to setting $\alpha = 0$). For the monoscale thin-membrane model counterpart, we use the parameter α_M of the pyramidal graph. Let us number the 64 nodes at the finest scale as node i, $i = 1, 2, \ldots 64$ starting from the left. Figure 3-3(a) shows the correlation between node 8 and node i, where i runs from 8 through 37 for the pyramidal graph, the tree, and the monoscale thin-membrane model.

The correlations in the tree model show severe blockiness and depend solely on the length of the paths between the two nodes on the tree. Specifically, since node 8 and node 9 are far apart on the tree (the shortest path between them consists of seven edges), the correlation between the two nodes is very small. Note that this is an extremely naive implementation of a tree-structured graph. In practice, people use more sophisticated models such as overlapping trees [14] or wavelet trees [10].

The finest scale of the pyramidal graph has long-range correlations compared to its monoscale counterpart as shown in Figure 3-3(a), since coarser scale variables impose additional long-range correlations to the thin-membrane model. So, the pyramidal



(a)



(b)

Figure 3-3: The correlation decays of a pyramidal graph and its tree and monoscale counterparts. (a) Correlations of the monoscale thin-membrane model and of the finest scale in the pyramidal graph and in the tree. (b) Conditional correlations at the finest scale of the pyramidal graph, plotted together with marginal correlations at the finest scale and marginal correlations of the monoscale thin-membrane model.

graph is more powerful in modeling processes with long-range correlations such as fractional Brownian motion (fBm) [10]. In addition, the conditional correlation of one scale, conditioned on adjacent scales, decays very quickly since the long-range correlations are captured by coarser nodes. Figure 3-3(b) shows the correlations of the monoscale thin-membrane model and the marginal and conditional correlations at the finest scale of the pyramidal graph. Although the marginal correlation decays slowly, the conditional correlation falls faster than marginal correlation of the monoscale counterpart.

The condition number [3] of a symmetric and positive definite matrix A is given as the ratio of the largest and the smallest eigenvalue:

$$\kappa(A) = \frac{\max eig(A)}{\min eig(A)}. \tag{3.11}$$

A matrix is called well-conditioned if its condition number is small.

Proposition 3.1. *The conditional covariance of one scale of the pyramidal graph conditioned on adjacent scales is well-conditioned compared to the monoscale thin-membrane model with the same parameter.*

Proof. Let us consider the conditional prior covariance matrix of the finest scale as an example. The same analysis can be applied to all other scales as well as to posterior covariance matrices. From (3.1), (3.3) and (3.4), the conditional prior J matrix of the finest scale conditioned on coarser scales is

$$\bar{J}_{[M,M]} \triangleq \alpha_M J_{sM} + \beta_{M-1} I_{N_M}. \tag{3.12}$$

Let $\{\lambda_i\}$ be the eigenvalues of J_{sM}, then the eigenvalues of $\bar{J}_{[M,M]}$ are $\{\alpha_M \lambda_i + \beta_{M-1}\}$. The condition number of $\bar{J}_{[M,M]}$ is

$$\begin{aligned}\kappa(\bar{J}_{[M,M]}) &= \frac{\max eig(\bar{J}_{[M,M]})}{\min eig(\bar{J}_{[M,M]})} = \frac{\alpha_M \lambda_{max} + \beta_{M-1}}{\alpha_M \lambda_{min} + \beta_{M-1}} = \frac{\lambda_{max}}{\lambda_{min}} \cdot \frac{1 + \beta_{M-1}/(\alpha_M \lambda_{max})}{1 + \beta_{M-1}/(\alpha_M \lambda_{min})} \\ &\leq \frac{\lambda_{max}}{\lambda_{min}} = \kappa(J_{sM})\end{aligned} \tag{3.13}$$

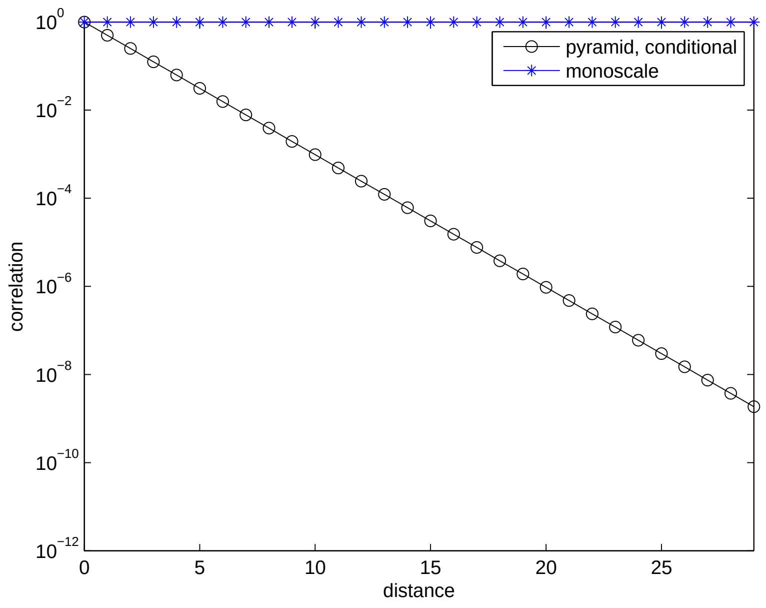


Figure 3-4: Prior conditional correlations of the finest scale in the pyramidal graph conditioned on coarser scales and prior correlations of its monoscale thin-membrane counterpart.

If we focus on prior covariances, J_{sM} is nearly singular, so λ_{min} is close to zero. In this case,

$$\frac{1+\beta/(\alpha_M\lambda_{max})}{1+\beta/(\alpha_M\lambda_{min})} \ll 1 \tag{3.14}$$

and the condition number of $\alpha_M J_{sM}$ is reduced significantly by adding the term $\beta_{M-1} I_{N_M}$ in (3.12). $\square$

Figure 3-4 shows the conditional prior correlations at the finest scale of the pyramidal graph with parameters in (3.10) and the prior correlations of its monoscale counterpart. The condition number of the monoscale thin-membrane model is 5.7646×10^{17}, but it reduces to 9 in the conditional covariance of the pyramidal graph.

Therefore, we may ignore the conditional correlations between the pair of nodes more than a few edges apart, and approximate the structure of each scale as a banded covariance matrix when conditioned on adjacent scales. This indicates that once we have estimates at adjacent scales, we can perform approximate inference at each scale

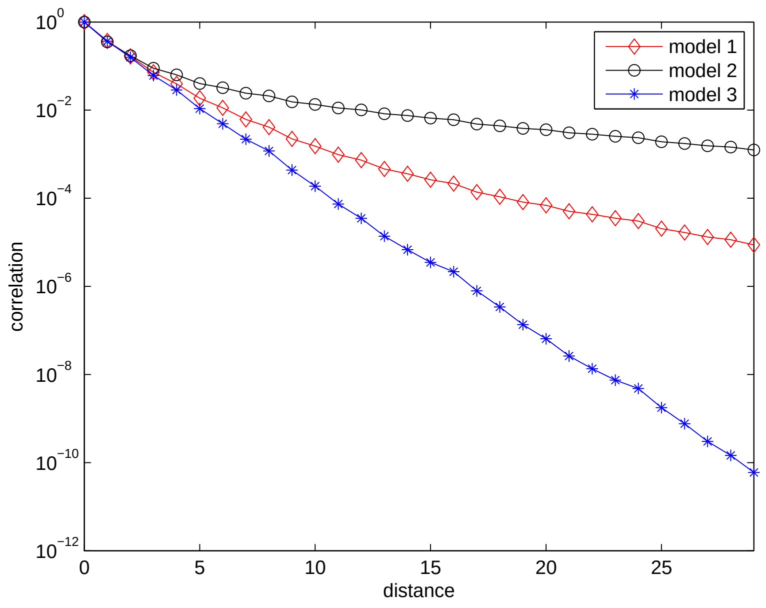


Figure 3-5: Different correlation decays at the finest scale realized by pyramidal graphs with four scales. model 1 : $\alpha = [1/64, 1/16, 1/4, 1]$, $\beta = [1/32, 1/8, 1/2]$; model 2 : $\alpha = [1, 1, 1, 1]$, $\beta = [1, 1, 1]$; model 3 : $\alpha = [0.0001, 0.0001, 0.0001, 1]$, $\beta = [1/32, 1/8, 1/2]$.

by only passing messages between nearby nodes. This is reminiscent of multipole algorithms in which far-away effects are considered at coarser scales and the finer scales only compute interactions among nearby regions. We may take a step further and efficiently construct a pyramidal structure which combines a quadtree and an FIR model in each scale. It is not straightforward, however, to model this structure in graphical model framework, so we leave it as one of our future research topics.

Now, let us change the parameters of the pyramidal graph and observe the resulting marginal and conditional covariance at the finest scale. Figure 3-5 shows several different types of correlation decays all realized by pyramidal graphs. Model 1 corresponds to the model considered so far with parameters defined in (3.10) with $\varphi = 1$. If we do not decrease the parameters at coarser scales and set $\alpha_m = 1$, and $\beta_m = 1$ for all scales, the correlation at the finest scale decays even more slowly. If we want to realize exponential decaying correlations as in monoscale thin-membrane models, we leave β in the original model unchanged, but make α_m at coarser scales extremely small (0.0001 for the model 3 in the plot). This demonstrates that by adjusting the parameters in pyramidal graphs, we can capture long-range correlations as well as fast-decaying correlations.

Chapter 4

Inference on the Pyramidal Graph

In this chapter, we introduce efficient inference algorithms for models on the pyramidal graph defined in Chapter 3. We begin in Section 4.1 with a brief introduction to the multipole algorithm in computational physics. Then, we propose a class of multipole-motivated inference algorithms which is guaranteed to converge because of the walk-summability of the pyramidal graphical model. In Section 4.2, we decompose the pyramidal graph into a quadtree and disconnected chains at each resolution and use the Lagrangian Relaxation method, introduced in Section 2.2.3, to get estimates of conditional means and upper bounds on variances using inference algorithms on each subgraph. We focus on estimating approximate variances in Section 4.3, and apply the low-rank approximation methods (see Section 2.2.4) to our pyramidal graph. The re-estimation problem is discussed in Section 4.4, and we conclude this chapter with experimental results in Section 4.5.

4.1 Multipole-motivated Approach

Multipole algorithms use multiple scales in order to reduce computational complexity, but in a different context from multigrid methods. Instead of using coarser scale estimates as an initial estimate at the finer scale, it is assumed that far-field effects are captured by coarser scales, and each fine scale only computes effects due to nearby nodes. Although multipole algorithms were not developed in the graphical

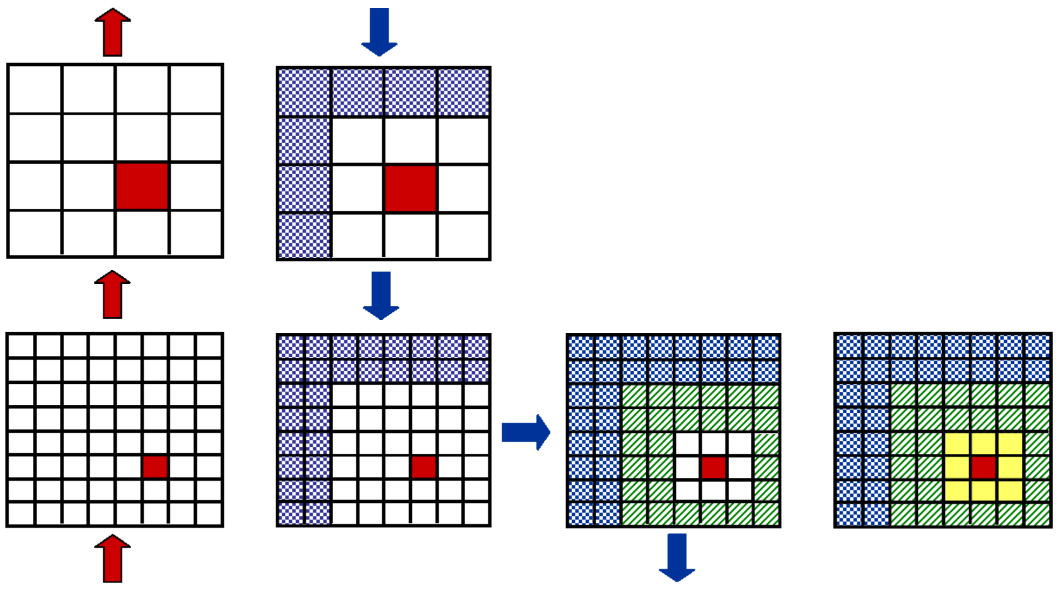


Figure 4-1: Illustration of the multipole algorithm. (a) Upward pass. (b) Downward pass. (c) At the finest scale.

model framework, we can adopt the basic idea which performs only local operations within each scale. This approximation is based on our observation from Chapter 3 that conditioned on adjacent scales, the correlation of one scale decays quickly. We first develop a simple iterative algorithm in which the order of inference steps follow the spirit of multipole algorithms, and then extend the idea to more sophisticated iterations using adaptive ET algorithms.

4.1.1 Multipole Algorithm

The multipole algorithm [18] was developed to evaluate potentials due to distributions of charges. Assume that charges are located at $u_1, u_2, \cdots u_m$ and we are interested in calculating potentials at locations $v_1, v_2, \cdots v_n$ far away from u_i's. If we compute all pairwise interactions, it will take $\mathcal{O}(mn)$ computations. However, when the u_i's and v_j's are located far apart from each other, we may approximate the potentials by clustering u_i's and computing the influence of the cluster $\{u_i\}$ to each of v_j. In this way, we can reduce the computation to $\mathcal{O}(m + n)$.

Figure 4-1 illustrates the upward-downward algorithm to approximate far-field potentials. During the upward pass (Figure 4-1(a)), each region calculates its local potential and passes the message upward. The coarser scale region collects the local potentials of its children to compute its own local potential.

When the algorithm reaches the coarsest scale, it starts the downward pass (Figure 4-1(b)). When a region gets a message from its parent, it assumes that all the necessary information about far-field regions are included in the message. For example, in Figure 4-1(b), the colored region in the finer scale gets a message from its parent (the colored region in the coarser scale) and ignores the dotted areas. Then, it computes the potentials due to the areas with slanted lines and passes the message to its children. When the algorithm reaches the finest scale, each region calculates potentials due to its nearest neighbors and adds them to the far-field potentials computed at the coarser scales. Usually, these estimates are used then as preconditioners for other iterative algorithms.

4.1.2 Multipole-motivated Inference Algorithm

Computing the optimal estimates of a Gaussian process given measurements $y = Cx + v$ is equivalent to solving the linear equation:

$$(J_{prior} + C^T R^{-1} C)\hat{x} = h \tag{4.1}$$

where R is the covariance matrix of the noise process v, and $h = C^T R^{-1} y$. Since the pyramidal graph has measurements only at the finest scale nodes, the matrix $C^T R^{-1} C$ and the vector h have entries corresponding to coarser scale nodes equal to zero:

$$C^T R^{-1} C = \begin{pmatrix} 0 & & & 0 \\ & \ddots & & \vdots \\ & & 0 & 0 \\ 0 & \cdots & 0 & C_M^T R^{-1} C_M \end{pmatrix} \qquad h = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ C_M R^{-1} y \end{pmatrix} \tag{4.2}$$

where C_M is the matrix which maps the finest scale nodes to measurements: $y =$

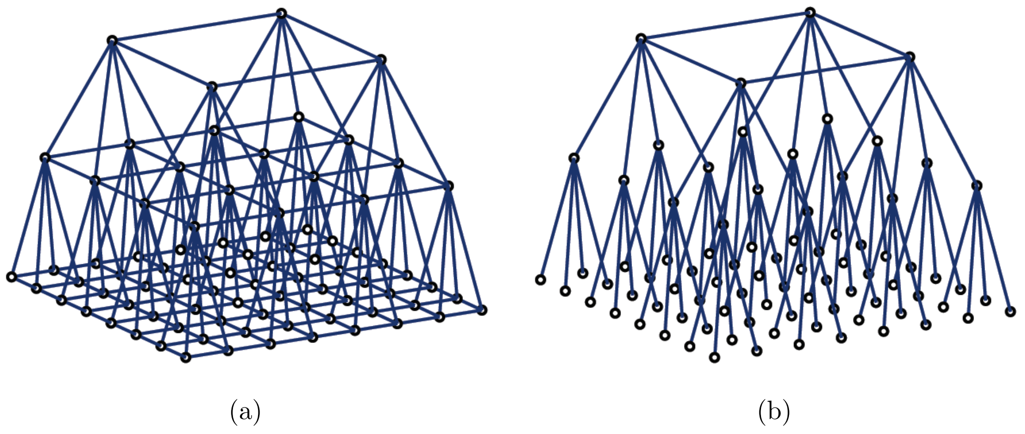


Figure 4-2: A pyramidal graphical model with multiple nodes at the coarsest scale. (a) A pyramidal graph. (b) An embedded subgraph used for the initialization step of the multipole-motivated inference.

$C_M x_M + v$. We use h_m to denote the subvector of h corresponding to scale m, even though $h_m = 0$ for all $m \neq M$.

Remember that the pyramidal graph with the prior defined in Chapter 3 is walk-summable. Therefore, we may use any combination of embedded subgraph iterations introduced in Section 2.2.1, and the algorithm will eventually converge to the solution $\hat{x}$ of (4.1). However, to speed up the convergence, we design an inference algorithm to guide the order of subgraphs to follow the spirit of multipole algorithms.

Note that the main objective in the upward pass of the multipole algorithm is to collect local information and to pass it upward to coarser scales. Since the coarser scale nodes in the pyramidal graph do not have any measurements at the beginning, we need to distribute the measurement information to coarser scales rapidly. At this stage, we are not interested in getting smooth estimates at each scale, so we assume that our prior model has only the J_t component and ignore J_s in (3.1) except at the coarsest scale. While the pyramidal graph in Figure 3-1 has one node at the coarsest scale, we may stop the pyramid at a scale below this coarsest scale, i.e. one with multiple nodes as in Figure 4-2(a). Even in this case, the number of variables at the coarsest scale is often small enough to make exact, global inference possible. For example, for a pyramidal graph with 5 scales, the number of nodes at the coarsest scale is reduced by 1024 compared to the finest scale. Therefore, we use the subgraph shown in Figure 4-2(b) and use the J_t matrix and the $J_{s[1,1]}$ matrix as a prior to get rough initial estimates.

$$\hat{x}^{(0)} = (J_t + [J_{s[1,1]}] + C^T R^{-1} C)^{-1} h \tag{4.3}$$

where the notation $[J_{s[1,1]}]$ means that $J_{s[1,1]}$ is zero-padded to an $N \times N$ matrix as follows:

$$[J_{s[1,1]}] = \begin{pmatrix} J_{s[1,1]} & 0 & \cdots & 0 \\ 0 & 0 & & \\ \vdots & & \ddots & \\ 0 & & & 0 \end{pmatrix} \tag{4.4}$$

Starting with the initial estimates in (4.3), we perform a coarse-to-fine sweep (downward pass) to smooth the estimates within each scale. Since different scales in the pyramidal graph are statistically connected to each other, the changes at finer scales affect the nodes at coarser scales. Therefore, we need to perform the upward- and downward- pass multiple times before the iterative inference algorithm converges. Let $\hat{x}^{d(2n-1)}$ be the estimates computed by the downward pass at iteration $(2n-1)$, and let $\hat{x}^{u(2n)}$ be the estimates computed by the upward pass at iteration $(2n)$. Let $x^{u(0)} = x^{(0)}$.

