A GRAPHICAL MODEL FOR THE COMMUNICATIONS CHANNEL

BY

THOMAS JOHANNES RIEDL

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2009

Urbana, Illinois

Adviser:

Professor Andrew C. Singer

ABSTRACT

We consider the problem of channel modeling and channel estimation. The widely used wide sense stationary uncorrelated scattering model for the communications channel neglects correlations between different multipath arrivals, but this seems to oversimplify the real channel in many cases. One example is the underwater acoustic channel, whose impulse response is fairly continuous in delay and hence indeed exhibits a certain correlation structure in delay.

To address this shortcoming we introduce a novel channel model that is based on a Gaussian Markov random field (MRF) for the complex channel gains. This graphical model is used to capture the local nature of the statistical dependencies (in time and space) of the channel taps.

In order for the MRF model to fit the actual physical channel well, its parameters must be adapted appropriately. Our approach is to find the maximum likelihood (ML) estimate of theses parameters based on given observations. Once these parameters are known the MRF model can then either be used for channel estimation directly or it can be embedded into an iterative (turbo) receiver, where it is expected to improve the data estimation performance significantly as the parameterized MRF carries prior knowledge on the channel.

TABLE OF CONTENTS

CHAPTER 1

Introduction

Physical channels such as underwater acoustic channels and ionospheric radio channels which result in time-varying multipath propagation of the transmitted signal may be characterized mathematically as time-varying linear filters [1]. At the receiver, the incident signal is sampled and hence the effect of this time-varying channel on the transmitted sequence $x[i]$ can be modeled by a discrete time, time-varying filter $h[i, j]$. The sampled received signal $y[i]$ can then be represented as

$$y[i] = \sum_{j=0}^{L-1} h[i, j]x[i - j] + w[i], \qquad i = 0, \ldots, M - 1 \tag{1.1}$$

where $w[i]$ denotes additive complex white Gaussian noise with zero mean and circular symmetric variance σ_n^2, M is the total number of observations and the delay spread of the channel is assumed to be at most L symbols.

The complex gains $h[i, j]$ are modeled as random variables, where i indicates time and different j correspond to different multipath arrivals. It is typically assumed that $h[i, j]$ is Gaussian, uncorrelated in delay j and wide sense stationary in time i. The variability of the wireless channel over time is then fully characterized by the autocorrelation of $h[i, j]$ in time defined by $R_j(m) = E[h[i + m, j]h[i, j]^\star]$. Random processes of that kind can be well approximated by a Gaussian *autoregressive* (AR) process, i.e., a random process that is generated by an autoregressive filter when driven by white Gaussian noise. This

channel model captures the dependencies of $h[i, j]$ in time i but neglects any dependence of $h[i, j]$ in delay j. Figure 1.1 shows a typical channel impulse re-

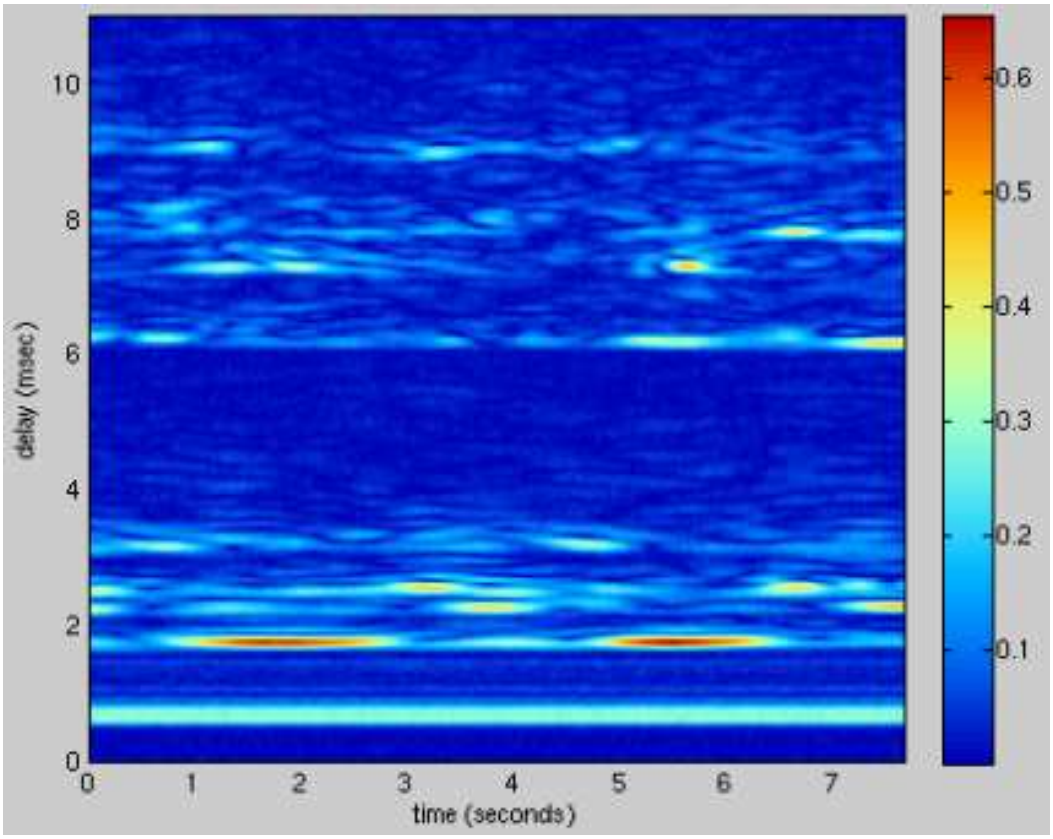

Figure 1.1: Typical channel realization

sponse as measured from real underwater acoustic communication data. A certain continuity in delay is observed but this kind of local dependency in delay is neglected in the channel model just described. The channel estimators proposed in [2–5] are built on this model and hence will suffer a performance loss whenever they face a channel exhibiting strong dependencies in delay.

The underwater acoustic communications channel is rapidly time-varying and often sparse. Experimental results such as the channel plot above, however, suggest that the statistics of this channel are substantially less fluctuating and the statistical dependencies between consecutive taps in time and delay are local and relatively strong. Drawing on theses observations we propose a new ap-

proach to channel modeling and channel estimation that is based on a *Markov random field* (MRF) model for the complex channel gains. This graphical model effectively captures the local statistical dependencies of the channel taps and can easily be embedded in iterative (turbo) receivers [6]. This is expected to improve the data estimation performance significantly as the MRF carries prior knowledge of the channel.

Markov random fields have been used successfully in the field of image processing and restoration for modeling spatial continuity [7]. MRFs are also used in the context of *magnetic resonance* (MR) imaging, where they help to unwrap MR phase images by imposing smoothness constraints on the true phase function [8].

The mathematical tools that we will draw on in our derivations later are introduced in Chapter 2. Chapter 3 is dedicated to our new approach to channel modeling and channel estimation based on a MRF for the complex channel gains. We elaborate on the theory behind MRFs in Section 3.2. This concept allows us to carry the conditional independence property of a *Markov chain* over to two-dimensional random processes, like $h[i, j]$, and to define the global behavior of $h[i, j]$ by local constraints on the probability distribution of $h[i, j]$. A performance evaluation of the proposed MRF channel model is presented in Section 3.5. Channel estimation based on our MRF model yields competitive results in case there is no correlation in delay and it clearly outperforms common channel estimation techniques if the channel is correlated in delay. The MRF is parametrized and the channel estimation performance depends on how well the MRF model parameters fit the actual channel. In Chapter 4 we propose an algorithm capable of finding appropriate parameters. Our approach involves finding the *maximum likelihood* (ML) estimate of theses parameters based on given observations.

CHAPTER 2

Fundamentals

2.1 Factor Graphs and the Sum-Product Algorithm

Many algorithms in artificial intelligence, signal processing and digital communications boil down to calculating the marginals of a possibly complicated global function of many variables. In many instances this global function factors into a product of local functions, each of which depends on only a subset of the variables. Exploiting the distributive law then potentially reduces the computational complexity of the marginalization dramatically. The way in which a global function factors can be visualized by a bipartite graph that expresses which variables each local function has as its arguments. A graph of this kind is referred to as a *factor graph*. This graphical representation then readily suggests how we can exploit the distributive law for the marginalization of the global function. The algorithm that operates on the factor graph in order to obtain the various marginals derived from the global function is called the *sum-product algorithm*.

2.1.1 Factor graphs

Let $x_1, x_2, \ldots, x_n$ be a collection of variables, in which, for each i, x_i takes on values in some domain $\mathbb{A}_i$ and let $g(x_1, \ldots, x_n)$ be a real valued function of

these variables, i.e.,

$$g : \mathbb{A}_1 \times \mathbb{A}_2 \times \ldots \times \mathbb{A}_n \to \mathbb{R}$$

Each element of the domain $\mathbb{D}$ of g is called a particular configuration of the variables. Suppose that the global function $g(x_1, \ldots, x_n)$ factors into a product of several other functions f_i called *local functions* in the remainder. Each local function f_i depends on only a subset $\mathbb{X}_i$ of the set of all variables $\{x_1, \ldots, x_n\}$ and so $g(x_1, \ldots, x_n)$ can be written as

$$g(x_1, \ldots, x_n) = \prod_{i \in \mathbb{I}} f_i(\mathbb{X}_i)$$

where $\mathbb{I}$ denotes a finite index set.

With the above notation we can now readily define the notion of a *factor graph.*

Definition 1. *A factor graph is a bipartite graph that contains two different types of nodes:* variable nodes *and* factor nodes. *Each variable x_i is associated with a variable node and each local function f_i is associated with a factor node. There is an edge connecting the variable node x_i to the factor node f_j if and only if x_i is an argument of f_j.*

We illustrate the definition of a factor graph by an example. Consider a global function $g(x_1, \ldots, x_5)$ of five variables and suppose that this function factors as follows:

$$g(x_1, \ldots, x_5) = f_A(x_1) f_B(x_2) f_C(x_1, x_2, x_3) f_D(x_3, x_4) f_E(x_3, x_5)$$

The global function $g(x_1, \ldots, x_5)$ can then be represented by a factor graph as

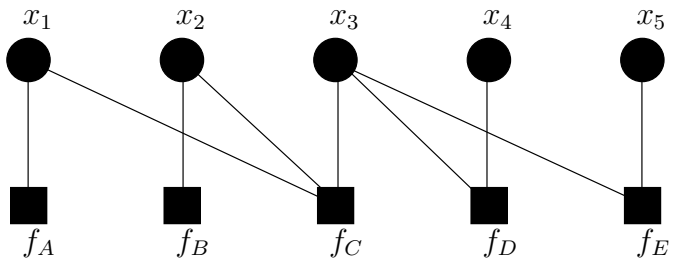

Figure 2.1: A factor graph for the product $f_A(x_1)f_B(x_2)f_C(x_1, x_2, x_3)\cdot f_D(x_3, x_4)f_E(x_3, x_5)$

shown in Figure 2.1.

2.1.2 Sum-product algorithm

In this subsection we will derive an algorithm known as the *sum-product algorithm* that operates on a factor graph in order to obtain all the *marginal* functions $g_i(x_i)$ associated with a given global function $g(x_1, \ldots, x_n)$. These marginal functions are obtained by integrating $g(x_1, \ldots, x_n)$ over all configurations of the variables such that x_i is fixed, i.e.,

$$g_i(x_i) = \int_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n} g(x_1, \ldots, x_n)$$

This type of integration will be frequently used in the remainder and it makes sense to introduce a more concise notation for it. Instead of indicating the variables being integrated over, we indicate those variables not being integrated over. This notation was introduced in [9] and was called a *summary* there. By use of this notation we can now describe the above integral more elegantly. The summary of $g(x_1, \ldots, x_n)$ for x_i is denoted by

$$g_i(x_i) = \int_{\sim x_i} g(x_1, \ldots, x_n) \tag{2.1}$$

The goal of this subsection is the derivation of an efficient method for computing the marginals $g_i(x_i)$. The method we are going to propose exploits the way the global function $g(x_1, \ldots, x_n)$ factors by employing the distributive law for the marginalization.

For now we assume that the function $g(x_1, \ldots, x_n)$ can be represented by a cycle-free factor graph, i.e., a factor tree. We consider the rooted factor tree T_{x_i} that has x_i as its root and all other nodes as its children.

Theorem 1. *If the edge connecting the factor node f_j with the variable node x_p is removed from a given factor tree, we obtain two separate trees $T_{f_j \to x_p}$ and $T_{x_p \to f_j}$. $T_{f_j \to x_p}$ denotes the one that includes the factor node f_j, $T_{x_p \to f_j}$ the one that includes the variable node x_p.*

We are interested in developing an expression for equation 2.1. Let us pick any factor node f_j in the given rooted factor tree T_{x_i}, take x_p as the variable node parenting f_j and designate f_j the root of $T_{f_j \to x_p}$. Then without loss of generality g can be written in the form

$$g(x_1, \ldots, x_n) = G(\mathbb{X}_i, x_p) F_{f_j \to x_p}(x_p, \mathbb{X}_p)$$

where $\mathbb{X}_p$ is the set of all variables in the rooted subtree $T_{f_j \to x_p}$, $F_{f_j \to x_p}(x_p, \mathbb{X}_p)$ is the product of all local functions in that subtree, the disjoint sets $\mathbb{X}_i$ and $\{x_p\}$ comprise all variables in the tree that remains when $T_{f_j \to x_p}$ is cut out of T_{x_i} and $G(\mathbb{X}_i, x_p)$ is the product of all local functions in that remaining tree. This decomposition is represented by the generic factor tree of Figure 2.2, in which $F_{f_j \to x_p}(x_p, \mathbb{X}_p)$ is shown in expanded form.

Definition 2. *The variables x_{jl} are the children of f_j in the rooted subtree $T_{f_j \to x_p}$, the set $\mathbb{X}_{jl}$ comprises the variables in the subtree $T_{x_{jl} \to f_j}$ that are dif-*

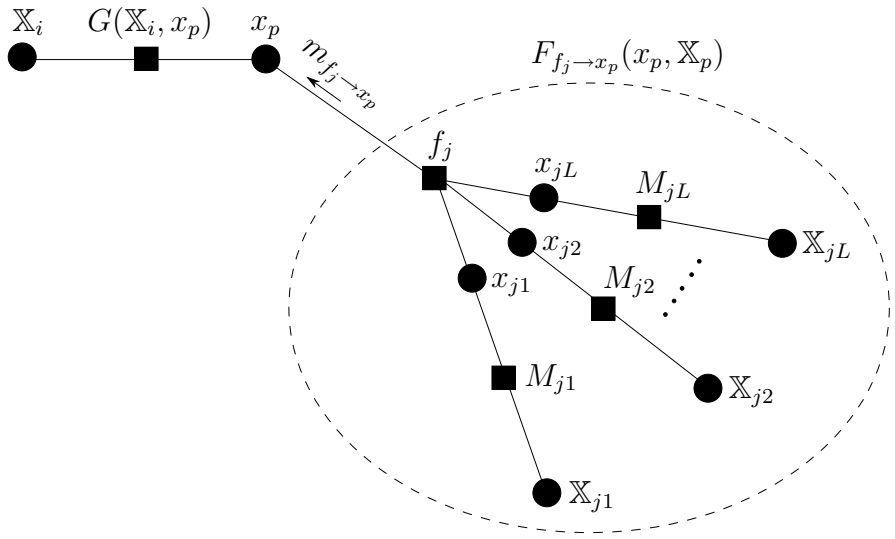

Figure 2.2: A generic factor tree

ferent from x_{jl} and the function $M_{jl}(x_{jl}, \mathbb{X}_{jl})$ is the product of the local functions in the subtree $T_{x_{jl} \to f_j}$.