As the equivalent step to the downward pass in the multipole algorithm, we perform some in-scale operations within each scale starting from the coarsest scale and proceeding downward. At scale m, we first get messages from adjacent scales and perform inference within scale m. This is basically a block Gauss-Seidel iteration introduced in Section 2.2.1 with the nodes at scale m being the nodes to be updated.

$$\hat{x}_m^{d(2n-1)} = J_{[m,m]}^{-1}(h_m - J_{[m,m-1]} \cdot \hat{x}_{m-1}^{d(2n-1)} - J_{[m,m+1]} \cdot \hat{x}_{m+1}^{u(2n-2)}) \tag{4.5}$$

Since inverting $J_{[m,m]}$ is not tractable for finer scales with a large number of nodes, we apply a hybrid of ET and block Gauss-Seidel iterations.

$$\hat{x}_m^{d(2n-1)} = J_{S_n}^{-1}(h_m - K_{S_n} \cdot \hat{x}_m^{u(2n-2)} - J_{[m,m-1]} \cdot \hat{x}_{m-1}^{d(2n-1)} - J_{[m,m+1]} \cdot \hat{x}_{m+1}^{u(2n-2)}) \tag{4.6}$$

1. **Initialization:** get initial estimates based on the tree prior and the thin-membrane prior within the coarsest scale.
2. **In-scale inference:** starting from the coarsest scale and proceeding to finer scales, smooth the estimates using a single Gauss-Jacobi iteration within each scale.
3. **Tree inference:** apply one ET iteration using the embedded quadtree.
4. Repeat the in-scale inference and tree inference steps until a stopping criterion is met.

Table 4.1: Multipole-motivated inference algorithm using the quadtree and Gauss-Jacobi iterations.

where S_n is a tractable subgraph embedded in $\mathcal{G}_n = (V_m, \mathcal{E}_{V_m})$ and $K_{S_n} = J_{[m,m]} - J_{S_n}$.

Recall from Chapter 3 that in the pyramidal graph, the correlations within one scale decay quickly with distance once conditioned on adjacent scales. Therefore, within each scale, we may ignore messages from far-away nodes and only pass messages among nearby nodes. In the multipole algorithm context, this can be interpreted as computing potentials from nearby nodes at each scale, assuming that far-field effects are captured at coarser scales. Therefore, we choose the subgraph S_n to be a fully disconnected graph at each scale. This is essentially applying a single Gauss-Jacobi iteration within each scale. So, J_{S_n} is a diagonal matrix with entries taken from $J_{[m,m]}$:

$$(J_{S_n})_{ij} = \begin{cases} (J_{[m,m]})_{ij} & \text{if } \ i = j \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

For each fine-to-coarse sweep, or the upward pass, we use the quadtree structure connecting different scales. Although it is sufficient for this step to pass messages upward, to facilitate convergence analysis, we pass messages both upward *and* downward to perform exact inference on the quadtree. Then, this step is equivalent to applying one ET iteration. Let J_n be defined as the associated J matrix corresponding to the quadtree as follows:

$$(J_n)_{ij} = \begin{cases} (J)_{ij} & \text{if} \quad i = j \quad \text{or} \quad j \in \mathcal{C}(i) \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

where $\mathcal{C}(i)$ is the set of child nodes of i, i.e. the neighbors at the next finer scale. An equivalent representation is $J_n = J_t + diag(J_s) + C^T R^{-1} C$, where $diag(J_s)$ is a diagonal matrix with entries taken from J_s. Let $K_n = J - J_n$. Then, the ET iteration using the quadtree structure can be represented as:

$$\hat{x}^{u(2n)} = J_n^{-1}(h - K_n \hat{x}^{d(2n-1)}) \tag{4.9}$$

The algorithm is summarized in Table 4.1. Since the pyramidal model is walk-summable, and since every edge is included infinitely often in either the *tree inference* or the *in-scale inference* step, the iteration converges for any combination of the two inference steps.

4.1.3 Adaptive Iterations

In the previous section, we used Gauss-Jacobi iterations for the in-scale inference steps and the quadtree for the tree inference steps. In order to reduce the number of iterations, we apply the adaptive ET iterations developed by Chandrasekaran *et al.* [6] to adaptively choose these subgraphs to reduce the estimation error quickly. Let the error at iteration n be $e^{(n)} = \hat{x} - \hat{x}^{(n)}$ and the residual error be $h^{(n)} = h - J\hat{x}^{(n)}$. In [6], it is shown that minimizing a loose upper bound of the error $\parallel e^{(n)} \parallel_{\ell_1}$ is equivalent to solving the maximum spanning tree problem:

$$\arg \max_{S_n \text{ a tree}} \sum_{(i,j) \in S_n} \omega_{ij} \tag{4.10}$$

1. **Initialization:** get initial estimates based on the tree prior and the thin-membrane prior within the coarsest scale.
2. **In-scale inference:** starting from the coarsest scale and proceeding to finer scales, smooth the estimates using an adaptively chosen spanning tree within each scale.
3. **Tree inference:** choose a spanning tree of the pyramidal graph using the adaptive ET algorithm, and apply one ET iteration.
4. Repeat the in-scale inference and tree inference steps until a stopping criterion is met.

Table 4.2: Multipole-motivated inference algorithm using the adaptive ET iterations.

where ω_{ij} is the absolute walk-sum over the walks that live solely on edge (i, j) re-weighted by the absolute residual of nodes i and j:

$$\omega_{ij} = \frac{|r_{ij}|}{1 - |r_{ij}|} \cdot (|h_i^{(n-1)}| + |h_j^{(n-1)}|) \tag{4.11}$$

where r_{ij} is the partial correlation coefficient between x_i and x_j as defined in (2.21).

In principle, we may treat the pyramidal graph as a general graphical model and apply the adaptive ET iterations without any guidance to utilize the hierarchical structure. However, since the adaptive algorithm chooses a spanning tree in a greedy fashion, it tends to focus on the edges within the finest scale in which the residual errors of the nodes are usually largest. We need to force the algorithm to utilize coarser scale structure more actively, which may not reduce the error immediately at the next step, but eventually will lead to faster convergence. So, we again use the combination of global and in-scale computations to speed up the convergence. In other words, we use the same upward-downward procedure as in the previous section, but replace both the Gauss-Jacobi iteration and the quadtree with spanning trees chosen by adaptive ET algorithms. The algorithm is summarized in Table 4.2. Again, from the walk-summability of the pyramidal graph, this procedure is guaranteed that the iteration converges.

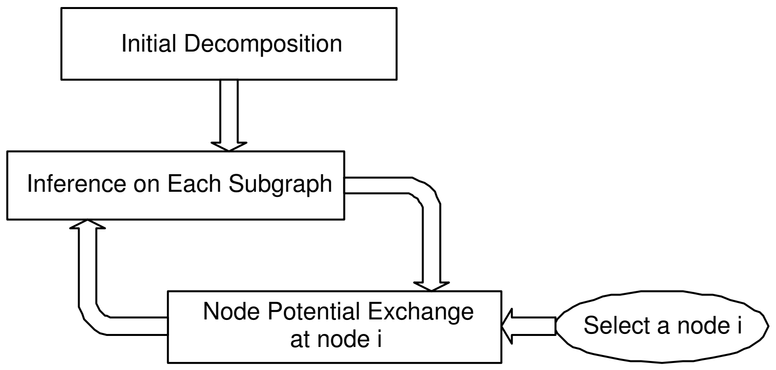


Figure 4-3: A block diagram of the Lagrangian relaxation method. A more efficient implementation is illustrate in Figure 4-4.

4.2 Lagrangian Relaxation Methods

In Section 2.2.3, we introduced the Lagrangian relaxation method which decomposes an intractable graph into tractable subgraphs and computes the approximate moments of the original graph by identifying a valid decomposition which satisfies the reweighted moment-matching conditions in (2.29). In this section, we describe an iterative method developed by Johnson [21] to solve the reweighted moment-matching conditions, and apply it to our pyramidal graph.

Consider a graphical model $\mathcal{G} = (V, \mathcal{E})$ with associated information matrix J and potential vector h. Assuming that inference on the original graph is intractable, we decompose $\mathcal{G}$ into tractable subgraphs $\mathcal{G}^k = (V^k, \mathcal{E}^k)$. These subgraphs may share nodes, edges, or cliques, but we focus our attention here on the case when the subgraphs only have common nodes as shown in Figure 4-6. Consider a valid initial decomposition $\{J^k\}$ and $\{h^k\}$ such that $J = \sum_k J^k$, $h = \sum_k h^k$, and $J^k \succ 0$ for all k. We alternately solve inference problems on each subgraph and then modify the decomposition to force the reweighted moment-matching condition at a single node. Note that when we perform inference again on each subgraph using the modified decomposition, the reweighted moment-matching condition will not be satisfied at any other node in general. So we iteratively cycle through all of the nodes in the original graph. This procedure is illustrated in the block diagram in Figure 4-3.

Inference on Each Subgraph We perform inference on each subgraph $\mathcal{G}^k = (V^k, \mathcal{E}^k)$, and compute the marginal statistics $\hat{J}_i^k \triangleq (P_i^k)^{-1}$ and $\hat{h}_i^k \triangleq \hat{J}_i^k \hat{x}_i^k$ at node i.

Node Potential Exchange For each subgraph $\mathcal{G}^k$, we update the node potentials J_{ii}^k and h_i^k by exchanging potentials among the subgraphs which share the same node:

$$\begin{aligned} J_{ii}^k &\leftarrow J_{ii}^k + (\rho_k(\sum_k \hat{J}_i^k) - \hat{J}_i^k) \\ h_i^k &\leftarrow h_i^k + (\rho_k(\sum_k \hat{h}_i^k) - \hat{h}_i^k) \end{aligned} \tag{4.12}$$

where the sum is over k such that $i \in V^k$. The weight ρ_k of each subgraph $\mathcal{G}^k$ is given as the function of J^k as in (2.27).

Theorem 4.1. *After the node potential exchange step at node i, the following conditions are satisfied.*

1. *All subgraphs are valid: $J^k \succ 0$ for all k.*
2. $J = \sum_k J^k$, $h = \sum_k h^k$
3. *When inference is again performed on each subgraph with the modified decomposition, the re-weighted moment-matching condition is satisfied at node i, i.e. $(\hat{J}_i^k)^{-1}\hat{h}_i^k$ and $\rho_k(\hat{J}_i^k)^{-1}$ are constants independent of k.*

Proof. See [21]. □

In order to find the valid decomposition which satisfies the reweighted moment-matching conditions at all nodes, we iteratively cycle through all of the nodes of the original graph. The order of nodes can be specified arbitrarily as long as each node is revisited during each cycle. Then, the algorithm is guaranteed to converge as stated in the following theorem.

Theorem 4.2. *If we follow the procedure in the block diagram in Figure 4-3 and iterate through all of the nodes of the original graph, the node potentials at all nodes*

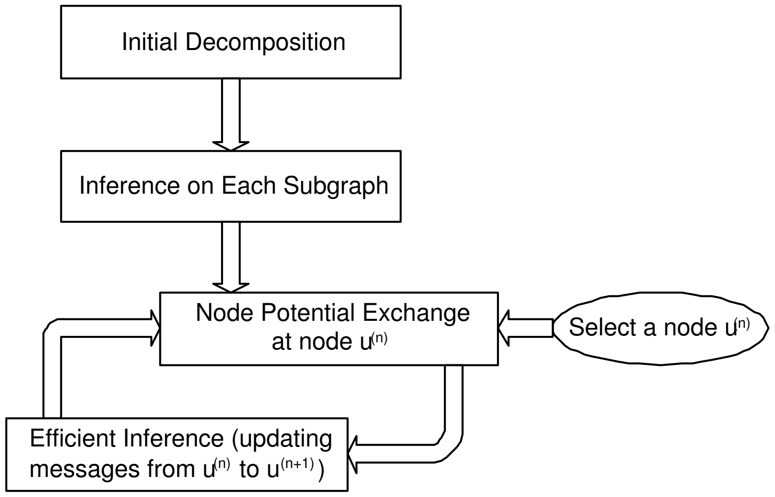

Initial Decomposition
Inference on Each Subgraph
Node Potential Exchange at node u(n)
Select a node u(n)
Efficient Inference (updating messages from u(n) to u(n+1))

Figure 4-4: A block diagram of the efficient implementation of the Lagrangian relaxation method. $u^{(n)}$ is a single node in V and $u^{(n+1)}$ is the node to be updated next.

converge to fixed points. After the convergence, the pseudo-moments of the original graph at each node $i \in V$ can be computed as follows:

$$\begin{aligned} \tilde{x}_i &= (\hat{J}_i^k)^{-1}\hat{h}_i^k \\ \tilde{P}_i &= \rho_k(\hat{J}_i^k)^{-1} \end{aligned} \tag{4.13}$$

In addition, $\tilde{x}_i$ converges to the correct conditional mean $\hat{x}_i$, and $\tilde{P}_i$ is an upper bound on the true covariance P_i.

Proof. See [21]. □

In principle, after updating each node potential in each subgraph, we need to propagate the changes in the messages to all other nodes in each subgraph so that the other nodes can compute correct marginal statistics before updating their own node potentials. However, this procedure takes $\mathcal{O}(N)$ computation for the update of each node potential. Instead, we design an efficient message-passing algorithm to update only a subset of messages after the update of each node potential as illustrated in the block diagram in Figure 4-4. Kolmogorov [30] proposed a similar idea for an efficient implementation of tree-reweighted message passing algorithms [53], but he only considered the case when each subgraph is a first-order chain model and when

1. Initialization
    (a) Decompose J and h into a set of subgraphs $\mathcal{G}^k = (V^k, \mathcal{E}^k)$ and associated $J^k \succ 0$ and h^k such that $J = \sum_k J^k$ and $h = \sum_k h^k$.
    (b) Run BP on each subgraph until convergence.
2. Specify an ordering of nodes $\{u^{(1)}, u^{(2)}, \ldots, u^{(N)}\}$, where $u^{(n)}$ is a single node in V. For $n = 1, 2, \ldots N$, at sub-iteration n:
    (a) For all subgraphs $\mathcal{G}^k$ such that $u^{(n)} \in V^k$, compute the marginal statistics $\hat{J}^k_{u^{(n)}}$ and $\hat{h}^k_{u^{(n)}}$ using (4.14).
    (b) Update the node potentials of $u^{(n)}$ in each subgraph using (4.12).
    (c) Update the messages on the path from $u^{(n)}$ to $u^{(n+1)}$.
3. Update the weights ρ_k's using (2.27).
4. Repeat Step 2 and 3 until convergence.
5. Compute the pseudo-moments of the original graph using (4.13).

Table 4.3: Lagrangian relaxation methods with the efficient message update scheme.

the nodes are updated sequentially from one end of the chain to the other end.

We begin with a formal definition of *up-to-date messages*. Let $\mathcal{T}_{j\setminus i}$ denote the subtree rooted at j away from i. If the message $m_{j\to i} \triangleq \exp\left(-\frac{1}{2}\Delta J_{j\to i} x_j^2 + \Delta h_{j\to i} x_j\right)$ contains all the necessary information from $\mathcal{T}_{j\setminus i}$, a node i in a tree can compute its exact marginal statistics by combining messages and its local potentials [37]:

$$\begin{aligned} \hat{J}_i &= J_{ii} + \sum_{j\in\mathcal{N}(i)} \Delta J_{j\to i} \\ \hat{h}_i &= h_i + \sum_{j\in\mathcal{N}(i)} \Delta h_{j\to i} \end{aligned} \tag{4.14}$$

Therefore, when a node potential of node $k \in \mathcal{T}_{j\setminus i}$ is changed, the message $m_{j\to i}$ needs to be updated to compute the correct marginal statistics at node i.

Definition 4.3. *A message $m_{j\to i}$ is **up-to-date** if it represents the result of eliminating all variables in $\mathcal{T}_{j\setminus i}$.*

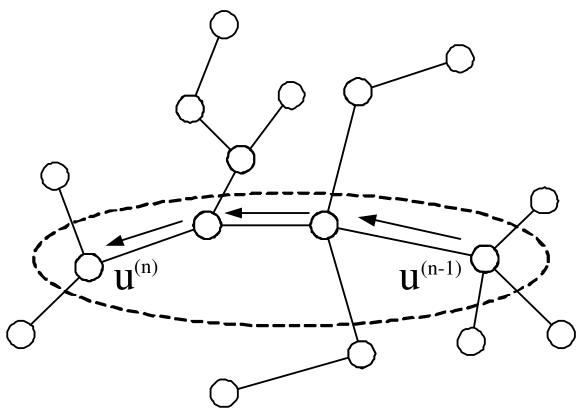


Figure 4-5: An illustration of Theorem 4.4.

Table 4.3 presents the Lagrangian Relaxation method with an efficient message passing scheme which can be applied to the case when each subgraph is cycle-free and shares only nodes with other subgraphs. After the initial decomposition, we run the Belief Propagation (BP) algorithm [40] until convergence in each subgraph. Note that after BP converges in a tree-structured graph, every message is up-to-date.

Let N be the number of nodes in the original graph. At sub-iteration n, we update the node potential of a single node $u^{(n)} \in V$. Let us specify an ordering of nodes $\{u^{(n)}\}$ in which each node is included once. We define one iteration as one cycle of N sub-iterations in which the potentials of every node are updated exactly once. After one iteration, we specify a new ordering of nodes and repeat the N sub-iterations.

The following theorem guarantees that we can indeed compute the correct marginal statistics at node $u^{(n)}$ by only updating messages on the path from $u^{(n-1)}$ to $u^{(n)}$.