So

$$F_{f_j \to x_p}(x_p, \mathbb{X}_p) = f_j(x_p, x_{j1}, \ldots, x_{jL}) \prod_l M_{jl}(x_{jl}, \mathbb{X}_{jl})$$

Now, by the distributive law and noting that the sets $\mathbb{X}_i$, $\{x_p\}$, $\mathbb{X}_p$ are pairwise disjoint, we obtain

$$\begin{aligned} g_i(x_i) &= \int_{\sim x_i} g(x_1, \ldots, x_n) \\ &= \int_{\sim x_i} G(\mathbb{X}_i, x_p) F_{f_j \to x_p}(x_p, \mathbb{X}_p) \\ &= \int_{(\mathbb{X}_i \cup \{x_p\}) \setminus \{x_i\}} G(\mathbb{X}_i, x_p) \int_{\mathbb{X}_p} F_{f_j \to x_p}(x_p, \mathbb{X}_p) \\ &= \int_{\sim x_i} G(\mathbb{X}_i, x_p) \underbrace{\int_{\sim x_p} F_{f_j \to x_p}(x_p, \mathbb{X}_p)}_{m_{f_j \to x_p}(x_p)} \end{aligned}$$

Note that $\int_{\sim x_i}(\cdot) = \int_{(\mathbb{X}_i \cup \{x_p\} \cup \mathbb{X}_p) \setminus \{x_i\}}(\cdot)$ in the first two equations and $\int_{\sim x_i}(\cdot) =$

$\int_{(\mathbb{X}_i \cup \{x_p\}) \setminus \{x_i\}}(\cdot)$ in the last equation. Introducing the notation $m_{f_j \to x_p}(x_p)$ and again exploiting the distributive law yields

$$\begin{aligned} m_{f_j \to x_p}(x_p) &= \int_{\sim x_p} F_{f_j \to x_p}(x_p, \mathbb{X}_p) \\ &= \int_{x_{j1}, \ldots, x_{jL}} f_j(x_p, x_{j1}, \ldots, x_{jL}) \prod_l \int_{\mathbb{X}_{jl}} M_{jl}(x_{jl}, \mathbb{X}_{jl}) \\ &= \int_{\sim x_p} f_j(x_p, x_{j1}, \ldots, x_{jL}) \prod_l \int_{\mathbb{X}_{jl}} M_{jl}(x_{jl}, \mathbb{X}_{jl}) \end{aligned} \tag{2.2}$$

where we used the pairwise disjointedness of the sets $\{x_{j1}, \ldots, x_{jL}\}$, $\mathbb{X}_{jl}$. This then allows us to write $g_i(x_i)$ as

$$g_i(x_i) = \int_{\sim x_i} G(\mathbb{X}_i, x_p) m_{f_j \to x_p}(x_p) \tag{2.3}$$

The function $m_{f_j \to x_p}(x_p)$ hence summarizes the contribution of the subtree $T_{f_j \to x_p}$ towards the marginalization of g and is often interpreted as the message sent from the factor node f_j to the variable node x_p.

We can apply formula 2.3 to any factor node f_j in a given factor tree and in particular we can apply this formula to factor nodes f_j that have at least one variable node among their descendants and for which the sets $\mathbb{X}_{jl}$ are all empty and hence the integration operators $\int_{\mathbb{X}_{jl}}$ in 2.2 vanish.

Definition 3. *Factor nodes f_j that have at least one variable node among their descendants and for which the sets $\mathbb{X}_{jl}$ as defined in 2 are all empty are said to be reducible.*

Theorem 2. *Under the assumption that there are more variable nodes than just the root node x_i left in a given factor tree, at least one factor node in that tree is reducible.*

Proof. We only consider finite length factor trees. So there exists a node n with

a depth greater than or equal to the depth of all other nodes. By assumption the factor tree contains two or more variable nodes and it follows that node n has a depth of at least 2, and so, on the unique path between the root node and node n there must be at least one factor node. The factor node that is closest to n on this path then fulfills the required properties. □

Also note that each time we apply formula 2.3 to a factor node f_j that has at least one variable node among its descendants and for which the sets $\mathbb{X}_{jl}$ are all empty we reduce the number of nodes in the factor tree. So recursive application of formula 2.3 to reducible factor nodes f_j readily suggests an efficient algorithm for the calculation of the marginal $g_i(x_i)$ that eventually terminates when the only variable node left is the root node.

We illustrate this procedure by an example. As in the previous example, we consider the global function

$$g(x_1, \ldots, x_5) = f_A(x_1) f_B(x_2) f_C(x_1, x_2, x_3) f_D(x_3, x_4) f_E(x_3, x_5)$$

The corresponding factor graph is a tree and taking x_1 as the root yields the factor tree shown in Figure 2.3. Summarizing the contribution of the subtrees $T_{f_D \to x_3}$ and $T_{f_E \to x_3}$ yields

$$m_{f_D \to x_3}(x_3) = \int_{x_4} f_D(x_3, x_4)$$

and

$$m_{f_E \to x_3}(x_3) = \int_{x_5} f_E(x_3, x_5)$$

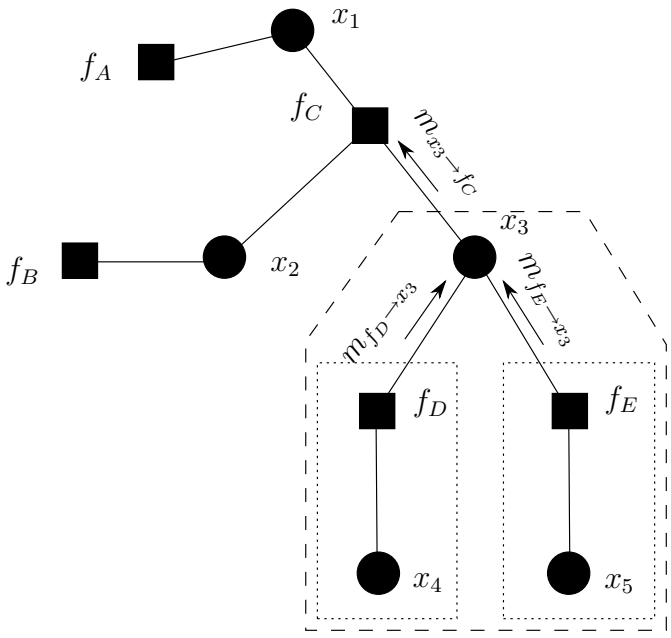

Figure 2.3: An example factor graph

respectively. For the subtree $T_{f_C \to x_1}$ we obtain

$$m_{f_C \to x_1}(x_1) = \int_{x_2, x_3} f_C(x_1, x_2, x_3) m_{f_D \to x_3}(x_3) m_{f_E \to x_3}(x_3) f_B(x_2)$$

The product $m_{f_C \to x_1}(x_1) f_A(x_1)$ finally represents the marginal $g_1(x_1)$.

It is easily recognized that applying the above formula 2.3 on a tree T_{x_i} as a "bottom up" procedure is equivalent to applying the following two rules "bottom up":

$$m_{x \to f}(x) = \prod_{g \in n(x) \setminus \{f\}} m_{g \to x}(x) \tag{2.4}$$

$$m_{f \to x}(x) = \int_{\sim \{x\}} \left(f(X) \prod_{u \in n(f) \setminus \{x\}} m_{u \to f}(u) \right) \tag{2.5}$$

where $n(v)$ denotes the set of nodes that neighbor a given node v in a factor

graph, i.e., those connected to node v by a single edge. The function $m_{x\to f}(x)$ summarizes the contribution the subtree $T_{x\to f}$ makes towards the marginalization of g and is often interpreted as the message sent from the variable node x to the factor node f.

2.2 Wirtinger Calculus

Many problems in communications and signal processing center on the optimization of a real-valued objective function over one or more complex variables. The popular numerical optimization method of steepest ascent relies on the partial derivatives of the the objective function, but it is not immediately clear how one would use this method if the objective function is a real-valued function of complex variables. In particular, non-constant real-valued functions of complex variables are not differentiable in the standard complex-variables sense.

Any complex number is uniquely described by an element of $\mathbb{R}^2$. For that reason we can view any real-valued objective function of n complex variables equivalently as a real-valued function of $2n$ real components. In case the partial derivatives with respect to these real-valued components exist, we would then be in a position to employ a gradient composed by those partial derivatives.

Because the real gradient perspective arises within a complex variables framework, a direct reformulation of the problem to the real domain is awkward. Instead, it greatly simplifies derivations if one introduces a redefined partial derivative operator and remains in the complex domain.

Notation-wise, a *real function* refers to a real-valued function of a real variables and a *complex function* refers to a complex or real-valued function of complex variables. As mentioned above, for a real-valued function of a complex variable we can easily shift back and forth between the real function $\mathbb{R}^2 \to \mathbb{R}$

perspective and the complex function $\mathbb{C} \to \mathbb{C}$ perspective.

2.2.1 Complex differentiability and holomorphic functions

We start with the definition of complex differentiability:

Definition 4. *Let $\mathbb{A} \subset \mathbb{C}$ be an open set. The function $f : \mathbb{A} \to \mathbb{C}$ is said to be (complex) differentiable at $z_0 \in \mathbb{A}$ if the limit*

$$\lim_{h \to 0} \frac{f(z_0 + h) - f(z_0)}{h}$$

exists independent of the manner in which $h \to 0$, where $h \in \mathbb{C}$. This limit is then denoted by $f'(z_0) = \left.\frac{df(z)}{dz}\right|_{z=z_0}$ and is called the derivative of f with respect to z at the point z_0.