Theorem 4.4. *Using the message passing scheme in Table 4.3, every incoming message for node $u^{(n)}$ is up-to-date when the marginal statistics at node $u^{(n)}$ are computed.*

We describe a simple illustration here and provide a detailed proof in Appendix A. Figure 4-5 shows one subgraph with the two nodes $u^{(n-1)}$ and $u^{(n)}$ to be updated at sub-iteration $(n-1)$ and n, respectively. If $u^{(n-1)}$ had all incoming messages up-to-date at sub-iteration $(n-1)$, then after the node potential of $u^{(n-1)}$ is modified, all

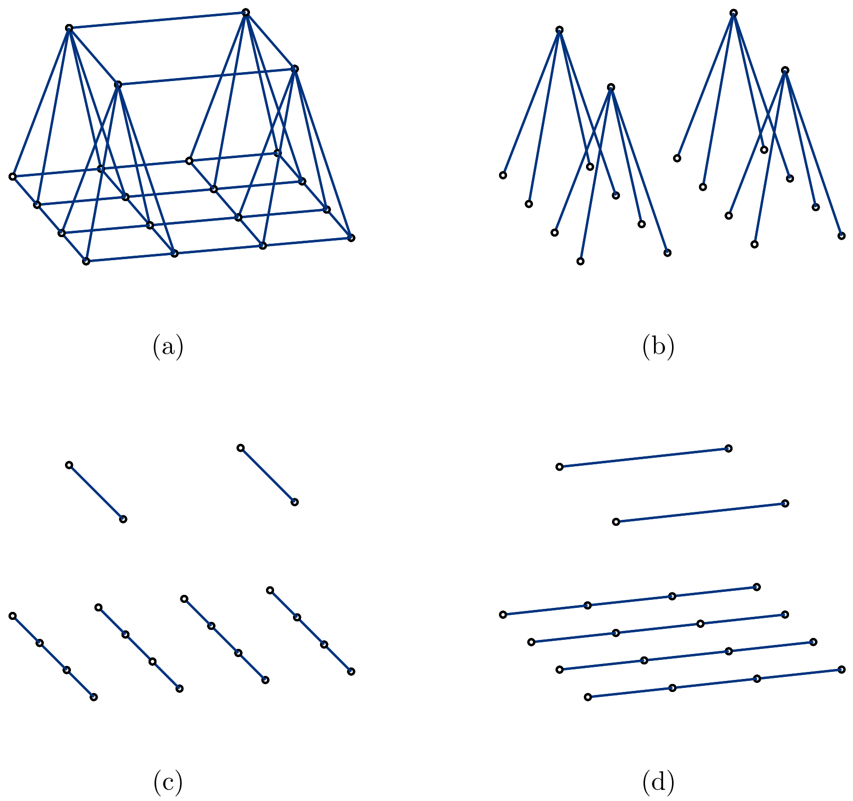


Figure 4-6: A pyramidal graphical model and its decomposition into subgraphs. (a) The original pyramidal graph with two scales. (b) Subgraph 1 : Disconnected quadtrees. (c) Subgraph 2 : Vertical chains. (d) Subgraph 3 : Horizontal chains.

messages coming from outside of the circled area still remain up-to-date. Therefore, if we update the messages on the path from $u^{(n-1)}$ to $u^{(n)}$, the node $u^{(n)}$ has all incoming messages up-to-date.

Figure 4-6 shows one natural decomposition of the pyramidal graph into three subgraphs. We first decompose the pyramidal graph into the quadtrees and separated multiple grid models, and then further decompose the grid models into vertical and horizontal chains. The quadtrees and coarser scale nodes provides paths through which far-away nodes can communicate rapidly, so the subgraphs can exchange potentials more efficiently than in the monoscale counterpart. Note that this decomposition enables a simple implementation of the Lagrangian relaxation method, since all subgraphs are trees and only share nodes with each other, as assumed in the analysis of this section. However, we may construct other subgraphs with more complicated

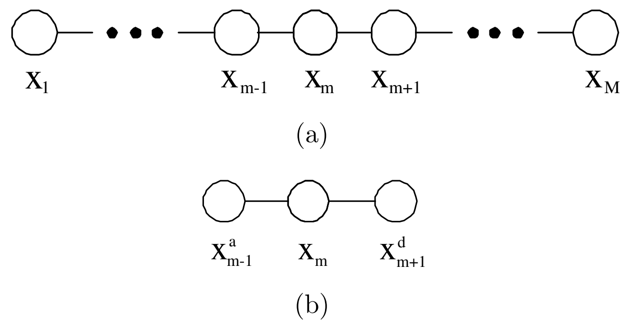


Figure 4-7: An illustration of the pyramidal graph as a Markov chain model. (a) A first-order Markov chain in which each node represents one scale in the pyramidal graph. (b) A reduced chain in which all nodes except the node $m-1$, m, $m+1$ are eliminated from (a).

structure in order to get faster convergence, and the analysis in this section can be extended to the case when subgraphs share edges. In addition, note that the convergence of this algorithm is not based on the walk-summability of the model as in embedded subgraph algorithms. Therefore, we may use other prior models for the pyramidal graph (such as a thin-plate model for each scale) and still be able to prove convergence.

4.3 Low-rank Variance Approximation Algorithms

The diagonal elements of the error covariance matrix P correspond to the uncertainties in the estimates at each node, and provide valuable information. For example, from the estimates of error variances, we may detect regions in which the prior model should be modified or more measurements need to be taken (see Section 4.4). Therefore, we wish to get approximate values of the diagonal elements of P, which, of course, satisfies $J \cdot P = I$, where I is an identity matrix, without inverting J. We decompose the J and P matrix by scale as in (3.6), and let $J_{[i,j]}$ and $P_{[i,j]}$ denote the sub-matrix of J and P respectively, corresponding to scale i and scale j. Notice that even though $J_{[i,m]}$ is zero except for $i = m-1, m, m+1$, the covariance matrix P is usually a full matrix.

Theorem 4.5. *The marginal covariance of nodes at scale m can be represented as* *follows:*

$$P_{[m,m]} = \left(J_{[m,m]}\right)^{-1} + \left(J_{[m,m]}\right)^{-1} Q_m \left(J_{[m,m]}\right)^{-1} \tag{4.15}$$

where

$$Q_m = \begin{cases} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix}^T \begin{pmatrix} P_{[m-1,m-1]} & P_{[m-1,m+1]} \\ P_{[m+1,m-1]} & P_{[m+1,m+1]} \end{pmatrix} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix} & 1 < m < M \\ J_{[1,2]}\ P_{[2,2]}\ J_{[2,1]} & m = 1 \\ J_{[M,M-1]}\ P_{[M-1,M-1]}\ J_{[M-1,M]} & m = M \end{cases} \tag{4.16}$$

Proof. Consider representing our pyramidal model as a first-order Markov chain model as shown in Figure 4-7(a), in which each node m is associated with x_m, the collection of all variables at scale m. We eliminate all the coarser nodes from 1 to $m-2$ sequentially and denote the resulting vector at node $m-1$ as x^a_{m-1}, which corresponds to the ancestors of nodes at scale m. Similarly, we eliminate all finer nodes and let x^d_{m+1} represent the descendants of nodes at scale m. Figure 4-7(b) shows the reduced chain model. Let $J^a_{[m-1]}$ and $J^d_{[m+1]}$ denote the sub-matrix of the reduced J matrix associated with x^a_{m-1} and x^d_{m+1}, respectively, and let $\hat{J}_{[m]}$ denote the marginal J matrix for scale m when nodes at all other scales are eliminated. Using Gaussian elimination [37], the marginal J matrix for scale m is given by

$$\begin{aligned} \hat{J}_{[m]} &= J_{[m,m]} - J_{[m,m-1]} \left(J^a_{[m-1]}\right)^{-1} J_{[m-1,m]} - J_{[m,m+1]} \left(J^d_{[m+1]}\right)^{-1} J_{[m+1,m]} \\ &= J_{[m,m]} - \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix}^T \begin{pmatrix} \left(J^a_{[m-1]}\right)^{-1} & 0 \\ 0 & \left(J^d_{[m+1]}\right)^{-1} \end{pmatrix} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix} \end{aligned}$$

Using the Woodbury Identity [41],

$$\left(\hat{J}_{[m]}\right)^{-1} = \left(J_{[m,m]}\right)^{-1} + \left(J_{[m,m]}\right)^{-1} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix}^T A^{-1} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix} \left(J_{[m,m]}\right)^{-1} \tag{4.17}$$

where A is the $(N_{m-1} + N_{m+1}) \times (N_{m-1} + N_{m+1})$ matrix:

$$A = \begin{pmatrix} J^a_{[m-1]} & 0 \\ 0 & J^d_{[m+1]} \end{pmatrix} - \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix} \left(J_{[m,m]}\right)^{-1} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix}^T . \qquad (4.18)$$

The above equation corresponds exactly to the Gaussian elimination step of eliminating node m from the chain model in Figure 4-7(b). So, the inverse of A is simply the marginal covariance of the nodes at scale $m-1$ and scale $m+1$ when we eliminated nodes at all other scales. Substituting A in (4.17), and denoting the marginal covariance of scale m as $P_{[m,m]} \triangleq \left(\hat{J}_{[m]}\right)^{-1}$, we get the expression in (4.15) for the covariance of nodes at scale m. □

Evaluating the right side of (4.15) is still intractable since we need to invert $J_{[m,m]}$ and compute the joint covariance matrix of scales $m-1$ and $m+1$. So, we focus on computing lower bounds on the diagonal elements of $P_{[m,m]}$. Let p_{ii} denote the i^{th} diagonal element of $P_{[m,m]}$, which corresponds to the error variance of node i at scale m.

In Section 3.2, we defined the conditional covariance $\bar{P}_{[m,m]} \triangleq \left(J_{[m,m]}\right)^{-1}$ at scale m conditioned on its parent scale $m-1$ and and its child scale $m+1$. We denote the (i,j) entry of $\bar{P}_{[m,m]}$ as $\bar{p}_{ij}$, which corresponds to the conditional covariance between node $i \in V_m$ and node $j \in V_m$ conditioned on nodes at the coarser scale $m-1$ and the finer scale $m+1$. Then, we consider the following lower bound on the error variance at node i:

$$\begin{aligned} p_{ii} &= \bar{p}_{ii} + \sum_{j \in V_m} \sum_{k \in V_m} \bar{p}_{ij} \cdot \bar{p}_{ik} \cdot (Q_m)_{jk} \\ &> \bar{p}_{ii} + \bar{p}_{ii} \cdot \bar{p}_{ii} \cdot (Q_m)_{ii} + \sum_{j \in \mathcal{N}_m(i)} \sum_{k \in \mathcal{N}_m(i)} \bar{p}_{ij} \cdot \bar{p}_{ik} \cdot (Q_m)_{jk} \qquad (4.19) \end{aligned}$$

where $\mathcal{N}_m(i)$ is the set of neighboring nodes of i within scale m. The first equality follows from (4.15), and the inequality follows from the fact that the pyramidal graph is an attractive model. Since the partial correlation coefficient for every pair of node is positive, every walk-sum is positive (see Section 2.2.2), and the covariance, as well as the conditional covariance between any pair of node is positive. It is also straightforward to show that every element of the matrix Q_m is also nonnegative.

We have seen in Figure 3-3(b) that conditional covariances decay quickly, so $\bar{p}_{ij}$ becomes very small when i and j are not neighbors. Therefore, although the lower bound in (4.19) is not tight, it closely approximates the true value. In addition, we can estimate $\bar{p}_{ij}$ rapidly for $j = i$ or $j \in \mathcal{N}_m(i)$ using the low-rank variance approximation algorithm [36] introduced in Section 2.2.4. The algorithm computes approximate covariances rapidly for models with exponentially decaying correlations, and the shorter the correlation length is, the more accuracy the algorithm guarantees.

Now, let's consider computing the matrix Q_m defined in (4.16). The $N_m \times (N_{m-1} + N_{m+1})$ matrix $\begin{pmatrix} J_{[m,m-1]} & J_{[m,m+1]} \end{pmatrix}$ is a sparse matrix with only 5 nonzero elements at each row. The middle component of (4.16), the 2×2 block matrix, is the full joint covariance matrix of scales $m-1$ and $m+1$ which is difficult to compute and to store. Since we are only interested in computing a subset of elements of Q_m, as a first approach, we further relax the lower bound in (4.19), and approximate the joint covariance matrix of scales $m-1$ and $m+1$ with a diagonal matrix. Then, the approximate variances can be computed iteratively using coarse-to-fine sweeps. Let $\rho_i^{(n)}$ denote the approximate variance at node $i \in V_m$ computed at n^{th} coarse-to-fine sweep, then from (4.19),

$$\rho_i^{(n)} = \bar{p}_{ii} + \bar{p}_{ii} \cdot \bar{p}_{ii} \cdot (\tilde{Q}_m^{(n)})_{ii} + \sum_{j \in \mathcal{N}_m(i)} \sum_{k \in \mathcal{N}_m(i)} \bar{p}_{ij} \cdot \bar{p}_{ik} \cdot (\tilde{Q}_m^{(n)})_{jk} \tag{4.20}$$

where $\tilde{Q}_m^{(n)}$ is defined as follows:

$$\tilde{Q}_m^{(n)} = \begin{cases} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix}^T \begin{pmatrix} \Upsilon_{[m-1]}^{(n)} & 0 \\ 0 & \Upsilon_{[m+1]}^{(n-1)} \end{pmatrix} \begin{pmatrix} J_{[m-1,m]} \\ J_{[m+1,m]} \end{pmatrix} & 1 < m < M \\ J_{[M,M-1]} \, \Upsilon_{[M-1]}^{(n)} \, J_{[M-1,M]} & m = M \end{cases} \tag{4.21}$$

$\Upsilon_{[m]}^{(n)}$ is a diagonal matrix with i^{th} diagonal element corresponding to the approximate variance at $i \in V_m$ computed at n^{th} coarse-to-fine sweep, i.e. $\rho_i^{(n)}$. It is tractable to

1. Compute the exact variances of nodes at scale 1.
2. For all finer scales, use the low-rank variance approximation algorithm (see Section 2.2.4) to compute conditional covariance $\bar{P}_{[m,m]}$ conditioned on adjacent scales.
3. Initialize the variances of nodes at finer scales as the conditional variances computed at Step 2, i.e. $\rho_i^{(0)} = \bar{p}_{ii}$ for all $i \in V \backslash V_1$.
4. At n^{th} coarse-to-fine sweep, for $m = 2, 3, \ldots, M$:
   (a) Compute $\tilde{Q}_m^{(n)}$ in (4.21) using the approximate variances of nodes at adjacent scales.
   (b) Compute the lower bound on variances $\rho_i^{(n)} < p_{ii}$ for $i \in V_m$ using (4.20).
5. Repeat Step 4 until a stopping criterion is met.

Table 4.4: The coarse-to-fine variance computation using the low-rank approximation algorithm.

compute the variances of the nodes at the coarsest scale exactly, so we define $\Upsilon_{[1]}^{(n)}$ to be a diagonal matrix with entries taken from $P_{[1,1]}$. Table 4.3 summarizes the iterative variance approximation algorithm.

The approximate lower bound computed by the algorithm in Table 4.3 is close to the true error variance when we have dense measurements. However, for a model with sparse measurements, even conditional correlations may have slow decay, and the terms ignored in the lower bound in (4.20) may have significant values. This problem can be resolved by using the wavelet-based approach proposed in [38].

It is well known that wavelet coefficients tend to be less correlated than the original signal. Therefore, instead of constructing the low-rank matrix B (see Section 2.2.4) by combining columns of the identity matrix I, Malioutov *et al.* [38] combine the columns of a wavelet basis matrix W, in which each column corresponds to a wavelet function. In addition, by using wavelet functions at multiple scales, they achieve more compression at finer scales with larger number of nodes, resulting in significant computational savings.

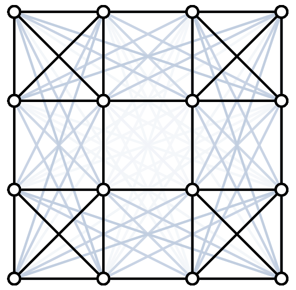


Figure 4-8: The marginal model when the four coarser nodes in Figure 4-6 are eliminated. Edges with larger edgeweights are plotted with darker lines (except for the edges forming the original grid model, which have edgeweights about 10 times larger than the strongest edge newly introduced by the marginalization).

Although this multiscale approach is different from the multiscale modeling considered in this thesis, the algorithm can be applied easily to our pyramidal graph. We provide a brief description here, and refer the reader to [35] for further details. Assume that we are only interested in estimating the covariances at the finest scale nodes. It is easy to design a low-rank matrix for a regular grid model(see [36]), so we let B_M be the spliced wavelet basis for the embedded grid model at scale M. Then, the algorithm can be easily extended to our pyramidal graph by using the matrix $B = (0 \ B_M^T)^T$ with 0 for all coarser scales to compute the approximate variances at the finest scale nodes.

We showed in Figure 3-3(a) that the monoscale thin-membrane model cannot capture long-range correlations, so in order to model slowly decaying correlations using a single scale model, we need to use a more densely connected graph. For example, Figure 4-8 shows the resulting single scale graph when the coarser scale nodes in the pyramidal graph in Figure 4-6 are marginalized out. Note that in general, the low-rank approximation algorithm involves designing a matrix B such that b_i and b_j are orthogonal to each other when i and j are close (see Section 2.2.4). This process is simple for a regular grid model, and can be easily extended to our pyramidal graph, but becomes challenging for a more densely connected graph. Therefore, it is easier to apply the low-rank approximation algorithm on the pyramidal graph than on the single scale model in Figure 4-8.

In addition, the efficiency of this algorithm critically depends on how fast we can solve the inference problem in (2.12). Since algorithms to compute conditional means on the pyramidal graph converge faster than those on the monoscale models (see Section 4.5), we can reduce the computational complexity significantly by using the pyramidal graph.

4.4 Re-estimation

Let us re-state here the problem of computing the estimates of a Gaussian process given measurements and a prior model:

Estimation problem Compute $\hat{x} = J^{-1}h$, where $J = J_{prior} + C^T R^{-1} C$ and $h = h_{prior} + C^T R^{-1} y$.

The re-estimation problem arises when we wish to update the estimates $\hat{x}$ to account for additional local information. In Section 1.2, we introduced two possible scenarios when we need to solve the re-estimation problem. The first case is when new measurements are introduced in a local region. In this case, the measurement vector y and the mapping matrix $C^T R^{-1} C$ are changed from our estimation problem.

The second case is modifying the model locally, for example, to get more accurate estimates around discontinuities. By comparing the difference between measurements and estimates with the value of estimated error variances, we may detect areas in which the assumption of the prior model is seriously violated. Specifically, the estimated error variance has information about the variance of the noise and the sparsity of the measurements. If the difference between the estimates and measurements are much larger than this estimated error variance, it can be assumed that the prior model is not modeling the region accurately. Fieguth *et al.* [14] demonstrated that surface discontinuities can be detected using this method. Alternatively, human experts may analyze the estimates and indicate the local region which needs to be estimated again.