If the function f is differentiable in z_0 for all z_0 in some open subset $\mathbb{U}$ of $\mathbb{C}$, it is called *analytic* in this subset. Interestingly, it can be shown [10] that if a function f is analytic in $\mathbb{U}$, then its derivative f' is analytic in $\mathbb{U}$ as well. Drawing on that theorem it readily follows by induction that if a function f is analytic in $\mathbb{U}$, it is arbitrarily often differentiable in $\mathbb{U}$. Note that in general this is not true for real functions, as the derivative of a real function can exist but still be non-differentiable itself. The basic properties for the derivative of a sum, product, and composition of two functions known from real-valued analysis, however, remain valid in the complex domain [11].

It is well known [12] that a necessary condition for f being analytic in $\mathbb{U}$ is that the following system of partial differential equations, termed Cauchy-

Riemann equations, holds for every $z = x + \mathrm{i}y \in \mathbb{U}$:

$$\frac{\partial U(x,y)}{\partial x} = \frac{\partial V(x,y)}{\partial y} \tag{2.6}$$

$$\frac{\partial U(x,y)}{\partial y} = -\frac{\partial V(x,y)}{\partial x} \tag{2.7}$$

where we decomposed $f(z)$ and z into their real and imaginary parts

$$\begin{aligned} z &= x + \mathrm{i}y \\ f(z) = F(x,y) &= U(x,y) + \mathrm{i}V(x,y) \end{aligned}$$

such that x, y, $U(x,y)$, and $V(x,y)$ are now real valued. If the partial derivatives of $U(x,y)$ and $V(x,y)$ with respect to x and y are continuous, the Cauchy-Riemann equations are also sufficient for $f(z)$ being analytic [10]. The only real-valued functions of complex variables that are analytic are constant, as for non-constant real-valued functions of complex variables the Cauchy-Riemann equations 2.6 and 2.7 are not fulfilled.

2.2.2 Differentials of analytic and non-analytic functions

The total differential of the bivariate function $F(x,y)$ reads as

$$dF = \frac{\partial F(x,y)}{\partial x}dx + \frac{\partial F(x,y)}{\partial y}dy \tag{2.8}$$

For the above partial derivatives to exist, we assumed that the real-valued functions $U(x,y)$ and $V(x,y)$ are differentiable. By defining the differentials

$$\begin{aligned} dz &= dx + \mathrm{i}dy \\ dz^\star &= dx - \mathrm{i}dy \end{aligned}$$

we can write dF as

$$\begin{aligned} dF =& \frac{1}{2}\left[\frac{\partial U(x,y)}{\partial x} + \frac{\partial V(x,y)}{\partial y} + \mathrm{i}\left(\frac{\partial V(x,y)}{\partial x} - \frac{\partial U(x,y)}{\partial y}\right)\right] dz \\ &+ \frac{1}{2}\left[\frac{\partial U(x,y)}{\partial x} - \frac{\partial V(x,y)}{\partial y} + \mathrm{i}\left(\frac{\partial V(x,y)}{\partial x} + \frac{\partial U(x,y)}{\partial y}\right)\right] dz^\star \end{aligned} \tag{2.9}$$

It is easily seen that the factor in front of $dz^\star$ is zero if and only if the Cauchy-Riemann equations hold. Equivalently we can say that the function $f(z)$ is analytical if and only if dF does not depend on $dz^\star$.

Rearranging the terms in equation 2.9 again, we obtain

$$dF = \frac{1}{2}\left[\frac{\partial}{\partial x} - \mathrm{i}\frac{\partial}{\partial y}\right] F(x,y)dz + \frac{1}{2}\left[\frac{\partial}{\partial x} + \mathrm{i}\frac{\partial}{\partial y}\right] F(x,y)dz^\star \tag{2.10}$$

and introducing the partial derivative operators $\frac{\partial}{\partial z}$ and $\frac{\partial}{\partial z^\star}$

$$\begin{aligned} \frac{\partial}{\partial z} =& \frac{1}{2}\left[\frac{\partial}{\partial x} - \mathrm{i}\frac{\partial}{\partial y}\right] \\ \frac{\partial}{\partial z^\star} =& \frac{1}{2}\left[\frac{\partial}{\partial x} + \mathrm{i}\frac{\partial}{\partial y}\right] \end{aligned}$$

leads to a nice description of the differential df, where the real-valued partial derivatives are hidden.

Theorem 3. *The differential df of a complex-valued function $f(z) : \mathbb{A} \to \mathbb{C}$ with $\mathbb{A} \subset \mathbb{C}$ can be expressed as*

$$df = \frac{\partial f(z)}{\partial z}dz + \frac{\partial f(z)}{\partial z^\star}dz^\star. \tag{2.11}$$

The partial derivatives $\frac{\partial}{\partial z}$ and $\frac{\partial}{\partial z^\star}$ are often referred to as the Wirtinger derivatives [13].

Assume that $f(z)$ and $g(z)$ are differentiable with respect to their real-valued

real and imaginary components. Then, the following propositions hold [14]:

Proposition 1. *The Wirtinger derivatives are linear operators*

$$\begin{aligned}\frac{\partial}{\partial z}(\alpha f(z)+\beta g(z)) =&\alpha\frac{\partial}{\partial z}f(z)+\beta\frac{\partial}{\partial z}g(z)\\ \frac{\partial}{\partial z^\star}(\alpha f(z)+\beta g(z)) =&\alpha\frac{\partial}{\partial z^\star}f(z)+\beta\frac{\partial}{\partial z^\star}g(z)\end{aligned}$$

Proposition 2. *The Wirtinger derivatives obey the product rule*

$$\begin{aligned}\frac{\partial}{\partial z}(f(z)g(z)) &= \left(\frac{\partial}{\partial z}f(z)\right)g(z)+\left(\frac{\partial}{\partial z}g(z)\right)f(z)\\ \frac{\partial}{\partial z^\star}(f(z)g(z)) &= \left(\frac{\partial}{\partial z^\star}f(z)\right)g(z)+\left(\frac{\partial}{\partial z^\star}g(z)\right)f(z)\end{aligned}$$

Proposition 3. *The following chain rule identities hold*

$$\begin{aligned}\frac{\partial f(g(z))}{\partial z} =&\frac{\partial f(g)}{\partial g}\frac{\partial g(z)}{\partial z}+\frac{\partial f(g)}{\partial g^\star}\frac{\partial g^\star(z)}{\partial z}\\ \frac{\partial f(g(z))}{\partial z^\star} =&\frac{\partial f(g)}{\partial g}\frac{\partial g(z)}{\partial z^\star}+\frac{\partial f(g)}{\partial g^\star}\frac{\partial g^\star(z)}{\partial z^\star}\end{aligned}$$

Proposition 4. *$z^\star$ can be regarded as a constant when differentiating with respect to z, and z can be regarded constant when differentiating with respect to $z^\star$*

$$\frac{\partial}{\partial z}z^\star=\frac{\partial}{\partial z^\star}z=0$$

Proposition 5. *Derivatives of the conjugate function $f^\star(z)$ satisfy the relation-*

ships

$$\frac{\partial f^\star(z)}{\partial z} = \left(\frac{\partial f(z)}{\partial z^\star}\right)^\star$$
$$\frac{\partial f^\star(z)}{\partial z^\star} = \left(\frac{\partial f(z)}{\partial z}\right)^\star$$

2.2.3 Optimization of functions of complex variables

Given a real function f, a necessary condition for an interior point x being a stationary point is that the gradient of f at this point vanishes. The same condition holds for every real-valued function f of complex variables if it is viewed as as a function of the real-valued real and imaginary components of the complex variables and the partial derivatives are taken with respect to these real components. A direct reformulation of the problem to the real domain is, however, usually quite awkward and it greatly simplifies derivation if we can stay in the complex domain and use methods based on the Wirtinger derivatives instead.

For a real-valued function f of a complex variable, it can easily be concluded from Theorem 3 that

$$df = 2\Re\left\{\frac{\partial f(z)}{\partial z}dz\right\} = 2\Re\left\{\frac{\partial f(z)}{\partial z^\star}dz^\star\right\} \tag{2.12}$$

which is equivalent to

$$dF = \frac{\partial F(x,y)}{\partial x}dx + \frac{\partial F(x,y)}{\partial y}dy \tag{2.13}$$

In order to maximize or minimize an objective function numerically, often gradient-based iterative algorithms like the gradient ascent or gradient descent algorithm, respectively, are applied. For real functions the gradient has

the property that it points to the direction of the steepest ascent. For any real-valued function of a complex variable we can shift to the real function $\mathbb{R}^2 \rightarrow \mathbb{R}$ perspective and also set up a gradient that has the interpretation of pointing towards the steepest ascent.

The following equation establishes a connection between the Wirtinger derivatives of f and the gradient of its real function equivalent.

$$df = 2\Re\left\{\frac{\partial f(z)}{\partial z^\star}dz^\star\right\} = 2\Re\left\{\frac{\partial f(z)}{\partial z^\star}\right\}dx + 2\Im\left\{\frac{\partial f(z)}{\partial z^\star}\right\}dy \tag{2.14}$$

It is easily recognized that $2\Re\left\{\frac{\partial f(z)}{\partial z^\star}\right\}$ and $2\Im\left\{\frac{\partial f(z)}{\partial z^\star}\right\}$ are the partial derivatives of f with respect to x and y, respectively, and thus the steepest ascent points to the direction of $2\frac{\partial f(z)}{\partial z^\star}$.

An iterative implementation of the gradient ascent algorithm could therefore read as

$$z \leftarrow z + 2\frac{\partial f(z)}{\partial z^\star}ds \tag{2.15}$$

where ds is a step-size parameter.

All the results above extend in a straightforward manner to the case where f is a function of several complex variables.

CHAPTER 3

Channel Modeling

3.1 Notation and System Definition

We consider the channel model as introduced in Chapter 1. For notational convenience we represent the channel taps in the matrix $\boldsymbol{H}$ by identifying $[\boldsymbol{H}]_{(i,j)}$ with $h[i,j]$ and also introduce the vectors $\boldsymbol{y}$ and $\boldsymbol{x}$ with $[\boldsymbol{y}]_{(i)} = y[i]$ and $[\boldsymbol{x}]_{(i)} = x[i]$. Due to the Gaussian assumption on the channel noise, the distribution of $\boldsymbol{y}$ given $\boldsymbol{x}$ and $\boldsymbol{H}$ then reads

$$p(\boldsymbol{y}|\boldsymbol{H},\boldsymbol{x}) \sim \prod_{i=0}^{M-1} \underbrace{\exp(-|y[i] - \sum_{j=0}^{L-1} h[i,j]x[i-j]|^2/\sigma_n^2)}_{=T_i(\boldsymbol{H})} \tag{3.1}$$

where the notation $\sim$ means that the left-hand side and the right-hand side are proportional to each other.

3.2 Markov Random Field Theory

We use an MRF to model the time-varying channel $\boldsymbol{H}$. By definition, an MRF is a set of random variables $\boldsymbol{H}$ with an associated undirected graph. The ran-

dom variables h in $\boldsymbol{H}$ are assumed to satisfy the following two conditions:

$$p(h) > 0 \tag{3.2}$$

$$p(h|\boldsymbol{H}_{\setminus h}) = p(h|k \in n(h)) \tag{3.3}$$

where $\boldsymbol{H}_{\setminus h}$ is the entire set of random variables $\boldsymbol{H}$ without the element h and $n(h)$ denotes the set of nodes k that neighbor h in the corresponding graph.[1] In other words, an MRF represents statistical dependencies of variables by an undirected graph. The vertices in an MRF stand for random variables and the edges impose statistical constraints on these random variables. A subset b of $\boldsymbol{H}$ is called a *clique* if it is a singleton or if every pair of elements h in b are neighbors in the corresponding graph.

The lattice shaped MRF considered in this thesis is depicted in Figure 3.1. Due to the geometric nature of lattices, we prefer to index the variables h by

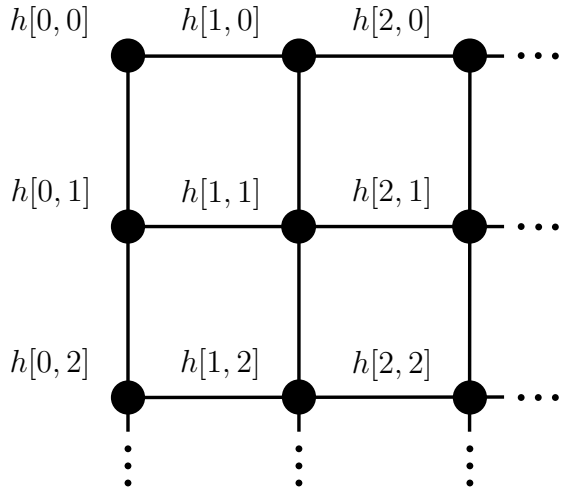

Figure 3.1: Lattice shaped Markov random field

the two dimensional index pair $[i, j]$. So here the set $\boldsymbol{H}$ represents the set of channel gains $h[i, j]$, and $\mathbf{N}[i, j]$ will denote the set $\{[l, m] : h[l, m]$ neighbors $h[i, j]\}$.

[1]Note that we use the same symbol to denote a random variable and its realization, whenever the meaning is clear from the context.

We easily identify a singleton clique for each vertex $h[i,j]$ and a pairwise clique for each pair of adjacent vertices. These pairwise cliques have the form $\{h[i,j], h[i,j-1]\}$ or $\{h[i,j], h[i-1,j]\}$. According to the Hammersley-Clifford theorem [15,16] $\boldsymbol{H}$ and its associated graph form an MRF if and only if the distribution on the variables in $\boldsymbol{H}$ is of the form

$$p(\boldsymbol{H}) = Z^{-1} \exp(-\frac{\sum_{b \in \mathrm{B}} V_b(\bar{b})}{K}) \tag{3.4}$$

This form of joint distribution is known as a Gibbs distribution. The parameter K is a temperature parameter chosen to be unity here, and Z is a normalization constant. The argument of the exponential function includes a sum of functions $V_b(\bar{b})$ which are indexed by the set B of all possible cliques b. The vector $\bar{b}$ is composed of the values of the random variables $h[i,j]$ in the clique b. The functions $V_b(\bar{b})$ are simply assumed to be nonnegative functions of their arguments and are called clique potentials.

3.3 Channel Modeling

We take the potential of pairwise cliques as the square of adjacent differences:

$$\begin{aligned} V_b(\bar{b}) = \alpha_{[i_b,j_b],[m_b,n_b]} | & (h[i_b,j_b] - \mu[j_b]) \\ & - (h[m_b,n_b] - \mu[n_b])|^2 \end{aligned} \tag{3.5}$$

where $[i_b,j_b]$ and $[m_b,n_b]$ are the coordinates of the vertices in the one pair clique b and $\alpha_{[i_b,j_b],[m_b,n_b]}$, $\mu[j_b]$ and $\mu[n_b]$ are parameters. The potentials of single cliques that are associated with channel gains $h[i,j]$ at time $i = 0$ form a Gaussian dis-

tribution

$$V_b(\bar{b}) = \alpha_{[j_b]}|h[i_b, j_b] - \mu[j_b]|^2 \tag{3.6}$$

with the parameters $\mu[j_b]$ and $\alpha_{[j_b]}$. The other potentials are assumed to be zero. So we have

$$p(\boldsymbol{H}|\boldsymbol{\theta}) = Z(\boldsymbol{\theta})^{-1} \exp\left(- \sum_{b \in \mathrm{B}_1} \alpha_{[j_b]}|h[i_b, j_b] - \mu[j_b]|^2 - \right.$$
$$\left. \sum_{b \in \mathrm{B}_2} \alpha_{[i_b,j_b],[m_b,n_b]}|(h[i_b, j_b] - \mu[j_b]) - (h[m_b, n_b] - \mu[n_b])|^2 \right)$$

where the vector $\boldsymbol{\theta}$ contains all the model parameters, i.e., the $\alpha_{[i_b,j_b],[m_b,n_b]}$, the $\alpha_{[j_b]}$ and the $\mu[j]$. The set B_1 comprises all single cliques that correspond to channel gains $h[i, j]$ at time $i = 0$ and the set B_2 contains all the pairwise cliques.