When we detect discontinuities, we may modify our smoothness prior locally to weaken the smoothness constraints. This results in putting relatively more confidence in measurements by reducing the weights of the prior. As a first approach, when the neighboring nodes i and j are on the different side of the discontinuities, we remove the constraint term regarding the two nodes from (3.2) or from (3.5), so the nodes on either side of discontinuities are are not connected with an edge. This procedure corresponds to modifying the J_{prior} matrix.

For either case, the re-estimation problem can be posed as follows:

Re-estimation problem Suppose that we have $\hat{x} = J^{-1}h$. Efficiently compute the updated estimates $\tilde{x} = (J + \Delta J)^{-1}(h + \Delta h)$, where ΔJ and Δh have nonzero elements only in a localized area.

Note that the term *localized* here means relative to the original J and h. For example, if there is a cliff which spans across the entire region as in Figure 1-3, ΔJ may have nonzero elements for all nodes around the cliff. Still, the number of variables affected is on the order of square root of the number of all variables.

We solve the re-estimation problem iteratively by updating a subset of variables at each iteration. Chandrasekaran *et al.* [6] developed adaptive block Gauss-Seidel algorithms to choose the next best subset of k variables to minimize an upper bound of the error $\| \hat{x} - \hat{x}^{(n)} \|_{\ell_1}$. Let $h^{(n)} = h - J\hat{x}^{(n)}$ be the residual error at iteration n and let V_n be the set of nodes to be updated at iteration n.

1. Set $V_n = \varnothing$. For each node, set the node weight as the residual at iteration $(n-1)$:
$$\omega_i = |h_i^{(n-1)}| \tag{4.22}$$

2. Pick the node $i^* \in V \backslash V_n$ with the maximum weights, and set $V_n \leftarrow V_n \cup i^*$.

3. If $|V_n| = k$, stop. Otherwise, update the neighboring nodes of i^* not yet in V_n, i.e. $j \in \mathcal{N}(i^*) \cap V \backslash V_n$:
$$\omega_j \leftarrow \omega_j + \left(|h_{i^*}^{(n-1)}| + |h_j^{(n-1)}|\right) \frac{|r_{i^*j}|}{1 - |r_{i^*j}|} \tag{4.23}$$
Then, go to step 2.

1. Let S be the set of nodes i for which either Δh_i or ΔJ_{ij} is nonzero.

2. For each node i at the coarsest scale, define a quadtree $\mathcal{T}_i$ which consists of i and the descendants of i (Figure 4-6(b) shows four such trees). Let $\mathcal{T}_S = \{\mathcal{T}_i\}$ be the set of disjoint quadtrees sitting on the region of interest, i.e. $\mathcal{T}_i \in \mathcal{T}_S$ only when there exist at least one node $j \in \mathcal{T}_i$ such that $j \in S$.

3. Tree inference: Perform exact inference on the subgraphs in $\mathcal{T}_S$.

4. Adaptive block Gauss-Seidel iteration:

   (a) Set $V_n = \varnothing$. For $i \in S$, compute the node weights using (4.22). For all other nodes $i \in V \backslash S$, set the node weights to zero.

   (b) Find the maximum weight node i^* from $V \backslash V_n$, and set $V_n \leftarrow V_n \cup i^*$.

   (c) If $|V_n| = k$, stop. Otherwise, update the weight of each node $j \in \mathcal{N}(i) \cap V \backslash V_n$. If $j \in S$, update the node weight using (4.23). If $j \in V \backslash S$, compute the node weight of j first using (4.22) and update it using (4.23). Go to step 2(b).

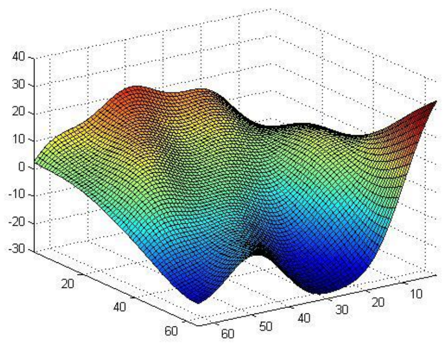
   (d) Apply the Gauss-Seidel iteration in (2.18) and update the nodes in V_n.

5. Repeat Step 3 and 4 until a stopping criterion is met.

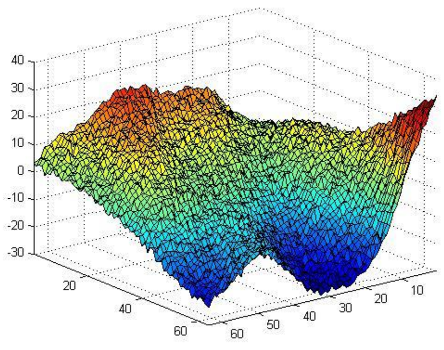
Table 4.5: Re-estimation algorithm to efficiently update the estimates to incorporate local changes in a pyramidal graphical model.

Although the adaptive GS algorithm tends to select the nodes in the region with nonzero entries in ΔJ or Δh, the algorithm is greedy in nature and thus tends to pick only the finest scale nodes. As with the adaptive ET algorithm in Section 4.1, including the coarser scale nodes may not reduce the error greatly at the immediate next iteration, but eventually leads to faster convergence by propagating the changes rapidly. In addition, we have seen in Section 1.2 that the changes in the far-apart nodes can be well approximated at coarser scales. Therefore, we alternate between tree-based inference iterations on the quadtrees sitting on the region of interest and adaptive block GS iterations to get a faster convergence.

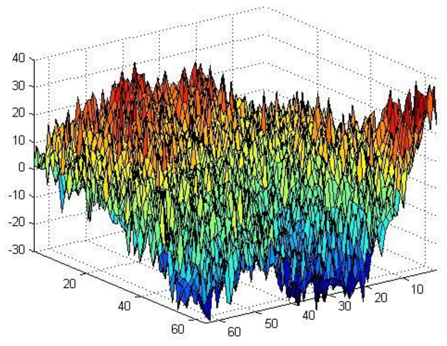
Table 4.5 describes the re-estimation algorithm for our pyramidal graph. Since ΔJ and Δh have nonzero elements only in a localized region, it is likely that the nodes in the region have larger node weights than the nodes outside of the region. Therefore,



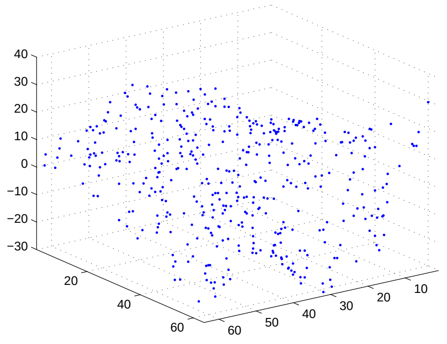
(a)



(b)



(c)



(d)

Figure 4-9: Test surface and measurements. (a) True surface. (b) Dense measurements with low-level noise ($\sigma^2 = 1$). (c) Dense measurements with high-level noise ($\sigma^2 = 25$). (d) Sparse measurements (10% of the finest scale nodes) with low-level noise ($\sigma^2 = 1$).

when applying the adaptive block GS iteration, instead of searching through all nodes to find the node with the maximum weights, we first consider the nodes inside the region as candidate nodes to be updated. When a node at the boundary of the region is selected, we include all the neighboring nodes in our candidate set. Note that using this modified version of the adaptive block Gauss-Seidel algorithm at step 4 in Table 4.5, we do not need to perform any global operation.

4.5 Simulation Results

In this section, we present simulation results using the inference algorithms presented in the previous sections. For estimation of conditional means and error variances, we use a 64×64 synthetic surface shown in Figure 4-9(a) as the "true" surface, and generate three sets of measurements in Figure 4-9(b)-(d):

- Dense measurements with low-level noise
- Dense measurements with high-level noise
- Sparse measurements with low-level noise

For dense measurements, each node at the finest scale generates one noisy measurement: so $y_i = x_i + v_i$ for $i \in V_M$, where v is a white Gaussian noise process with covariance matrix $R = \sigma^2 I$. Note that even in this case, the coarser scales nodes of the pyramidal graph do not have any measurements. We show inference results for both low ($\sigma^2 = 1$) and high ($\sigma^2 = 25$) level noise[1]. For sparse measurements, we randomly select 10% of the finest scale nodes to generate measurements with noise variance $\sigma^2 = 1$.

4.5.1 Estimation of Conditional Means

We test the multipole-motivated algorithms described in Table 4.1 and 4.2 on the pyramidal graph with four scales and compare their performances with the corresponding coarse-to-fine multigrid method and inference on a monoscale thin-membrane model. Let x_i be the height of the true surface at $i \in V_M$, then the RMS error of

[1]Recall that $J = J_{prior} + C^T R^{-1} C = J_{prior} + \frac{1}{\sigma^2} C^T C$. For dense measurements, $C^T C$ is a diagonal matrix with identity matrix at the finest scale and zero entries for all coarser scales. For the high-level noise with $\sigma^2 = 25$ and for the prior model with the parameter we used (see Table 4.6), each entry in the measurement matrix $\frac{1}{\sigma^2} C^T C$ has a value less than 5% of the corresponding entry in the J_{prior} matrix. For the low-level noise with $\sigma^2 = 1$, the measurement term is about 15% of the corresponding entry in the prior model, so we are putting more confidence in the measurement term than the high-level noise case.

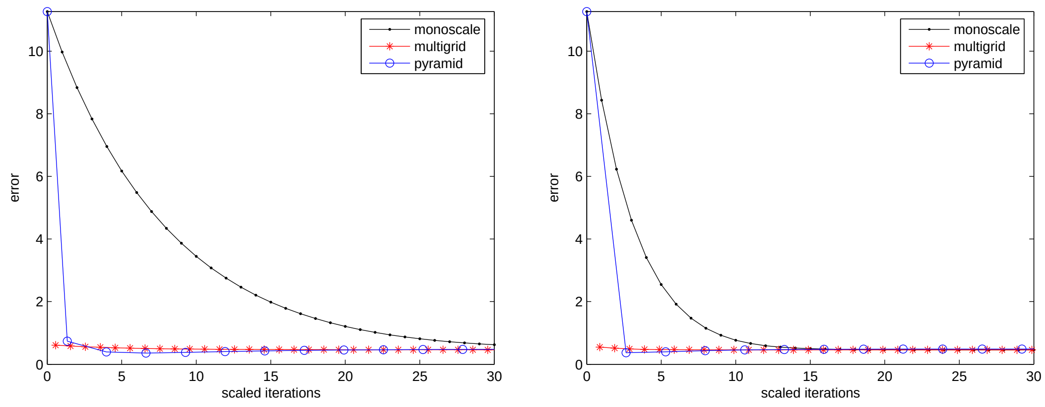
estimates at iteration n can be computed as follows:

$$e_{rms}^{(n)} = \sqrt{\frac{\sum_{i \in V_M} (x_i - \hat{x}_i^{(n)})^2}{N_M}} \tag{4.24}$$

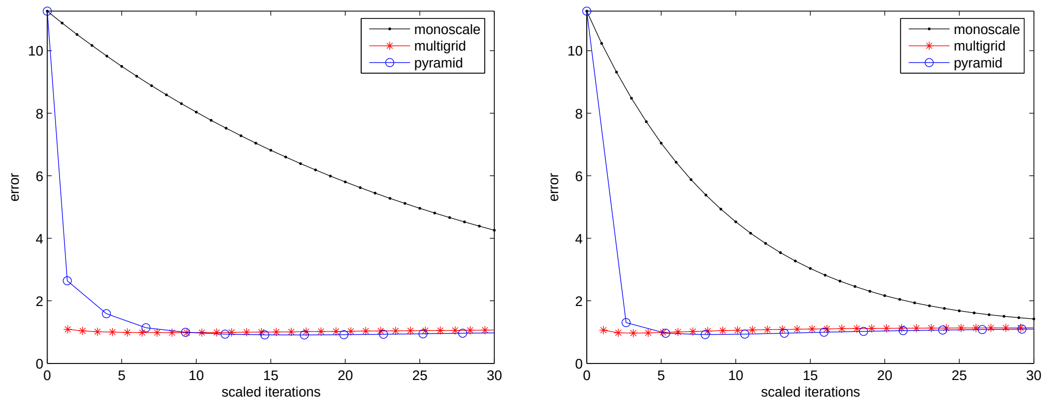
where $\hat{x}^{(n)}$ denotes the vector of estimates at iteration n and N_M is the number of nodes at the finest scale. Figure 4-10 shows the convergence of the RMS error for inference algorithms on the three models. For the plots in the left column, we applied the algorithm in Table 4.1 for the pyramidal graph, and Gauss-Jacobi (GJ) iterations for the monoscale thin-membrane model. To compare the performance with the coarse-to-fine multigrid methods, we constructed multiple grid models with four scales, and used GJ iterations within each grid model. For the plots in the right column, we used the algorithm described in Table 4.2 for the pyramidal graph, and also applied the adaptive ET iterations for each scale in the multigrid methods and for the monoscale model.

In order to account for the fact that the number of variables updated is different for each iteration in the pyramidal graph, in the multiple grids, and in the monoscale grid model, we convert the number of iterations to equivalent GJ iterations in the monoscale model. For the pyramidal graph, each iteration involves one downward sweep in which we perform a single GJ iteration within each scale, and a tree-inference step using the quadtree embedded in the pyramidal graph. So the number of operations is twice the number of nodes in the pyramidal graph, and we count one coarse-to-fine and fine-to-coarse sweep as being equivalent to $2 \times (1 + 1/4 + 1/16 + 1/64)$ iterations in a monoscale model. For the multigrid method, each iteration at scale m is re-scaled by $1/2^{M-m}$ to account for the fact that iterations at coarser scales involve smaller numbers of nodes. Both the inference algorithms in the pyramidal graph and the multigrid method require far fewer equivalent iterations than algorithms in the monoscale counterpart.

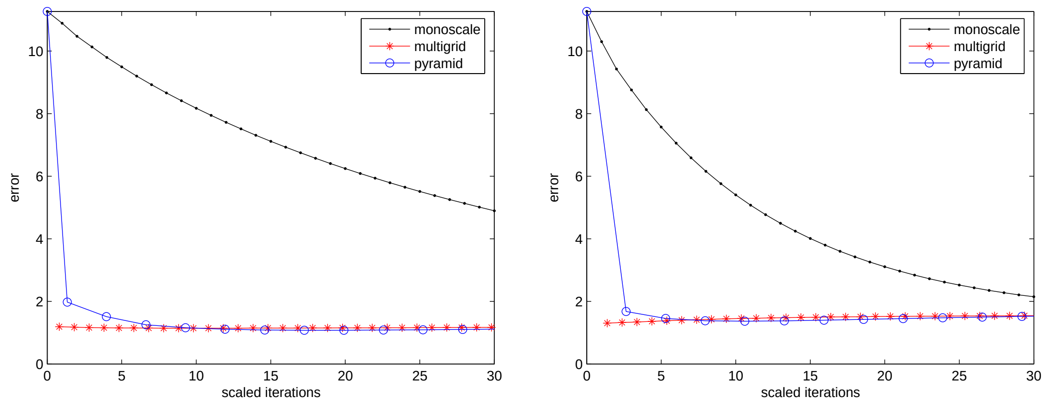
Note that the iterations in Figure 4-10 do not necessarily converge to the point where the RMS error is minimized. This is because the true surface is not exactly equal to the solution of the problem $\hat{x} = J^{-1}h$. In addition, for real problems, the



Dense measurements with low-level noise

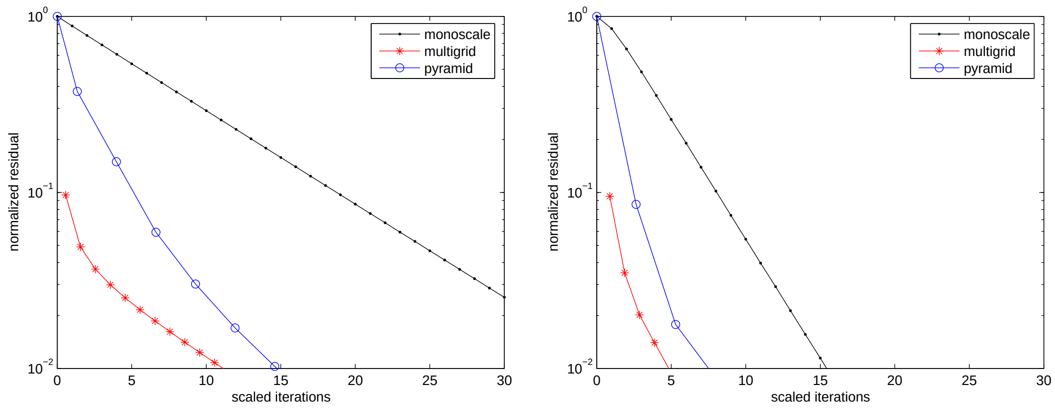


Dense measurements with high-level noise

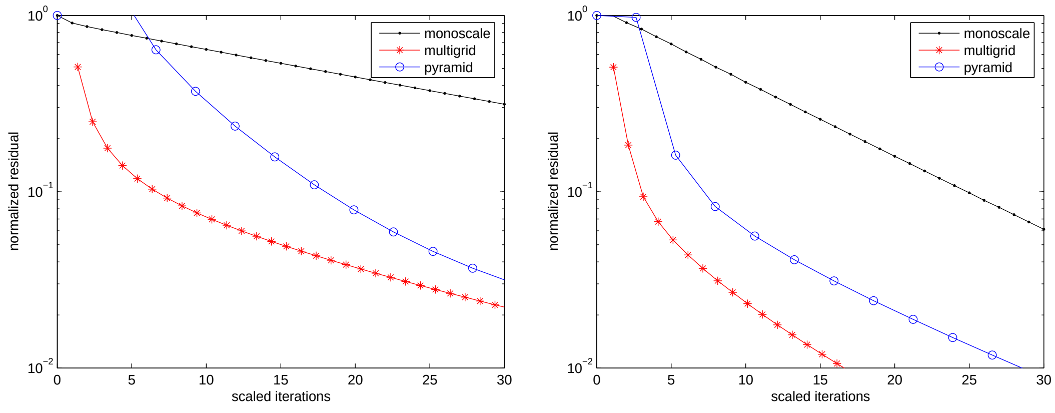


Sparse measurements with low-level noise

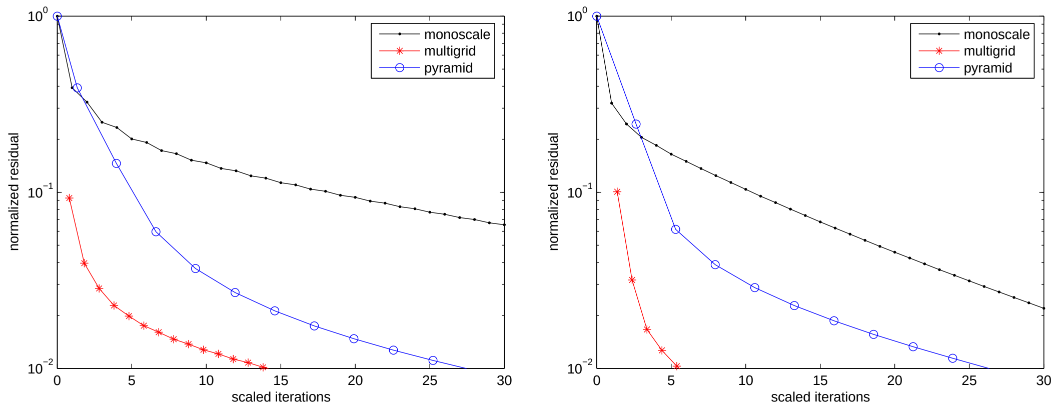
Figure 4-10: RMS errors in surface estimation using multipole-motivated algorithms on the pyramidal graph and corresponding multigrid methods and iterations on the monoscale model. Left: Gauss-Jacobi iterations. Right: Adaptive ET iterations.