Figure 3.2 shows the factor graph representation of $p(\boldsymbol{H}|\boldsymbol{\theta})$. The black boxes

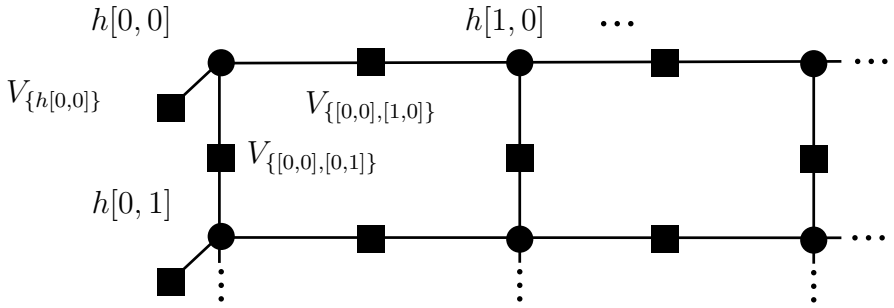

Figure 3.2: Factor graph based on MRF model

and circles represent function and variable nodes, respectively. Obviously, our MRF model can be interpreted as a sequence of vertically connected Markov chains and it accounts for the correlation between consecutive taps in delay by introducing function nodes between vertically neighboring variable nodes.

The reasons for the above choice of the potentials are as follows. First it

makes $\boldsymbol{H}$ jointly Gaussian distributed and $h[i,j]$ has mean $\mu[j]$. As we will see later in this thesis assuming a Gaussian distribution significantly simplifies the computation needed for the inference of the channel gains. Second, it imposes a certain continuity on the behavior of neighboring channel gains, as $p(h[i,j]|h[l,m] : [l,m] \in \mathbf{N}[i,j])$ for $i \neq 0$ then becomes a complex Gaussian distribution with mean

$$\mu[j] + \frac{\sum_{[l,m]\in\mathbf{N}[i,j]} \alpha_{[i,j],[l,m]}(h[l,m] - \mu[m])}{\sum_{[l,m]\in\mathbf{N}[i,j]} \alpha_{[i,j],[l,m]}} \tag{3.7}$$

and variance

$$\left(\sum_{[l,m]\in\mathbf{N}[i,j]} \alpha_{[i,j],[l,m]} \right)^{-1} \tag{3.8}$$

The mean of $h[i,j]$ is shifted by a weighted sum of the differences $h[l,m] - \mu[m]$. So if all the neighbors, for example, assumed values above their means, $h[i,j]$ is likely to assume a value that is above its mean as well. The $\alpha_{[i,j],[l,m]}$'s determine what impact each neighbor has on $h[i,j]$. For $i = 0$, $p(h[i,j]|h[l,m] : [l,m] \in \mathbf{N}[i,j])$ becomes a complex Gaussian distribution with mean

$$\mu[j] + \frac{\sum_{[l,m]\in\mathbf{N}[i,j]} \alpha_{[i,j],[l,m]}(h[l,m] - \mu[m])}{\alpha_{[j]} + \sum_{[l,m]\in\mathbf{N}[i,j]} \alpha_{[i,j],[l,m]}} \tag{3.9}$$

and variance

$$\left(\alpha_{[j]} + \sum_{[l,m]\in\mathbf{N}[i,j]} \alpha_{[i,j],[l,m]} \right)^{-1} \tag{3.10}$$

The $\alpha_{[j]}$ hence determines how likely it is that our concatenated Markov chains start with the mean $\mu[j]$.

3.4 MRF-Based Channel Estimation

3.4.1 Problem setup

The channel estimation task is based on the following MAP estimation approach:

$$\begin{aligned}\hat{\boldsymbol{H}} &= \underset{\boldsymbol{H}}{\operatorname{argmax}}\, p(\boldsymbol{H}|\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}) \\ &= \underset{\boldsymbol{H}}{\operatorname{argmax}}\, p(\boldsymbol{H}, \boldsymbol{y}, \boldsymbol{x}|\boldsymbol{\theta}) \\ &= \underset{\boldsymbol{H}}{\operatorname{argmax}}\, p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x}) p(\boldsymbol{H}|\boldsymbol{\theta})\end{aligned} \tag{3.11}$$

For now, we assume the values of the parameters $\boldsymbol{\theta}$ to be known.

3.4.2 Solution

The probabilities $p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x})$ and $p(\boldsymbol{H}|\boldsymbol{\theta})$ are both proportional to Gaussian densities in $\boldsymbol{H}$ and hence $p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x})p(\boldsymbol{H})$ can be considered as proportional to a Gaussian density in $\boldsymbol{H}$ as well. Gaussian densities assume their maximum at their mean and therefore the optimization problem above amounts to calculating the means of $\boldsymbol{H}$ with respect to $p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x})p(\boldsymbol{H})$. As $p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x})p(\boldsymbol{H})$ factors nicely, the sum-product algorithm can be used for an efficient marginalization. The means of the marginals with respect to $p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x})p(\boldsymbol{H})$ are easily identified as the means of $\boldsymbol{H}$ with respect to $p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x})p(\boldsymbol{H})$. Figure 3.3 shows the factor graph representation of $p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{x})p(\boldsymbol{H})$. Because of the Gausianity of the distributions involved, the marginalized functions are completely characterized by their mean and variance. Our factor graph has many cycles and so we need to iterate the message passing until it converges [9].

Following the notation of [9] we denote messages sent from a variable node

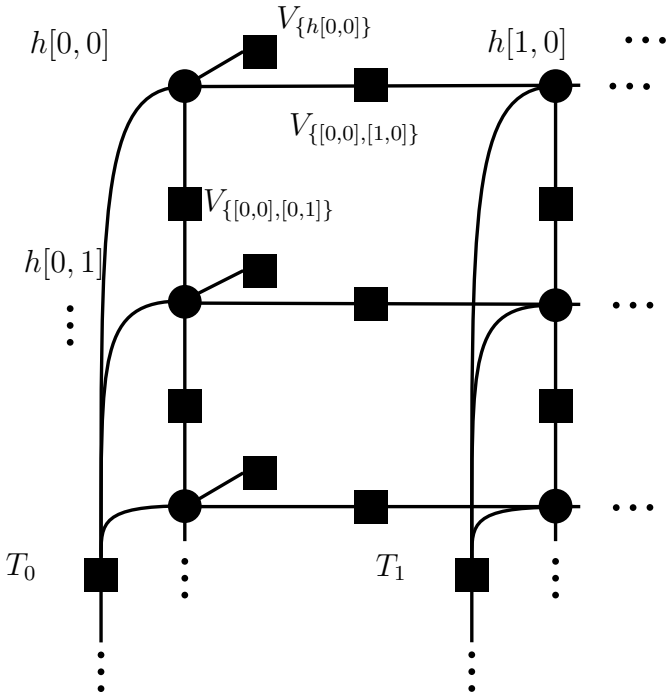

Figure 3.3: Factor graph for MAP channel estimation

$h[i,j] \in \boldsymbol{H}$ to a local function node as $m_{h\to f}(h)$. Here f represents either one of the functions T_i or one of the potential functions $V_b(\bar{b})$. Furthermore we denote messages sent from a local function node to a variable node as $m_{f\to h}(h)$. Also, let $n(v)$ denote the set of nodes that neighbor a given node v in a factor graph.