Dense measurements with low-level noise

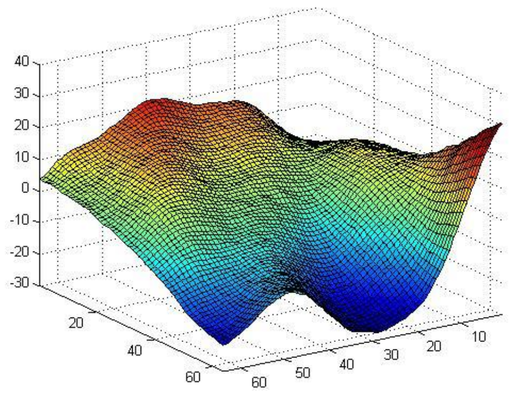


Dense measurements with high-level noise

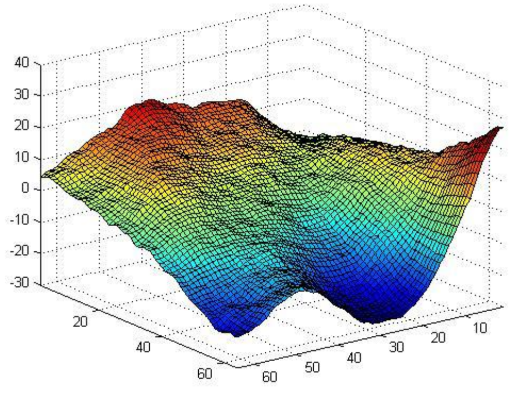


Sparse measurements with low-level noise

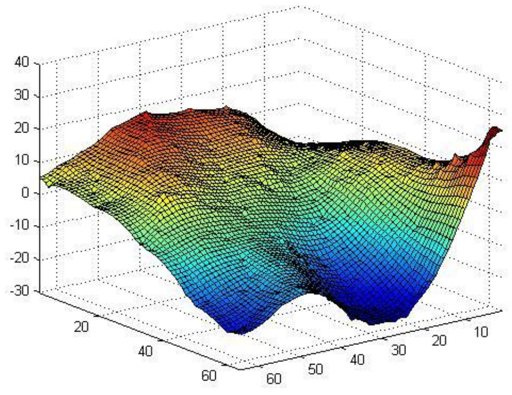
Figure 4-11: Convergence rates for the pyramidal graph, multiple grids, and the monoscale grid model. Left: Gauss-Jacobi iterations. Right: Adaptive ET iterations.



Dense measurements with low-level noise



Dense measurements with high-level noise



Sparse measurements with low-level noise

Figure 4-12: Estimates using adaptive ET iterations on the pyramidal graph when the normalized residual is reduced to 0.01.

	Pyramid	Monoscale grid
Dense measurements with low-level noise	1.5	2
Dense measurements with high-level noise	0.2	0.3
Sparse measurements with low-level noise	0.4	0.5

Table 4.6: The values of the parameter φ used for the prior models.

RMS error is not computable because we do not know the true value of x. Therefore, in practice, we use the residual $h^{(n)} = h - J\hat{x}^{(n)}$ to test the convergence of an algorithm. Figure 4-11 shows the the normalized residual error $\frac{\|h^{(n)}\|_{\ell_2}}{\|h\|_{\ell_2}}$ versus the the number of re-scaled iterations. Note that the h and J for the pyramidal graph are different from those in the multiple or monoscale grid models. The pyramidal graph has a more complicated graph structure than separate grid models used in the multigrid method, but still the multipole-motivated algorithms demonstrate comparable convergence rates to the multigrid methods. Figure 4-12 shows the estimation results using the pyramidal graph for each measurement set when the normalized residual is reduced to 1%.

The convergence rates of the normalized residuals critically depend on the values of parameters of the prior model (see Section 3.1). We set the parameters for the pyramidal graph following the ratios of physical distances between nodes as in (3.10). Since a pyramidal graph has correlations with slower decays compared to the monoscale counterpart with the same parameter at the finest scale, we adjuste the parameters of different models to yield approximately similar estimates for each set of measurements. The parameters used for the pyramidal graph and the monoscale model are listed in Table 4.6. The grid models used in the multigrid methods have the same parameter values as the monoscale model at the finest scale and the parameters are decreased by 4 as we go to coarser scales.

Next, we apply the Lagrangian relaxation method to the pyramidal graph with subgraphs in Figure 4-6. Based on the prior model of the pyramidal graph, we set

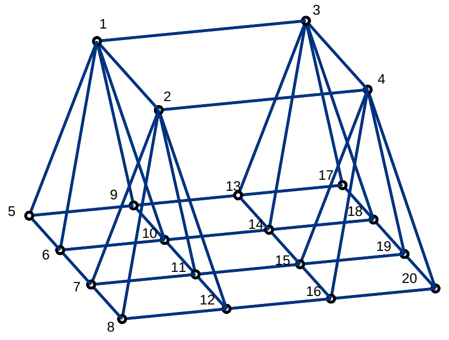


Figure 4-13: Ordering of nodes in the pyramidal graph.

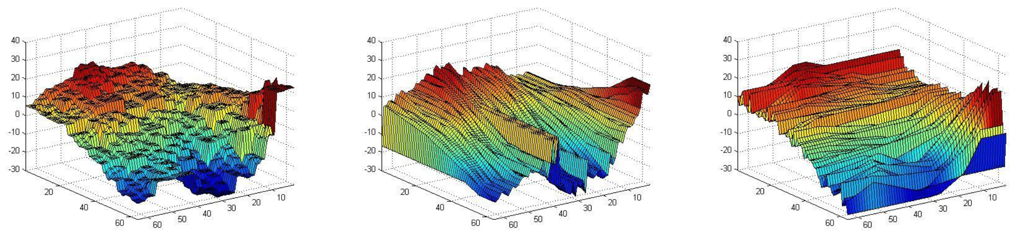
the initial decomposition as follows:

$$\begin{aligned} J^1 &= J_t + (1-2\epsilon)\ C^T R^{-1} C & h^1 &= (1-2\epsilon)\ C^T R^{-1} y \\ J^2 &= J_{sv} + \epsilon\ C^T R^{-1} C & h^2 &= \epsilon\ C^T R^{-1} y \\ J^3 &= J_{sh} + \epsilon\ C^T R^{-1} C & h^3 &= \epsilon\ C^T R^{-1} y \end{aligned} \tag{4.25}$$

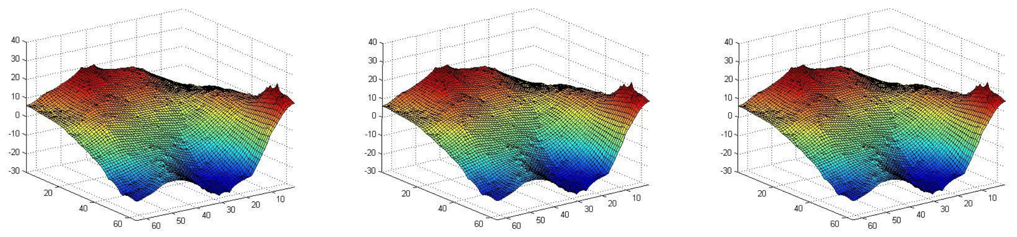
where J_{sv} corresponds to the matrix with smoothness constraints in the vertical chains in Figure 4-6(c) and J_{sh} corresponds to the horizontal chains in Figure 4-6(d) and $J_{sv} + J_{sh} = J_s$. The constant ϵ assigns weights of measurements to each subgraph and we set $\epsilon = 1/3$. It is straightforward to check that for all ϵ such that $0 < \epsilon < 1$, the following conditions are satisfied: $J^k \succ 0$ for all k, $J = \sum_k J^k$ and $h = \sum_k h^k$.

We number the nodes in the pyramidal graph sequentially from the coarsest scale nodes to the finest scale nodes. In each grid model, we use lexicographic ordering by columns, i.e. starting from the node at the upper left corner, we increase the number in the direction of vertical chains. Figure 4-13 shows the ordering of nodes for the pyramidal graph in Figure 4-6(a). We adopt the coarse-to-fine and fine-to-coarse philosophy here and set the ordering of node updates $\{u^{(n)}\}$ in Table 4.3 alternating between coarse-to-fine $\{1, 2, \cdots, N\}$ and fine-to-coarse $\{N, N-1, \cdots, 1\}$ orders.

The LR method is applied to reconstruct the surface from the sparse measurements in Figure 4-9(d). Figure 4-14 shows the estimates on each subgraph after the initialization step in Table 4.3 and after 30 iterations of the LR algorithm. Initially,



Estimates after the initialization step



Estimates after convergence

Figure 4-14: Estimates using the Lagrangian relaxation method for sparse measurements. Left: subgraph 1 (quadtrees). Middle: subgraph 2 (vertical chains). Right: subgraph 3 (horizontal chains)

the estimates on each subgraph show severe discontinuities and blockiness, but after convergence, estimates are smoothed out by node potential exchange procedures. At each iteration n, let $\hat{x}^{(n)}$ be the average of estimates in each subgraph, and Figure 4-15 shows the RMS error at each iteration.

Both the multipole-motivated algorithms and the LR method have linear complexity per iteration and converge in a few iterations compared to the number of nodes in the graph. However, the LR method updates each node sequentially while the multipole-motivated algorithms update all nodes in one scale in a parallel way. Since the operations per iteration are of different characteristics, it is not straightforward to compare the performance of these algorithms in general.

4.5.2 Estimation of Variances

In the previous sections, we introduced three methods to estimate variances of the nodes in the pyramidal graph. First, using the Lagrangian relaxation method, we obtain upper bounds of error variances. Secondly, the coarse-to-fine low-rank approx-

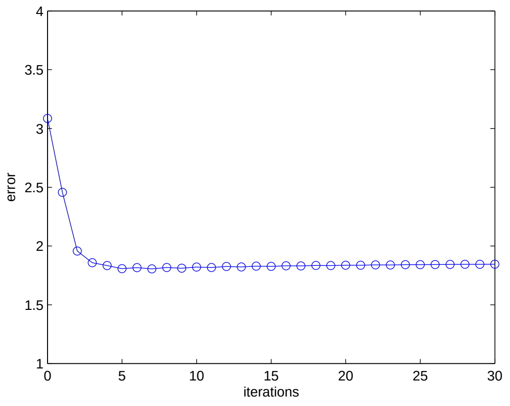


Figure 4-15: RMS error in surface estimation for sparse measurements using the Lagrangian relaxation methods on the pyramidal graph.

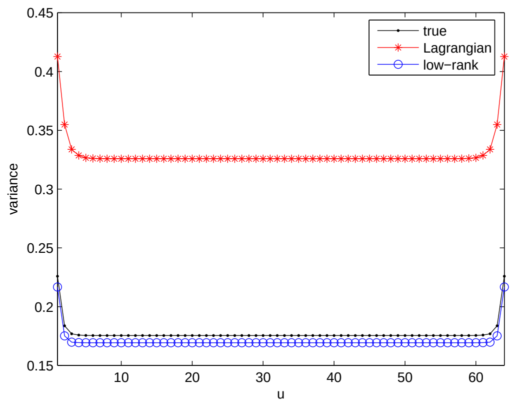
imation method described in Table 4.3 computes approximate lower bounds. Lastly, using the wavelet-based low-rank approach, the variances at the finest scale nodes can be estimated.

Figure 4-16 shows one cross section of the bounds on variances of the 64×64 synthetic surface given the three sets of measurements in Figure 4-9. The upper bounds show estimates computed by the LR method after 30 iterations, and lower bounds are computed by applying 5 coarse-to-fine sweeps of the low-rank approximation method. The upper bounds obtained by the LR method are rather loose, but they follow the shape of the true variances. In addition, note that these bounds are obtained while computing the optimal estimates without any additional cost. For the coarse-to-fine low-rank approximation, the simple spliced standard bases with 2^{m-1} columns are used to estimate the conditional correlations at scale m.

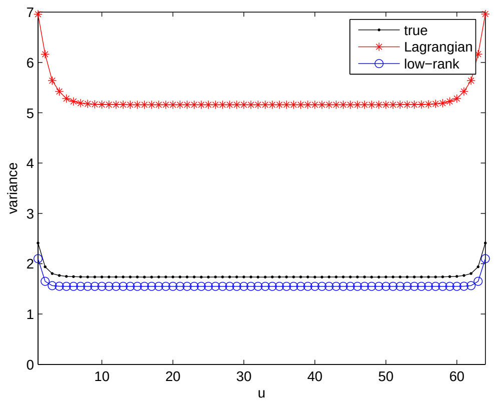
Figure 4-17 shows the variances estimated by the wavelet-based low-rank approximation methods. We used 160 columns to compute the finest scale variances, and it can be observed that the estimates are very close to the true variances.

4.5.3 Re-estimation

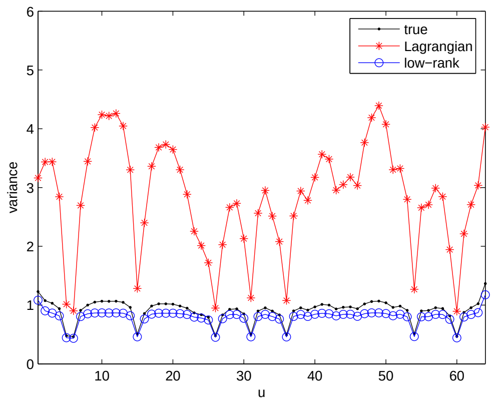
We revisit the example introduced in Section 1.2 in which sharp discontinuities are blurred in the estimation process because of the smoothness prior, and apply the



Dense measurements with low-level noise

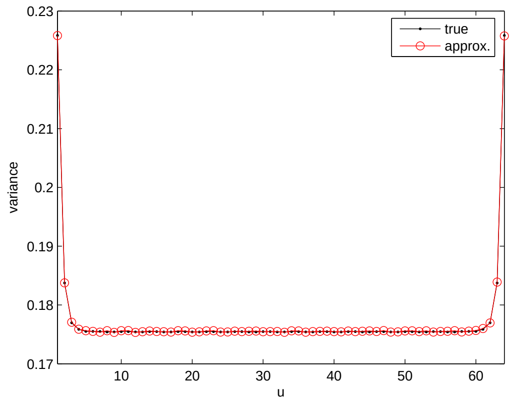


Dense measurements with high-level noise

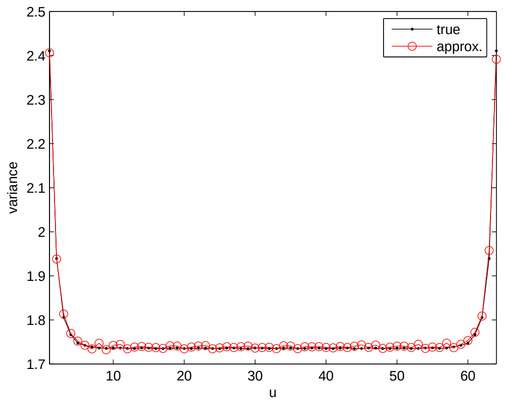


Sparse measurements with low-level noise

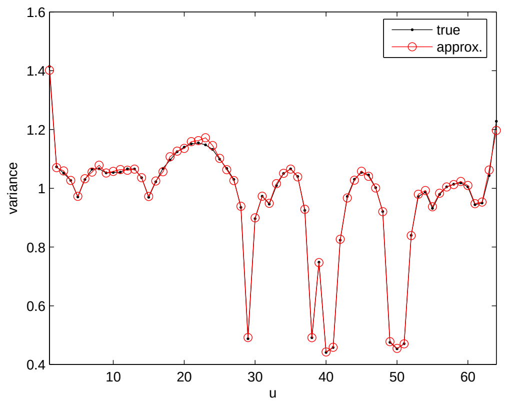
Figure 4-16: A cross section of estimates of approximate variances using the Lagrangian relaxation (LR) methods and the coarse-to-fine low-rank algorithm on the pyramidal graph.



Dense measurements with low-level noise

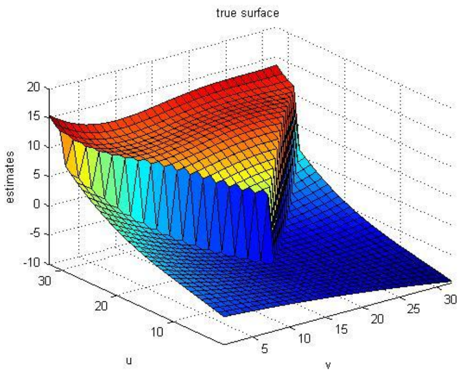


Dense measurements with high-level noise

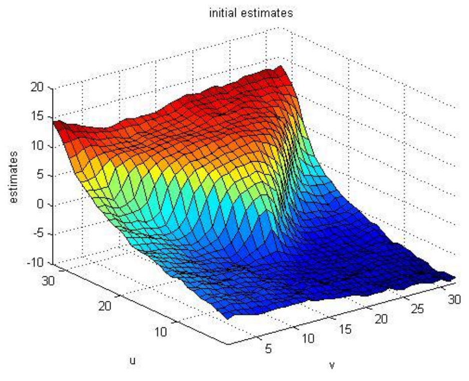


Sparse measurements with low-level noise

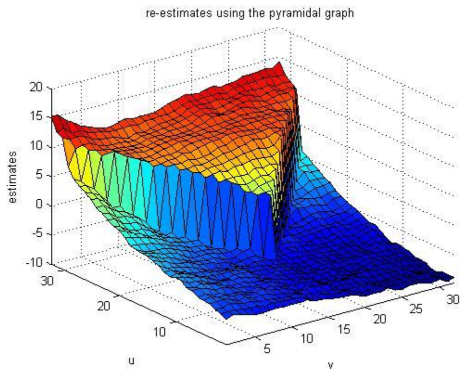
Figure 4-17: A cross section of estimates of variances using the wavelet-based low-rank approach on the pyramidal graph.



(a)



(b)



(c)

Figure 4-18: Re-estimation applied to a surface with discontinuities. (a) True surface. (b) Estimates using a smoothness prior on the pyramidal graph. (c) Re-estimates after 10 iterations. Each iteration involves a little more than the half of all nodes.

re-estimation algorithm to correct the estimates. Figure 4-18(a) and 4-18(b) show the true surface and blurred estimates, respectively. We pick the location of discontinuities manually and modify the prior of the pyramidal graph such that the two nodes on either side of the discontinuities are not connected with an edge. The pyramidal graph with four scales modeling this surface has 1360 nodes in total. Instead of updating all the nodes, we use the set of quadtrees sitting along the discontinuities, which involves the half of all nodes. Then we apply the tree inference steps alternating with the adaptive block Gauss-Seidel iteration which selects and updates 10 nodes at each iteration. Figure 4-18(c) shows the results after 10 iterations, with much more accurate estimates around the discontinuities.

Next, consider applying the re-estimation algorithm to the case when a set of

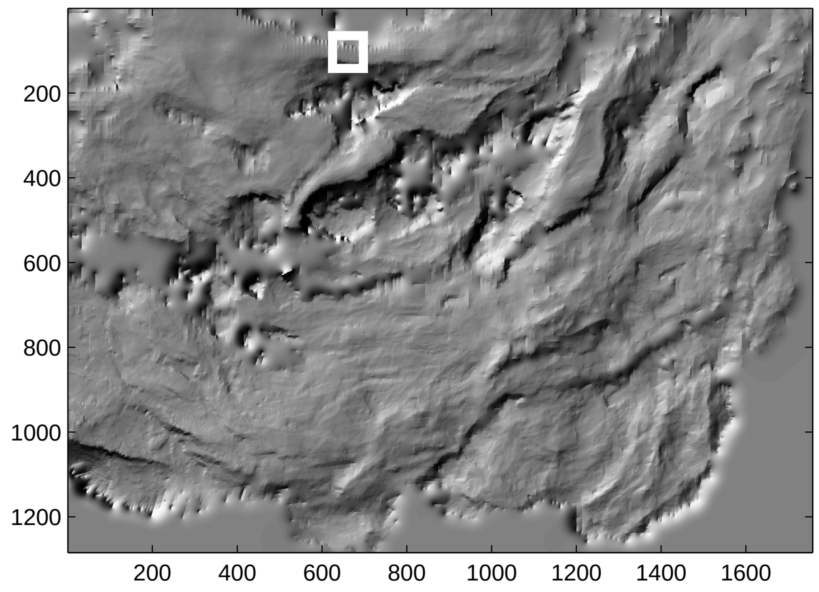
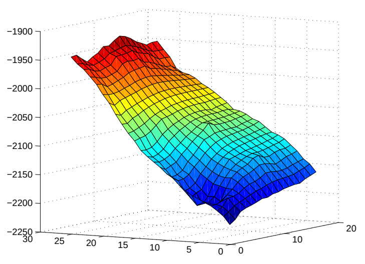


Figure 4-19: The estimates of the top surface of a salt deposit.

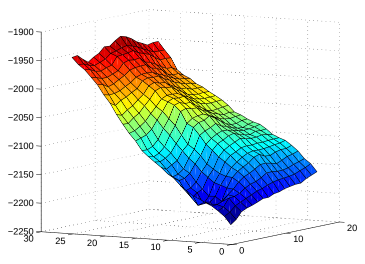
measurements is added to a local region. We use the real problem of estimating the top surface of a large salt deposit located below the sea floor of Gulf of Mexico. The measurements, provided by Shell International Exploration, Inc., consist of $377,384$ picks by analysts interpreted from seismic data. We set the resolution at the finest scale as 60 feet and estimate the surface at 1757×1284 nodes using a pyramidal graph with four scales. The total number of nodes in the pyramidal graph is about 3 million. Figure 4-19 shows a 2D illustration of the estimates of the salt-top surface.

We introduce 100 new measurements in the small 17×17 region indicated as the white square in Figure 4-19. The re-estimation algorithm uses a tree-inference step involving 765 nodes alternating with an adaptive block Gauss-Seidel iteration which updates 100 nodes. Figure 4-20(b) shows the updated estimates after 10 iterations. The estimates show more detailed surface delineations in the region compared to the estimates before adding the measurements shown in Figure 4-20(a).

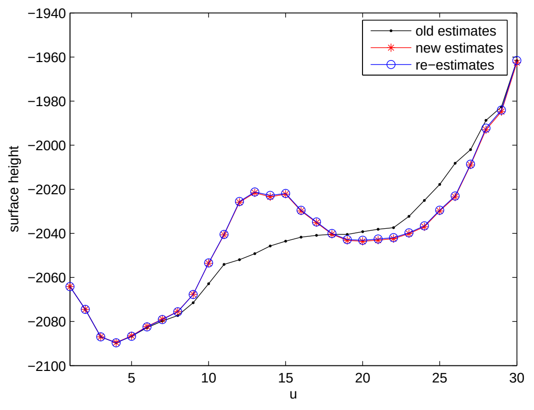
Figure 4-20(c) shows one cross section of the re-estimates together with the estimates before adding the measurements. To compare the performance, the figure also



(a)



(b)



(c)

Figure 4-20: Reestimation applied to the problem of updating estimates to incorporate a new set of measurements in a local region. (a) Estimates before adding measurements. (b) Re-estimates. (c) A cross section of re-estimates.

shows the updated estimates using a naive method: after modifying J and h to incorporate the new measurements, we simply perform inference on the entire pyramid. Using this naive implementation, 3 million nodes are updated at each iteration. The re-estimation algorithm updates less than 1000 nodes at each iteration, yet after 10 iterations, they converge to the same results.

4.5.4 Observations

From the preceding simulations, we first showed that multiscale approaches, using pyramidal graphical model or multigrid methods, require far fewer iterations than the counterpart monoscale approach. In particular, multipole-motivated inference algorithms on the pyramidal graph significantly outperform the inference algorithms on the monoscale model when the measurements are sparse or corrupted by high-level noise.

The pyramidal graph is a consistent graphical model, while in the multigrid methods, it is assumed that stochastic structure in different scales are separated from each other. The benefit of using the pyramidal graph is that various efficient algorithms developed for inference on GMRF can be easily applied. For example, using the Lagrangian relaxation method, we demonstrated that the inference on the pyramidal graph can be performed by iteratively solving inference problems on tractable subgraphs. In addition, the lower bound on variances can be computed by applying the low-rank approximation in each scale iteratively through coarse-to-fine sweeps. Using the wavelet-based low-rank algorithms, we also obtained close approximations of variances of nodes at the finest scale.

To solve the re-estimation problem efficiently, we first need to propagate the local changes rapidly and also adaptively select a subset of nodes to be updated. Combining the tree-inference step with the adaptive block Gauss-Seidel iteration on the pyramidal graph, the re-estimation algorithm rapidly computes more accurate estimates around the discontinuities or updates the estimates to incorporate a new set of measurements in a local area.

Chapter 5

Multiscale Parameter Estimation

The pyramidal graphical model defined in Chapter 3 has a set of parameters which control the strength of the smoothness constraints. In this chapter, we discuss estimating the parameters defining the prior model as well as the noise variance. We begin Section 5.1 by reviewing the parameters in the pyramidal graph and introducing necessary notation. When the ratios between the parameters are fixed and there remains a single free parameter for the prior model, the parameter estimation problem can be solved by the standard Expectation-Maximization (EM) algorithm. In Section 5.2, we describe the EM algorithm for the pyramidal graph and present simulation results. When the number of free parameters is increased, however, the problem becomes much more challenging because of the log partition function, and using a surrogate log partition function as described in Section 5.3 fails to provide satisfying results. We discuss the difficulties and possible directions for approximate parameter estimation in Section 5.4.

5.1 Parameters in the Pyramidal Graph

The prior model $J_{prior} = J_s + J_t$ defined on the pyramidal graph with M scales has M in-scale parameters α_m's associated with J_s and $M - 1$ inter-scale parameters β_m's associated with J_t (see (3.2), (3.3), and (3.4)). The pyramidal graph can be considered as a set of candidate models, and estimating the parameters corresponds to searching for the best candidate given measurements [33].

Throughout Chapter 3 and Chapter 4, we fixed the ratios of these parameters based on the physical distance between the corresponding pair of nodes as in (3.10), and allowed a single free parameter φ to control the overall strength of the constraints. In addition to this prior parameter, we define $\gamma \triangleq 1/\sigma^2$ as a parameter which specifies the noise variance. Then, the J matrix can be represented in terms of the two free parameters φ and γ as follows:

$$J = \varphi J_P + \gamma C^T C \tag{5.1}$$

where C is the measurement mapping matrix defined in Section 2.1.3 and J_P is the J_{prior} matrix with the fixed parameter $\varphi = 1$. We use the vector θ to denote all free parameters, so in this case, $\theta = (\varphi, \gamma)$.

Now, consider estimating α and β separately. We still force the coarser scale parameters to be decreased by 4, but allow the ratio between α_M and β_{M-1} to vary. Then, we have two free parameters for the prior model and the J matrix can be represented as follows:

$$J = \alpha J_S + \beta J_T + \gamma C^T C \tag{5.2}$$

where J_S is the J_s matrix in (3.4) with the fixed parameters $\alpha_m = \frac{1}{4^{M-m}}$ and J_T is the J_t matrix in (3.2) with $\beta_m = \frac{1}{4^{M-1-m}}$. The collection of free parameters is $\theta = (\alpha, \beta, \gamma)$.

Since J_s is a block diagonal matrix, it is straightforward to extend the above case to consider α_m's separately and to estimate $\theta = (\alpha_1, \ldots, \alpha_M, \beta, \gamma)$. Then,

$$\begin{aligned} J &= J_{prior} + \gamma C^T C \\ &= J_s + \beta J_T + \gamma C^T C \\ &= \begin{pmatrix} \alpha_1 J_{s1} & 0 & 0 & 0 \\ 0 & \alpha_2 J_{s2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \alpha_M J_{sM} \end{pmatrix} + \beta J_T + \gamma C^T C \end{aligned} \tag{5.3}$$

where J_{sm} is a thin-membrane model within each scale with a unit parameter in (2.9).

5.2 EM Algorithm

The Expectation Maximization (EM) algorithm [1, 12] is an iterative method to estimate parameters given incomplete (partially-observed) data. Let us denote the joint probability of x and y as an exponential family (see Section 2.1.2):

$$p(x, y|\theta) = \exp(\sum_{a \in \mathcal{A}} \theta_a \phi_a(x, y) - \Phi(\theta)) \tag{5.4}$$

Given observation y, we seek the parameter θ which maximizes the log-likelihood function:

$$l(\theta) \equiv \log p(y|\theta) = \log \int p(x, y|\theta) dx. \tag{5.5}$$

This log-likelihood is difficult to maximize or even evaluate due to the integral form. Instead, consider maximizing a concave lower bound obtained by using Jensen's inequality [46],

$$\begin{aligned} l(\theta) &= \log \int p(x, y|\theta) dx \\ &\geq \int q(x) \log \frac{p(x, y|\theta)}{q(x)} dx \triangleq \mathcal{L}(q, \theta). \end{aligned} \tag{5.6}$$

The EM algorithm can be interpreted as maximizing the lower bound using a coordinate ascent iteration [49]. Each iteration of the EM algorithm consists of two steps:

$$\begin{aligned} &\text{E-step:} \quad & q^{(i)} &= \arg\max\nolimits_q \mathcal{L}(q, \theta^{(i-1)}) \\ &\text{M-step:} \quad & \theta^{(i)} &= \arg\max\nolimits_\theta \mathcal{L}(q^{(i)}, \theta) \end{aligned}$$

In the expectation or E-step, we fix the parameters and optimize over the distribution q. In the maximization or M-step, the set of parameters which maximizes the lower bound defined by the distribution q is selected. The algorithm converges to the local maximum of $\mathcal{L}(q, \theta)$ [12].

Given the current parameter estimates $\theta^{(i-1)}$, it can be shown that the distribution $q(x)$ which maximizes the lower bound is the conditional distribution $p(x|y, \theta^{(i-1)})$ [49]. When $p(x, y|\theta)$ is an exponential family as in (5.4), it is sufficient to compute the expected values of the sufficient statistics $E[\phi_a(x, y)|y, \theta^{(i-1)}]$ for the subsequent M-step. Therefore, the E-step is equivalent to performing inference given measurements and current parameter estimates. When exact inference is intractable, it is common to use approximate inference algorithms such as BP or embedded subgraph algorithms, although the convergence of the EM is no longer guaranteed [49].

The M-step finds the next parameter estimates which maximize $\mathcal{L}(q^{(i)}, \theta)$. Substituting $q^{(i)}(x) = p(x|y, \theta^{(i-1)})$ in (5.6),

$$\mathcal{L}(q^{(i)}, \theta) = \int p(x|y, \theta^{(i-1)}) \log p(x, y|\theta) dx + H(q^{(i)}) \tag{5.7}$$

where the second term is the entropy of $q^{(i)}(x)$ and is irrelevant to θ. So, we define $Q(\theta, \theta^{(i-1)})$ as the first term of the above equation. Then, using the exponential family representation in (5.4):

$$\begin{aligned} Q(\theta, \theta^{(i-1)}) &= E[\log p(x, y|\theta) \mid y, \theta^{(i-1)}] \\ &= \sum_{a \in \mathcal{A}} \theta_a E[\phi_a(x, y) \mid y, \theta^{(i-1)}] - \Phi(\theta) \end{aligned} \tag{5.8}$$

Note that the expected values in the above equation are computed using the parameter values at the preceding E-step. Since the first term is linear in θ and the log partition function is a convex function of θ (see Section 2.1.2), the Q function is concave with respect to θ. Therefore, to find θ_a's which maximize the Q function, we set the gradient of Q as zero and solve the following equations:

$$\frac{\partial \Phi(\theta)}{\partial \theta_a} = E[\phi_a(x, y) \mid y, \theta^{(i-1)}] \qquad \forall a \in \mathcal{A} \tag{5.9}$$

Estimation of $\theta = (\varphi, \gamma)$ Consider applying the EM algorithm to our pyramidal graph with J defined in (5.1). Using $p(x, y|\theta) = p(x|\theta)p(y|x, \theta)$, the Q function can be separated into the terms involving either φ or γ. In particular, the log partition function can be decomposed as $\Phi(\theta) = \Phi_1(\theta_1) + \Phi_2(\theta_2)$ and we have

$$\begin{array}{llllll} \theta_1 & = & \varphi & \theta_2 & = & \gamma \\ \phi_1(x) & = & -\frac{1}{2}x^T J_P x & \phi_2(x,y) & = & -\frac{1}{2} \parallel y - Cx \parallel^2 \\ \Phi_1(\theta_1) & = & -\frac{1}{2}\log\det(\varphi J_P) & \Phi_2(\theta_2) & = & -\frac{1}{2}\log\det(\gamma C^T C) \end{array} \tag{5.10}$$

where we have assumed (for simplicity) that h_{prior} is zero and ignored the constant $\left(\frac{N}{2}\log(2\pi)\right)$ in the log partition function.

In the E-step, the expected values of ϕ_a's are evaluated using the conditional means $\hat{x}^{(i-1)} = E[x \mid y, \theta^{(i-1)}]$ and error variances $\hat{P}^{(i-1)} = E[(x-\hat{x}^{(i-1)})(x-\hat{x}^{(i-1)})^T \mid y, \theta^{(i-1)}]$. First, using the fact that the trace and the expectation operators commute,

$$\begin{aligned} \eta_1 \triangleq E[x^T J_P x \mid y, \theta^{(i-1)}] &= E[tr(J_P x x^T) \mid y, \theta^{(i-1)}] \\ &= tr(J_P E[x x^T \mid y, \theta^{(i-1)}]) \\ &= tr\left(J_P(\hat{P}^{(i-1)} + \hat{x}^{(i-1)}(\hat{x}^{(i-1)})^T)\right) \\ &= tr(J_P \hat{P}^{(i-1)}) + (\hat{x}^{(i-1)})^T J_P \hat{x}^{(i-1)} \end{aligned} \tag{5.11}$$

Due to the sparsity of J_P, we only need the variances of individual nodes and covariances between the pairs of neighboring nodes to compute both terms in the above equation [23]. Similarly,

$$\begin{aligned} \eta_2 \triangleq E[\parallel y - Cx \parallel^2 \mid y, \theta^{(i-1)}] &= E[tr((y-Cx)(y-Cx)^T \mid y, \theta^{(i-1)}] \\ &= \parallel y - C\hat{x}^{(i-1)} \parallel^2 + tr(C\hat{P}^{(i-1)}C^T) \end{aligned} \tag{5.12}$$

where $tr(C\hat{P}^{(i-1)}C^T)$ can be computed from the error variances of individual nodes. When exact inference is intractable, we approximately evaluate η_1 and η_2 using the approximation to $\hat{x}^{(i-1)}$ and $\hat{P}^{(i-1)}$ computed from the inference algorithms in Chapter 4.

In the M-step, we solve (5.9) to find $\varphi^{(i)}$ and $\gamma^{(i)}$ which maximize the Q function. Note that the log partition functions in (5.10) can be represented in terms of the parameters as follows:

$$\Phi_1(\theta_1) = -\frac{1}{2}(\log\det J_P - N\log\varphi) \tag{5.13}$$

$$\Phi_2(\theta_2) = -\frac{1}{2}(\log\det C^T C - N_{meas}\log\gamma) \tag{5.14}$$

where N and N_{meas} are the number of nodes and measurements, respectively. Substituting (5.13) and (5.14) into (5.9), we obtain the following expressions for the next parameter estimates:

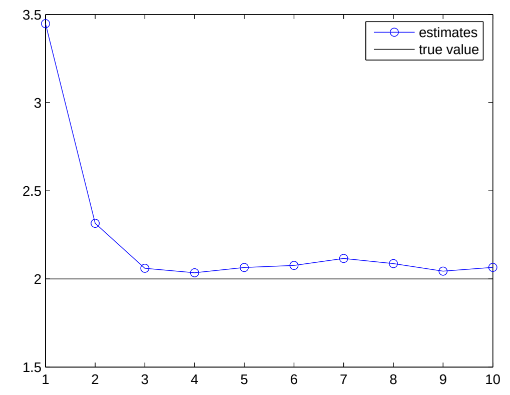
$$\varphi^{(i)} = \frac{N}{\eta_1} \tag{5.15}$$

$$\gamma^{(i)} = \frac{N_{meas}}{\eta_2} \tag{5.16}$$

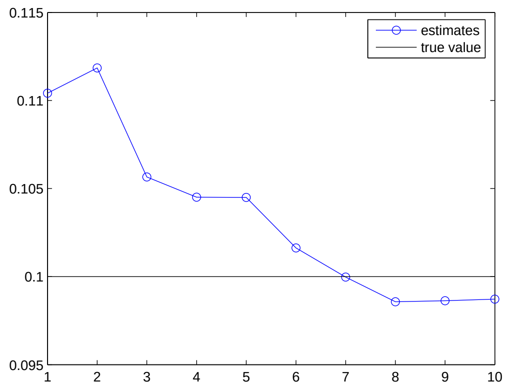
where η_1 and η_2 are the expected values in (5.11) and (5.12), respectively.