Before we derive more specific message computation rules, let us focus again on the factor graph shown in Figure 3.3. We see that the messages that are sent along the edges of our factor graph are of three kinds. Messages of the first kind come from a variable node, messages of the second kind come from one of the potential functions and messages of the third kind come from one of the functions T_i. These three types of messages are illustrated in Figure 3.4. The type of the message is superscribed in each case. As mentioned above, the messages are completely characterized by their mean μ_m and variance σ_m^2. The derivation of update rules for messages of the first two kinds is straightforward; the derivation of an update rule for messages of the third kind is more involved (see

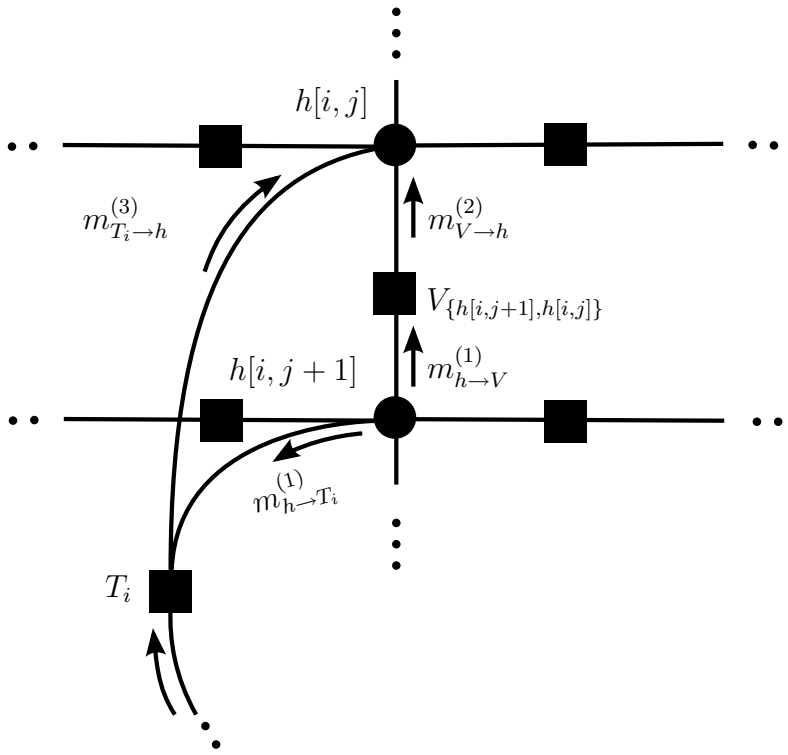

Figure 3.4: A closer look at the factor graph from Figure 3.3

Appendix A for the derivation). We just summarize the results here.

Update rule for messages of the first kind

$$\begin{aligned} \mu_{m_{h\to f}} =& \frac{\sum_{g\in n(h)\setminus\{f\}} \sigma_g^{-2}\mu_g}{\sum_{g\in n(h)\setminus\{f\}} \sigma_g^{-2}} \\ \sigma^{-2}_{m_{h\to f}} =& \sum_{g\in n(h)\setminus\{f\}} \sigma_g^{-2} \end{aligned} \tag{3.12}$$

where μ_g and σ_g^2 are the mean and the variance of the message $m_{g\to h}(h)$, respectively.

Update rule for messages of the second kind

$$\begin{aligned} \mu_{m_{V_b\to h}} =& \mu_u + \mu[j_h] - \mu[j_u] \\ \sigma^{-2}_{m_{V_b\to h}} =& \frac{\sigma_u^{-2}\alpha_{[i_h,j_h],[i_u,j_u]}}{\sigma_u^{-2} + \alpha_{[i_h,j_h],[i_u,j_u]}} \end{aligned} \tag{3.13}$$

where μ_u and σ_u^2 are the mean and the variance of the message $m_{u \to V_b}(u), \quad u \in n(V_b) \backslash \{h\}$, respectively, and $[i_h, j_h]$ and $[i_u, j_u]$ are the coordinates of the nodes h and u, respectively.

Update rule for messages of the third kind

$$\begin{aligned} \sigma^{-2}_{m_{T_i \to h[i,j]}} =& |x[i-j]|^2 \sigma_n^{-2} (1+z)^{-1} \\ z =& \sigma_n^{-2} \sum_{l=0, l \neq j}^{L-1} |x[i-l]|^2 \sigma^2_{m_{h[i,l] \to T_i}} \\ \mu_{m_{T_i \to h[i,j]}} =& x[i-j]^{-1} (y[i] - \sum_{l=0, l \neq j}^{L-1} x[i-l] \mu_{m_{h[i,l] \to T_i}}) \end{aligned} \tag{3.14}$$

The sum-product algorithm is guaranteed to converge to the correct marginals in singly connected graphs [9]. In a factor graph of arbitrary topology the marginals are in general not guaranteed to converge and even if they converge they might be incorrect. In this paper we consider the special case, where the graph features cycles and its nodes describe jointly Gaussian random variables, and in this case it can be shown that the means of the marginals are correct given that the algorithm converges [17]. As we are interested in finding the means of the marginals only, our algorithm yields a correct MAP estimate, given it converges.

3.5 Simulation Results

In order to evaluate the potential of the proposed channel model, we generate a channel (i.e., we draw the channel gains from the probability distribution that corresponds to our MRF model). In an attempt to mimic a real communications channel where the correlation in time is a lot stronger than that in delay,

the parameters $\alpha_{[i,j],[l,m]}$ between vertically adjacent gains were set to 100, while the $\alpha_{[i,j],[l,m]}$s between horizontally adjacent gains were set to 1000. The $\alpha_{[j]}$s were all set to 100 and the means were chosen to be

$$[m[j]] = \Big[\begin{array}{cccccccccc} 0 & 0.6 & 0.4 & 1 & 0.2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & .2 & .1 \end{array} \Big]$$

A typical channel realization for these parameters is depicted in Figure 3.5

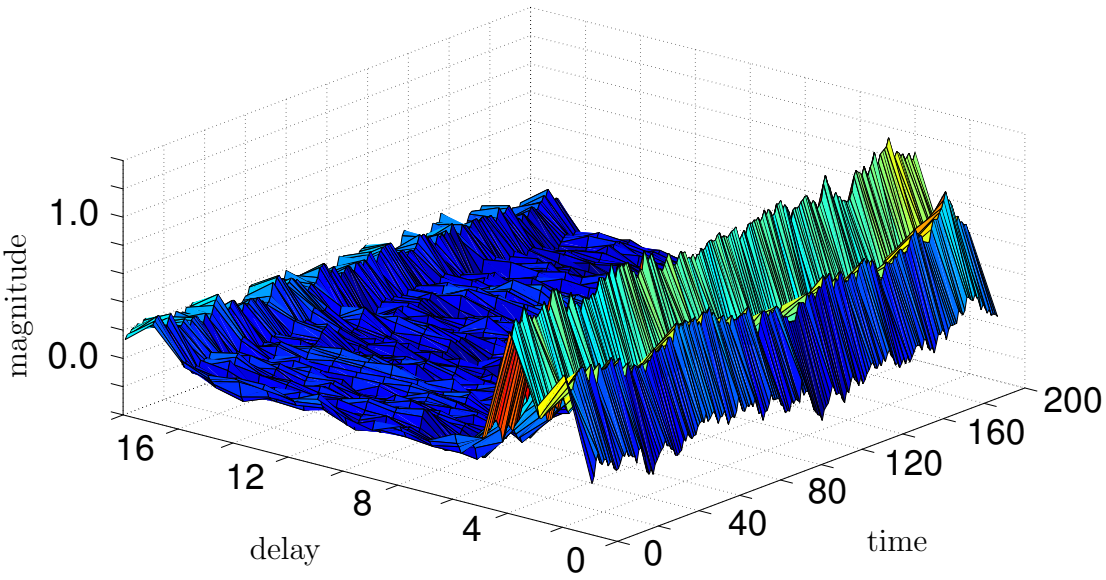

Figure 3.5: Channel realization used for simulations

We then applied three different channel estimators to the given observations. The first is an RLS algorithm parameterized with a forgetting factor of 0.999 to account for the strong correlation in time. The second is based on our MRF model but we set all the $\alpha_{[i,j],[l,m]}$s between vertically adjacent gains to zero in the estimator. This way the channel estimator neglects correlations of the channel gains in delay and channel estimation is equivalent to Kalman filtering based on a diagonal state transition matrix as proposed in [2]. The third method we evaluate is also based on our MRF model but this time the method

takes correlations over delay into account. The simulation results for these three methods are shown in Figure 3.6. Here NMSE denotes the normalized mean