Simulation Results In order to test the accuracy of the EM algorithm on the pyramidal graph, we first generate measurements from the pyramidal graph with known parameters and estimate parameters from these measurements. Consider a pyramidal graph with four scales with 16×16 nodes at the finest scale. Figure 5-1 shows the estimation results of γ for three different sets of parameters. For each parameter set, we randomly generate k sets of measurements, where k runs from 1 to 10, and apply the EM algorithm. To be consistent with the assumption throughout the thesis, we only generate measurements at the finest scale. For this problem, exact inference is still tractable, so at each iteration of the EM algorithm, we compute the conditional means and error variances exactly by matrix inversion. Figure 5-1 demonstrates that the estimate of γ converges to the true value as we increase the number of measurements.

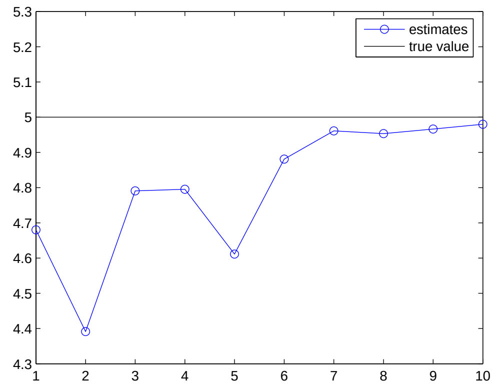
The accuracy of the estimates of φ can be increased by using a pyramidal graph with a larger number of nodes at the finest scale. Figure 5-2 shows the estimation results of φ using 5 sets of measurements generated from different sizes of pyramidal graphs. The numbers on the x-axis indicate the number of nodes at the finest scale.



true parameter: $\varphi = 0.5 \quad \gamma = 2$

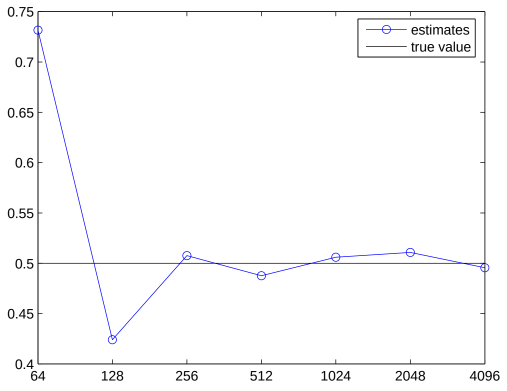


true parameter: $\varphi = 0.5 \quad \gamma = 0.1$

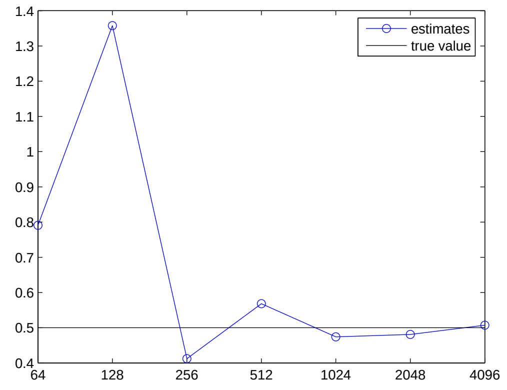


true parameter: $\varphi = 5 \quad \gamma = 5$

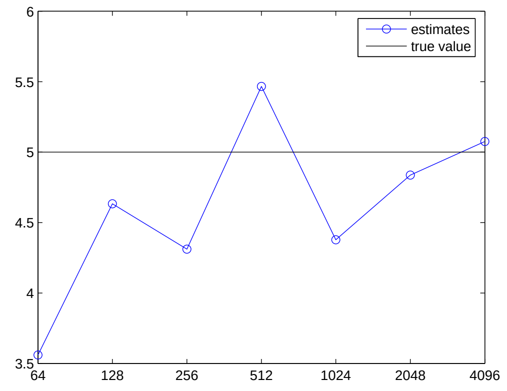
Figure 5-1: Parameter γ estimated from measurements generated by the pyramidal graph with 16×16 nodes at the finest scale. The x-axis show the number of sets of measurements, where each set is generated by the finest scale nodes of the pyramidal graph.



true parameter: $\varphi = 0.5 \quad \gamma = 2$



true parameter: $\varphi = 0.5 \quad \gamma = 0.1$



true parameter: $\varphi = 5 \quad \gamma = 5$

Figure 5-2: Parameter φ estimated from 5 sets of measurements generated by the finest scale nodes of the pyramidal graph. The x-axis show the number of nodes at the finest scale of the pyramidal graph.

	Model 1		Model 2		Model 3	
	φ	γ	φ	γ	φ	γ
True	0.5	2	0.5	0.1	5	5
Estimates1	0.5628	2.0655	0.5074	0.0987	4.1801	4.9800
Estimates2	0.4955	2.0390	0.5072	0.0993	5.0758	4.9130

Table 5.1: Parameter estimation results on measurements generated by pyramidal graphical models. Estimates1: Parameters estimated from 10 sets of measurements, where each set is randomly generated by 16×16 nodes at the finest scale of the pyramidal graph. Estimates2: Parameters estimated from 5 sets of measurements generated by 64×64 nodes.

For the pyramidal graphs with more than 1024 nodes at the finest scale, we use the multipole-motivated algorithm to estimate the conditional means and the wavelet-based low-rank algorithm to compute the covariances. As the number of nodes grows larger, the estimate of φ converges to the correct value. Table 5.1 lists the estimates of φ and γ using 10 sets of measurements generated by 16×16 nodes at the finest scale, and 5 sets of measurements generated by 64×64 nodes at the finest scale of the pyramidal graph.

Next, consider the surface used in Section 4.5 with dense measurements corrupted by high-level noise ($\sigma^2 = 25$, i.e. $\gamma = 0.04$) in Figure 4-9(c). Figure 5-3 shows the estimates of φ and γ at each iteration using two different inference algorithms. First, we apply the multipole-motivated algorithm with Gauss-Jacobi iterations (see Section 4.1.2) to estimate conditional means, and the wavelet-based low-rank approximation algorithm (see Section 4.3) to estimate error variances at individual nodes and covariances between the pairs of neighboring nodes. Secondly, the Lagrangian relaxation method (see Section 4.2) is used to estimate both conditional means and approximate covariances. Note that these two inference algorithms lead the estimates to converge to different values because the variances computed by the LR method produce a rather loose upper bound on the true variances. Table 5.2 lists the estimated values of the parameters after 25 iterations. Since the measurements are not generated using the pyramidal graph, the table lists the parameter value used for inference in Section 4.5 as 'true' value of φ, which produced satisfying results both visually and in terms of the RMS error.

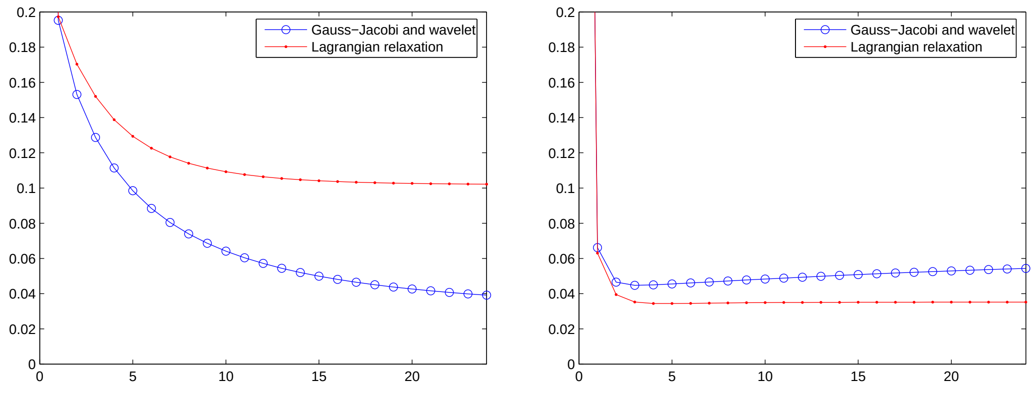


Figure 5-3: Estimates of parameters at each iteration. Left: φ, Right: γ.

	φ	γ
"true"	0.2	0.04
estimates using GJ & wavelet	0.0391	0.0543
estimates using LR	0.1022	0.0352

Table 5.2: Parameter estimation results on the synthetic surface used in Section 4.5 with dense measurements corrupted by noise with variance $\sigma^2 = 25$. The true φ is the value of parameter used for inference in Section 4.5.

Estimation of $\theta = (\alpha_1, \ldots, \alpha_M, \beta, \gamma)$ Now, let us consider applying the EM algorithm to the case with a larger number of parameters. For the analysis in this part, considering each element of α is more convenient than forcing the fixed ratio $\alpha_{m-1} = \frac{1}{4}\alpha_m$, so we use the J matrix defined in (5.3). In order to find $\theta^{(i)} = (\alpha_1^{(i)}, \ldots, \alpha_M^{(i)}, \beta^{(i)}, \gamma^{(i)})$ which maximizes the Q function, we solve the system of equations in (5.9). Using the Jacobi's formula: $\frac{\partial \log \det A}{\partial x} = tr(A^{-1}\frac{\partial A}{\partial x})$, we obtain

$$tr\left((J_{prior}^{-1})_{[m,m]} \cdot J_{sm}\right) - E[x_m^T J_{sm} x_m \mid y, \theta^{(i-1)}] = 0 \quad \forall m \tag{5.17}$$

$$tr\left(J_{prior}^{-1} \cdot J_T\right) - E[x^T J_T x \mid y, \theta^{(i-1)}] = 0 \tag{5.18}$$

$$\frac{N_{meas}}{\gamma} - E[\| y - Cx \|^2 | \, y, \theta^{(i-1)}] = 0 \tag{5.19}$$

Note that although J_{prior} is linear in terms of α_m's and β, it is not easy to represent J_{prior}^{-1} in terms of those parameters.

From (5.3), $J_T = \frac{J_{prior} - J_s}{\beta}$, and it follows that

$$\begin{aligned} tr\left(J_{prior}^{-1} \cdot J_T\right) &= \frac{1}{\beta} tr\left(I_N - J_{prior}^{-1} \cdot J_s\right) \\ &= \frac{1}{\beta}\left(N - \sum_{m=1}^{M} \alpha_m tr\left((J_{prior}^{-1})_{[m,m]} \cdot J_{sm}\right)\right) \end{aligned} \tag{5.20}$$

where we used the fact that J_s is a block diagonal matrix. Substituting (5.17) into the above equation and comparing the resulting expression with (5.18),

$$\beta = \frac{N_{nodes} - \sum_{m=1}^{M} \alpha_m E[x_m^T J_{sm} x_m \mid y, \theta^{(i-1)}]}{E[x^T J_T x \mid y, \theta^{(i-1)}]}. \tag{5.21}$$

Therefore, we can determine $\beta^{(i)}$ once we have estimates of α_m's. However, it is difficult to estimate α_m from (5.17) because of the term involving the inverse prior matrix.

Instead, recall from Theorem 4.5 that $(J^{-1})_{[m,m]}$ can be represented in terms of $\left(J_{[m,m]}\right)^{-1}$. Here, we ignore the second term in (4.15) and make the following approximation:

$$tr((J_{prior}^{-1})_{[m,m]} J_{sm}) \approx tr(((J_{prior})_{[m,m]})^{-1} J_{sm}). \tag{5.22}$$

Moreover, from (3.3), $(J_{prior})_{[m,m]} = \alpha_m J_{sm} + \beta c_m I$, where c_m is a constant which varies from scale to scale. Thus, it follows that eigenvalues of $(J_{prior})_{[m,m]}$ can be represented in terms of eigenvalues of J_{sm} which we denote as λ_i's:

$$\begin{aligned} tr(((J_{prior})_{[m,m]})^{-1} J_{sm}) &= \frac{\partial \log \det(J_{prior})_{[m,m]}}{\partial \alpha_m} \\ &= \frac{\partial \log \prod_i eig((J_{prior})_{[m,m]})}{\partial \alpha_m} \\ &= \frac{\partial \log \prod_i (\alpha_m \lambda_i + \beta c_m)}{\partial \alpha_m} \\ &= \sum_i \frac{\lambda_i}{\alpha_m \lambda_i + \beta c_m} \end{aligned} \tag{5.23}$$

Then, (5.17) reduces to the simple form:

$$\sum_i \frac{\lambda_i}{\alpha_m \lambda_i + \beta c_m} = E[x_m^T J_{sm} x_m \mid y, \theta^{(i-1)}]. \tag{5.24}$$

Thus, using (5.21) and (5.24), the values of α_m's and β can be computed iteratively. Once the eigenvalues of J_{sm} are computed at the beginning of the parameter estimation algorithm, the values can be used throughout the algorithm to compute $tr(((J_{prior})_{[m,m]})^{-1} J_{sm})$ efficiently.

However, we observed that the approximation in (5.22) is not close enough to give good estimates of the parameters. In Section 4.3, evaluating the lower bound on the second term of (4.15) is considered to improve the accuracy in variance estimation. However, it is not straightforward to apply the same approach to parameter estimation, since the second term in (4.15) can not be represented as a tractable function of α_m's and β.

5.3 The Log-determinant Bounds

The EM algorithm maximizes the lower bound on the log-likelihood function, but for the pyramidal graph with more than two free parameters for the prior model, it is intractable to maximize the lower bound because of the log partition function. In [55], a class of upper bounds on the log partition function is discussed for parameter estimation on a completely observed model, but the pyramidal graph is only partially observed when we have measurements only at the finest scale. In this and the next section, we discuss possible directions for approximate multiscale parameter estimation and leave their investigation as future research topics.

Instead of maximizing the lower bound in (5.6), we further relax the lower bound to get a more tractable form of the log partition function. This surrogate log partition function is derived in [22] to develop the Lagrangian relaxation method for the Gaussian case. Let us define a function related to the log partition function $\Phi(J)$:

$$\Psi(J) = -\frac{1}{N}\log detJ = \frac{2}{N}\Phi(J) + \log(2\pi) \tag{5.25}$$

where N is the dimension of the matrix J. Note that for all $c > 0$,

$$\Psi(cJ) = \Psi(J) - \log c. \tag{5.26}$$

Consider a valid additive decomposition $J = \sum_k J^k$. Using the convexity of the log-det function, we obtain a similar upper bound as in (2.26):

$$\Psi(J) = \Psi(\sum_k J^k) = \Psi(\sum_k \rho_k \left(\frac{J^k}{\rho_k}\right)) \le \sum_k \rho_k \Psi\left(\frac{J^k}{\rho_k}\right) = \sum_k \rho_k[\Psi(J^k) + \log \rho_k] \tag{5.27}$$

The upper bound can be explicitly minimized using the optimal weights defined in (2.27), with the tightest bound given as

$$\Psi(J) \le \tilde{\Psi}(J) \triangleq -\log \sum_k \exp^{-\Psi(J^k)} \tag{5.28}$$

Now, we consider a decomposition of the prior matrix $J_{prior} = \alpha J_S + \beta J_T$. Substituting $J^1 = \alpha J_S$ and $J^2 = \beta J_T$, and using the property in (5.26), we obtain a surrogate log partition function:

$$\Phi(J_{prior}) = -\frac{1}{2}\log detJ_{prior} \le -\frac{N}{2}\log(\alpha e^{\frac{\log detJ_S}{N}} + \beta e^{\frac{\log detJ_T}{N}}) \triangleq \tilde{\Phi}(J_{prior}) \tag{5.29}$$

Since the values of $c_S \triangleq e^{\frac{\log detJ_S}{N}}$ and $c_T \triangleq e^{\frac{\log detJ_T}{N}}$ do not change during the parameter estimation process, they can be computed once at the beginning of the algorithm. Using the notation $\eta_1^{(i-1)} \triangleq E[x^T J_S x|y, \theta^{(i-1)}]$, $\eta_2^{(i-1)} \triangleq E[x^T J_T x|y, \theta^{(i-1)}]$, and $\eta_3^{(i-1)} \triangleq E[\| y - Cx \|^2 |y, \theta^{(i-1)}]$ the Q function using the surrogate log partition function has the following simple form:

$$\tilde{Q}(\theta, \theta^{(i-1)}) = \frac{1}{2}(-\alpha\eta_1^{(i-1)} - \beta\eta_2^{(i-1)} + N\log(c_S\alpha + c_T\beta) - \gamma\eta_3^{(i-1)} + N_{meas}\log\gamma) \tag{5.30}$$

where we ignored the terms which are irrelevant to the parameters. The surrogate Q

function is a concave lower bound on the original Q function, and we observed that the error between the true value and the lower bound is about 10%. Unfortunately, however, the point which maximizes the surrogate Q function does not match the maximum point of the Q function, so using the surrogate log partition function for the parameter estimation process fails to provide satisfying results.

5.4 Discussion

The pyramidal graphical model has rich modeling power when the ratios between the parameters are allowed to vary as illustrated in Figure 3-3. However, we observed in this section that considering more than one free parameter for the prior model makes the parameter estimation problem intractable to solve because of the log partition function. In the previous sections, we considered approximating the log partition function as a tractable surrogate function, but it not produce satisfying results in parameter estimation given partially observed data.

The Lagrangian relaxation method, described in Section 2.2.3 and Section 4.2, minimizes the bound on the log partition function not only with respect to the weights, but also with respect to the optimal decomposition. Since the log partition functions of tractable subgraphs are easy to manipulate, we may obtain tighter, yet tractable bounds on the log partition function. However, recall that in order to apply the LR method, we need to perform inference on each subgraph. Thus, although we may use the LR method to find an upper bound on $\Phi(J)$, where $J = J_{prior} + C^T R^{-1} C$, it is not straightforward to apply the LR method to parameter estimation to compute tractable upper bounds on the log partition function of the prior model $\Phi(J_{prior})$ [1].

[1]Since J_{prior} is near-singluar, applying inference algorithms to compute J_{prior}^{-1} does not produce stable values.

Chapter 6

Conclusions

In this chapter, we summarize the contributions of this thesis, and discuss open research problems motivated by our work.

6.1 Contributions

In this thesis, we proposed a pyramidal graphical model, which is an extension of the multiscale tree-structured models augmented by in-scale edges, to incorporate both intra- and inter- scale dependency structures among Gaussian random variables. The prior model on the pyramidal graph is simply defined to be the thin-membrane model within each scale as well as between parent-child pairs. The pyramidal graph has rich modeling capability and produces smooth and slowly decaying correlations. Moreover, conditioned on other scales, the correlations within one scale decay quickly, motivating us to develop efficient inference algorithms in which far-apart nodes communicate approximately through coarser scales and nearby nodes interact at finer scales.

Our multipole-motivated inference algorithms consist of two steps: in the tree inference steps, different scales share information through a spanning tree embedded in the pyramidal graph. During the in-scale inference steps, nearby nodes within each scale pass messages to each other to obtain smooth estimates. This iterative procedure is guaranteed to converge thanks to the walk-summability of the pyramidal graph. We have presented empirical results to show that the iterations converge much faster than in the counterpart monoscale model.

A great advantage of using the pyramidal graph over other multiscale approaches such as coarse-to-fine multigrid methods, is that we are blessed with many efficient inference algorithms recently developed for Gaussian graphical models. Using the adaptive ET algorithms in our multipole-motivated inference, we improved the convergence rates even further. The Lagrangian relaxation method decomposes the pyramidal graph into simple tractable subgraphs and computes the optimal estimates as well as upper bounds on covariances by solving inference problems on each subgraph. When more accurate values for variances are needed, we may apply recently introduced low-rank approximation algorithms.

It is often required to correct or update estimates locally to incorporate new information. Even when extremely efficient inference algorithms are available, it is time-consuming to re-start the estimation process for large-scale problems. Using the hierarchical structure of the pyramidal graph and adaptive iteration methods, we developed the re-estimation algorithm to update estimates locally. Two motivating examples are considered. First, it is shown that smoothness priors tend to blur the discontinuities. By modifying the prior model locally and applying the re-estimation algorithm, we computed more accurate estimates around the discontinuities. Secondly, we introduced a new set of measurements to a local region and updated the estimates rapidly to incorporate the new measurements.

6.2 Open Research Questions

Pyramidal Graphs with Other Prior Models In this thesis, we focused on the pyramidal graph with a simple prior model extended from the thin-membrane model. It is of interest to explore other modeling approaches which may produce better estimation results or may be more appropriate for other applications.

First of all, the thin-plate model is also commonly used as a smoothness prior, and it has been observed to produce better estimates when the surface of interest has steep gradients. We can consider using the thin-plate model to define J_s or J_t

in the prior model of the pyramidal graph. Since the resulting graph is no longer walk-summable, the convergence of multipole-motivated inference algorithms are not guaranteed, but the Lagrangian relaxation method can be used for inference.

Alternatively, we may to use wavelet coefficients in modeling the coarser scale variables as commonly used in multiscale tree modeling [9, 10, 45]. Since the support lengths of general wavelet transforms are longer than two, additional edges are required between adjacent scales to capture parent-child dependencies correctly. Even so, the inference algorithm alternating the tree inference steps with the in-scale inference steps may still be applicable with just minor changes.

Finally, we considered only Gaussian processes here, and it would be an interesting extension to consider non-Gaussian cases. Specifically, nodes with discrete random variables play an important role in many applications [24, 42, 43]. However, in this case, it is not straightforward to design a hidden node to represent a coarser version of its children.

Structural Optimization of Pyramidal Graphs The pyramidal graph we suggested in this thesis has a nearest neighbor grid model at each scale and quadtree structure to connect different scales. In general, the computational complexity of an inference algorithm critically depends on the sparsity of the graph. So, we are interested in getting sparser structures by *graph-thinning*, i.e. reducing the number of edges while minimizing the resulting information loss. Johnson *et al.* [21] formulated a convex optimization problem for thinning Markov models using the maximum entropy relaxation principle. Using this framework, we are especially interested in thinning the finest scale with the aid of the coarser scales.

Alternatively, instead of starting from specifying the structure of the pyramidal graph, we may consider the problem of building the pyramidal graph given the probability distribution at the finest scale. This problem is closely related to the *multiscale realization problem* in which the goal is to construct a distribution with respect to a sparse multiscale graph $\mathcal{G}$ so that the marginal distribution of the finest scale is close to the original true (or empirical) distribution. Tucker [51] studied the problem for the case when $\mathcal{G}$ is tree-structured, and Chandrasekaran [5] addressed the problem for general graphs.

Adaptive and Interactive Re-estimation We have presented the re-estimation problem as updating the estimates given some local changes in the information parameters J or h. Using this framework, we considered two cases when we need to solve the re-estimation problem.

First, estimates of discontinuities blurred during the estimation process can be recovered more accurately by modifying the prior model locally and then applying the re-estimation algorithm. We demonstrated simulation results based on the assumption that those changes have been computed previously by edge detecting algorithms or by human analysts. It is of interest to incorporate the automatic edge detecting method into our re-estimation algorithm to detect the discontinuities and modify the prior model adaptively depending on the level of discontinuity.

Secondly, the re-estimation algorithm can update the estimates rapidly when a set of new measurements are introduced. This procedure may also enhance the efficiency of human-computer interaction by rapidly updating the estimates based on human input. In many cases, estimation algorithms are used as pre-processing steps for human experts, and it is important for these algorithms to provide fast and accurate feedback to humans.

In addition, we proposed algorithms to update optimal estimates but not error covariance computations. The re-estimation algorithm will be more powerful with efficient error variance update methods, which is an important and open problem.

Multiscale Parameter Estimation The parameter estimation algorithm we presented in Chapter 5 only worked well when we have a single free parameter to control the overall strength of the prior constraints. Allowing more freedom in the parameters of the pyramidal graph will greatly enhance the modeling power of the pyramidal graph. Finding a surrogate log partition function may provide a starting point for solving parameter estimation problem in multiscale pyramidal graphs.

Appendix A

Proof of Theorem 4.4

We prove the following lemma first.

Lemma A.1. *At sub-iteration n, the following conditions are satisfied:*

1. *Before step 2(a), $u^{(n)}$ has all incoming messages up-to-date.*
2. *After step 2(b), at each node in every subgraph, there exist at most one edge for which the incoming message is not up-to-date.*

Proof. We prove by induction.

At sub-iteration 1: Since the BP algorithm in each subgraph has converged, every message is up-to-date before step 2(a), and the first condition holds. After Step 2(b), only the node potentials of $u^{(1)}$ has been changed, so $u^{(1)}$ has all incoming messages up-to-date, and at every other node $i \neq u^{(1)}$ in each subgraph, only the message from the direction $u^{(1)}$ is not up-to-date, so the second condition holds.

At sub-iteration $(n-1)$: Assume that the two conditions are satisfied at iteration $(n-1)$. Since the node potential of $u^{(n-1)}$ is changed at step 2(b), every outward message from node $u^{(n-1)}$ is out-of-date before a set of messages are updated at step 2(c). For all nodes $i \in V$ except $u^{(n-1)}$, there exist a neighbor j such that $u^{(n-1)} \in \mathcal{T}_{j\setminus i}$ and $m_{j\to i}$ is not up-to-date. From the assumption that there exist at most one incoming message which is not up-to-date, all incoming messages $m_{k\to i}$ are up-to-date for $k \in \mathcal{N}(i)$, $k \neq j$.

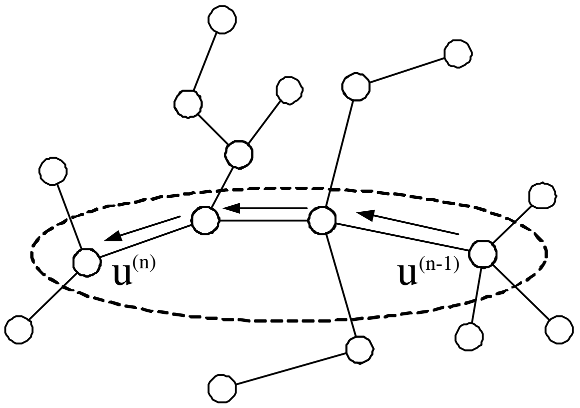


Figure A-1: An illustration of the efficient message passing scheme for the Lagrangian relaxation method.

At sub-iteration n: First, for any node i such that there exist a neighboring node j which satisfies $u^{(n-1)} \in \mathcal{T}_{j\backslash i}$ and $u^{(n)} \in \mathcal{T}_{j\backslash i}$ (the nodes outside of the circled area in Figure A-1), the information regarding $u^{(n-1)}$ and $u^{(n)}$ comes from the same incoming message $m_{j\to i}$. Since all other messages $m_{k\to i}$ are up-to-date by the assumption at sub-iteration $(n-1)$, and since those messages are not disturbed by the change in $u^{(n)}$, they are still up-to-date after step 2(b). So the second condition holds for the nodes outside of the circled area.

At sub-iteration $(n-1)$, all incoming messages to $u^{(n-1)}$ are up-to-date before step 2(a). So, when $u^{(n-1)}$ updates the message to its neighbor i on the path to $u^{(n)}$, the message $m_{u^{(n-1)}\to i}$ is up-to-date. Since i had up-to-date incoming messages from all other nodes except $u^{(n-1)}$, it now has all incoming messages up-to-date. Then, i updates the message to its neighbor j such that $u^{(n)} \in \mathcal{T}_{j\backslash i}$. This process can be repeated until the message update procedure reaches $u^{(n)}$, and all the nodes on the path from $u^{(n-1)}$ to $u^{(n)}$ have all incoming messages up-to-date.

When we change the node potential of $u^{(n)}$ at step 2(c), it does not affect the incoming messages at $u^{(n)}$, so the first condition is satisfied. For all other nodes on the path from $u^{(n-1)}$ to $u^{(n)}$, the message coming from the direction of $u^{(n)}$ becomes out-of-date, but since this is the only incoming message which is not up-to-date, the second condition holds for all nodes inside the circled area in Figure A-1 as well. □

The proof of Theorem 4.4 directly follows from the first condition of the lemma.

Bibliography

[1] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," International Computer Science Institute, Tech. Rep. ICSI-TR-97-021, 1998.

[2] C. A. Bouman and M. Shapiro, "A multiscale random field model for Bayesian image segmentation," *IEEE Transactions on Image Processing*, vol. 3, no. 2, pp. 162–177, Mar. 1994.

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

[4] W. L. Briggs, *A Multigrid Tutorial.* Philadelphia, PA: SIAM, 1987.

[5] V. Chandrasekaran, "Efficient learning, realization, and inference in Gaussian graphical models: maximum-entropy, marginalization-invariance Markovianity, and walk-sums," 2006, master's thesis proposal, MIT.

[6] V. Chandrasekaran, J. K. Johnson, and A. S. Willsky, "Estimation in Gaussian graphical models using tractable subgraphs: A walk-sum analysis," 2007, (preprint).

[7] K. C. Chou, A. S. Willsky, and A. Benveniste, "Multiscale recursive estimation, data fusion, and regularization," *IEEE Transactions on Automatic Control*, vol. 39, no. 3, pp. 464–478, Mar. 1994.

[8] M. L. Comer and E. J. Delp, "Segmentation of textured images using a multiresolution Gaussian autoregressive model," *IEEE Transactions on Image Processing*, vol. 8, no. 3, pp. 408–420, Mar. 1999.

[9] M. S. Crouse, R. Nowak, and R. Baraniuk, "Wavelet-based statistical signal processing using hidden Markov models," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 886–902, Apr. 1998.

[10] K. Daoudi, A. B. Frakt, and A. S. Willsky, "Multiscale autoregressive models and wavelets," *IEEE Transactions on Information Theory*, vol. 45, no. 3, pp. 828–845, Apr. 1999.

[11] V. Delouille, R. N. Neelamani, and R. G. Baraniuk, "Robust distributed estimation using the embedded subgraphs algorithm," *IEEE Transactions on Signal Processing*, vol. 54, no. 8, pp. 2998–3010, Aug. 2006.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum-likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 39, pp. 1–38, 1977.

[13] P. W. Fieguth, W. C. Karl, and A. S. Willsky, "Multiresolution optimal interpolation and statistical analysis of TOPEX/POSEIDON setellite altimetry," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 33, no. 2, pp. 280–292, Mar. 1995.

[14] ——, "Efficient multiresolution counterparts to variational methods for surface reconstruction," *Comput. Vis. Image Understanding*, vol. 70, no. 2, pp. 157–176, May 1998.

[15] P. Fieguth, "Multipole-motivated reduced-state estimation," in *Proc. ICIP*, vol. 1, 1998, pp. 635–638.

[16] B. Gidas, "A renormalization group approach to image processing problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 2, pp. 164–180, Feb. 1989.

[17] C. Graffigne, F. Heitz, P. Pérez, F. Prêteux, M. Sigelle, and J. Zerubia, "Hierarchical Markov random field models applied to image analysis: a review," in *SPIE Conf. Neural, Morphological Stochastic Methods in Image Signal Processing*, vol. 2568, San Diego, CA, Jul. 1995, pp. 12–17.

[18] L. Greengard and V. Rokhlin, "A fast algorithm for particle simulations," *J. Computational Physics*, vol. 73, no. 2, pp. 325–348, 1987.

[19] S. Haykin, *Communication Systems*, B. Zobrist, Ed. John Wiley & Sons, Inc., 2001.

[20] F. Heitz, P. Pérez, and P. Bouthemy, "Multiscale minimization of global energy functions in some visual recovery problems," in *CVGIP: Image Understanding*, vol. 59, no. 1, 1994, pp. 125–134.

[21] J. K. Johnson, "Recursive variational methods for approximate inference in graphical models," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, 2007, in preparation.

[22] J. K. Johnson, V. Chandrasekaran, and A. S. Willsky, "Learning Markov structure by maximum entropy relaxation," in *Eleventh International Conference in Artificial Intelligence and Statistics*, San Juan, Puerto Rico, Mar. 2007.

[23] J. Johnson and A. Willsky, "A recursive model-reduction method for estimation in Gaussian markov random fields," MIT, Cambridge, MA, LIDS Technical Report, 2007, (in review for publication).

[24] M. I. Jordan, Ed., *Learning in Graphical Models.* Cambridge, MA: MIT Press, 1998.

[25] M. I. Jordan, "Graphical models," *Statistical Sci.*, vol. 19, pp. 140–155, 2004.

[26] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine Learning*, vol. 37, pp. 183–233, 1999.

[27] Z. Kato, M. Berthod, and J. Zerubia, "Multiscale Markov random field models for parallel image classification," in *Proceddings ICCV*, Berlin, May 1993.

[28] ——, "A hierarchical Markov random field model and multitemperature annealing for parallel image classification," *Graphical Models and Image Processing*, vol. 58, no. 1, pp. 18–37, Jan. 1996.

[29] Z. Kato, J. Zerubia, and M. Berthod, "Unsupervised parallel image classification using Markovian models," *Pattern Recognition*, vol. 32, no. 4, pp. 591–604, Apr. 1999.

[30] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 1568–1583, Oct. 2006.

[31] S. L. Lauritzen, *Graphical Models.* Oxford: Oxford University Press, 1996.

[32] J. Li, R. M. Gray, and R. A. Olshen, "Multiresolution image classification by hierarchical modeling with two-dimensional hidden Markov models," *IEEE Transactions on Image Processing*, vol. 46, pp. 1826–1841, Aug. 2000.

[33] L. Ljung, *System Identification: Theory for the User.* Upper Saddle River N. J.: Prentice Hall, 1999.

[34] M. R. Luettgen, W. C. Karl, and A. S. Willsky, "Efficient multiscale regularization with application to optical flow," *IEEE Transactions on Image Processing*, vol. 3, no. 1, pp. 41–64, Jan. 1994.

[35] D. M. Malioutov, J. K. Johnson, M. J. Choi, and A. S. Willsky, "Low-rank variance approximation in GMRF models: single and multi-scale approaches," (in preparation).

[36] D. M. Malioutov, J. K. Johnson, and A. S. Willsky, "Low-rank variance estimation in large-scale GMRF models," in *IEEE ICASSP*, May 2006.

[37] ——, "Walk-sums and belief propagation in Gaussian graphical models," *Journal of Machine Learning Research*, vol. 7, pp. 2031–2064, Oct. 2006.

[38] ——, "GMRF variance approximation using spliced wavelet bases," in *IEEE ICASSP*, Apr. 2007.

[39] S. Mallat, *A Wavelet Tour of Signal Processing.* Academic Press, 1998.

[40] J. Pearl, *Probabilistic Reasoning in Intelligent Systems.* San Mateo, CA: Morgan Kaufman, 1988.

[41] K. B. Petersen and M. S. Pedersen, "The matrix cookbook," Technical University of Denmark, Feb. 2006, version 20051003. [Online]. Available: http://www2.imm.dtu.dk/pubdb/p.php?3274

[42] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," vol. 77, no. 2, pp. 257–286, Feb. 1989.

[43] I. U. Rahman, I. Drori, V. C. Stodden, D. L. Donoho, and P. Schröder, "Multiscale representations for manifold-valued data," *Multiscale Model. Simul.*, vol. 4, pp. 1201–1232, 2005.

[44] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning.* The MIT Press, 2005.

[45] J. Romberg, H. Choi, and R. Baraniuk, "Bayesian tree-structured image modeling using wavelet-domain hidden markov models," *IEEE Transactions on Image Processing*, vol. 10, no. 7, pp. 1056–1068, Jul. 2001.

[46] W. Rudin, *Real and Complex Analysis.* McGraw-Hill Science, 1986.

[47] T. P. Speed and H. T. Kiiveri, "Gaussian Markov distributions over finite graphs," *The Annals of Statistics*, vol. 14, no. 1, pp. 138–150, Mar. 1986.

[48] E. B. Sudderth, M. J. Wainwright, and A. S. Willsky, "Embedded trees: Estimation of Gaussian processes on graphs with cycles," *IEEE Transactions on Signal Processing*, vol. 52, no. 11, pp. 3136–3150, 2004.

[49] E. B. Sudderth, "Graphical models for visual object recognition and tracking," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, May 2006.

[50] D. Terzopoulous, "Image analysis using multigrid relaxation methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, pp. 129–139, Mar. 1986.

[51] D. Tucker, "Stochastic realization theory for exact and approximate multiscale models," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, Sep. 2005.

[52] M. J. Wainwright, "Stochastic processes on graphs with cycles: Geometric and variational approaches," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, Jan. 2002.

[53] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky, “Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches,” *IEEE Transactions on Information Theory*, vol. 51, no. 11, pp. 3697–3717, Nov. 2005.

[54] ——, “A new class of upper bounds on the log partition function,” *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2313–2335, Jul. 2005.

[55] M. J. Wainwright and M. I. Jordan, “Graphical models, exponential families, and variational inference,” University of California, Berkeley, Tech. Rep. 649, 2003.

[56] A. S. Willsky, “Multiresolution Markov models for signal and image processing,” *Proceedings of the IEEE*, vol. 90, no. 8, pp. 1396–1458, Aug. 2002.