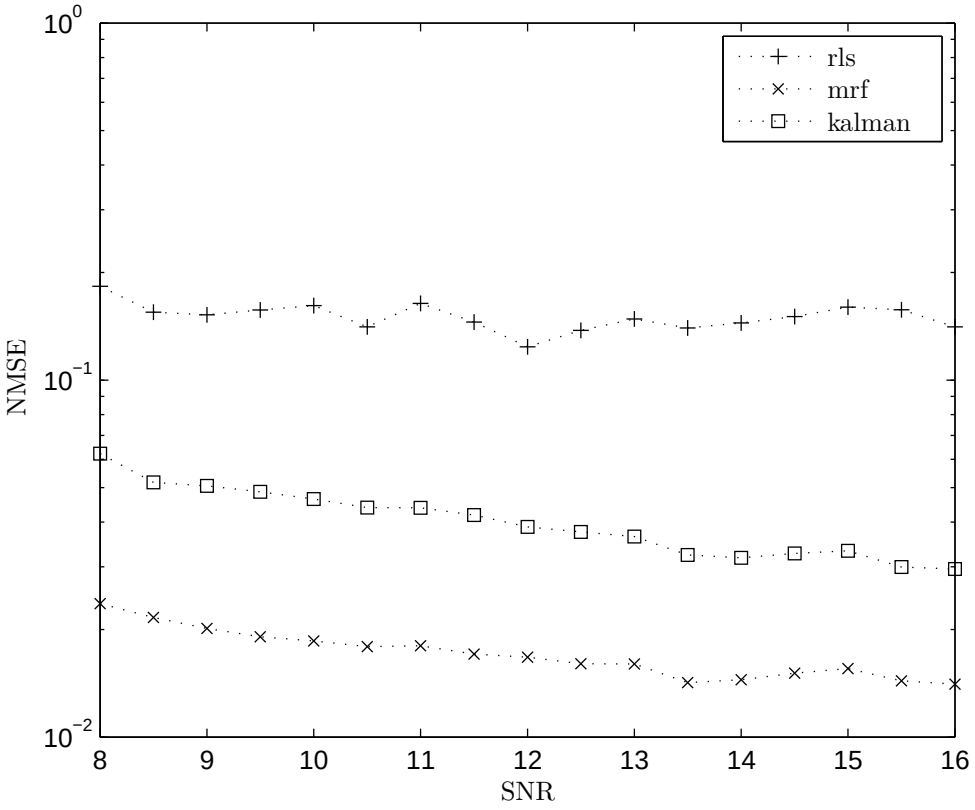

Figure 3.6: Channel estimation performance of our MRF model-based MAP estimation versus a conventional RLS method

squared error defined as

$$\mathrm{NMSE} = \underset{i}{\mathrm{avg}} \left[\frac{\| [\boldsymbol{H}]_{(i,:)} - [\hat{\boldsymbol{H}}]_{(i,:)} \|^2}{\| [\boldsymbol{H}]_{(i,:)} \|^2} \right] \tag{3.15}$$

where avg_i means time average. It is easily seen that taking the correlation of the channel gains in delay into account benefits the channel estimation quality significantly. In case $\alpha_{[i,j],[l,m]}$s between vertically adjacent gains are set to smaller values for the generation of the synthetic channel, the proposed method loses part of its performance advantage, but remains superior to the conven-

tional RLS method.

3.6 Conclusion

So far we have assumed the parameters $\boldsymbol{\theta}$ to be known. In a real communication setup this is, however, not the case. There are two ways to approach this problem. Either we guess some reasonable values a priori and accept the performance loss that comes with the introduced model mismatch, or we do actually estimate the unknown parameters from available channel observations.

In simulations we experienced that the estimation performance degraded gracefully when moving the parameters used in MRF channel estimation from their real values. Similar observations were made in [18]. So the MRF setup described in this paper achieves competitive simulation results for channel estimation as long as the MRF parameters were set in the right range.

Simulations on synthetic data, however, also showed that the estimation quality improves if the MRF model parameters match the actual channel accurately. To estimate the parameters from real channel observations, we used an *expectation maximization* (EM) like algorithm. The precise calculation of the integrals involved in the conduction of EM is not computationally feasible [19] and so we bypassed this issue by approximating the maximization step in EM by a gradient ascent step and obtained an EM-based parameter estimator.

CHAPTER 4

Parameter Estimation

4.1 Introduction

Many problems in signal processing can be cast into the framework of state estimation, in which we have state variables $h[i]$ whose values are not directly accessible and variables $y[i]$ whose values are available. Variables of the latter kind are also referred to as observations in this context. Usually there exists a statistical relationship $p(\boldsymbol{y}|\boldsymbol{h})$ between the state variables $h[i]$ and the observations $y[i]$ such that we can infer estimates $\hat{h}[i]$ of the states from the observations. In many cases prior knowledge about the states is also available (usually in form of a probability distribution $p(\boldsymbol{h})$ on the state variables) and we can use that knowledge to refine the state estimate.

In a variety of interesting problems, however, neither the statistical relationship between the state variables and the observations nor the prior distribution are perfectly known and hence are modeled as parameterized distributions $p(\boldsymbol{y}|\boldsymbol{h}, \theta)$ and $p(\boldsymbol{h}|\theta)$ with unknown parameters θ. These parameters are then also subject to estimation.

Here we restrict the prior distribution on the hidden state variables to the form of a parametrized Gaussian Markov random field and assume a simple parametrized linear observation model. We shall propose an efficient algorithm to estimate the unknown parameters. Our algorithm can be interpreted as an approximation to the well known expectation maximization (EM) algorithm.

4.2 Parameter Estimation

In the previous chapter we assumed the parameters $\boldsymbol{\theta}$ to be known. This section is dedicated to the design of an algorithm that is capable of estimating theses parameters from the channel output $\boldsymbol{y}$.

4.2.1 Problem setup

We consider an incomplete data problem where some of the variables are hidden and others are observable. The hidden variables are assumed to be modeled by the MRF proposed in Section 3.2 and 3.3. And the observations $y[i]$ and the hidden variables $\boldsymbol{H}$ are assumed to have the statistical relation introduced in Section 3.1, where $\boldsymbol{\theta}$ now also comprises the noise variance σ_n^2.

Our goal is to estimate the parameters $\boldsymbol{\theta}$ and we do so in a maximum likelihood (ML) fashion.

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\boldsymbol{y}|\boldsymbol{\theta}) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \int_{\boldsymbol{H}} p(\boldsymbol{y}, \boldsymbol{H}|\boldsymbol{\theta}) \tag{4.1}$$

where it can easily be checked that

$$p(\boldsymbol{y}, \boldsymbol{H}|\boldsymbol{\theta}) = p(\boldsymbol{y}|\boldsymbol{H}, \boldsymbol{\theta}) p(\boldsymbol{H}|\boldsymbol{\theta}) \tag{4.2}$$

The contribution of this section is the development of an efficient algorithm for the estimation of these parameters.

Numerical evaluation of maximum-likelihood estimates is often difficult. As a remedy we will use a powerful optimization method that has been used with great success in many applications: The expectation maximization (EM) algorithm [20]. A short review of this algorithm is in order:

1. Make some initial guess $\hat{\boldsymbol{\theta}}^{(0)}$

2. Expectation step: calculate

$$Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) = \langle \log p(\boldsymbol{y}, \boldsymbol{H}|\boldsymbol{\theta}) \rangle_{p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})} \tag{4.3}$$

   where $\langle \cdot \rangle_p$ represents expectation with respect to p.

3. Maximization step: compute

$$\hat{\boldsymbol{\theta}}^{(k+1)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) \tag{4.4}$$

4. Repeat 4.3-4.4 until convergence or until the available time is over.

Under rather general conditions this algorithm is proven to yield a nondecreasing sequence $p(\boldsymbol{y}|\hat{\boldsymbol{\theta}}^{(k)})$. However, it is well known that due to the interaction between the $h[i, j]$, the precise calculation of the partition function $Z(\boldsymbol{\theta})$ and the integral in (4.3) is not computationally tractable [19].

4.2.2 Solution

One can bypass the requirement of exactly knowing the partition function $Z(\boldsymbol{\theta})$ by approximating the maximization step above by a gradient ascent step. It should be noted, however, that taking all parameters in $\boldsymbol{\theta}$ as independent and distinct parameters would seriously overparameterize our model, since there would be more parameters than available observations. To tackle this problem we assume that all $\alpha_{[i,j],[l,m]}$s that correspond to a vertical pairwise clique are the same and equal α_v and similarly that all $\alpha_{[i,j],[l,m]}$s that correspond to a horizontal pairwise clique are the same and equal α_h. Also we take $\alpha_{[j_b]} = \alpha$.

As mentioned above we substitute the maximization step in the EM algorithm with a gradient ascent step. So let us proceed with the calculation of the gradient of $Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)})$ with respect to $\boldsymbol{\theta}$.

$$\begin{aligned}\frac{\partial}{\partial \sigma_n^2} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) = &- \frac{M}{\sigma_n^2} - \sum_{i=0}^{M-1} \langle |y[i] \\ &- \sum_{j=0}^{L-1} h[i,j] x[i,j]|^2 \rangle_{p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})} \end{aligned} \tag{4.5}$$

And the partial derivatives with respect to the MRF parameters θ_j have the following form:

$$\begin{aligned}\frac{\partial}{\partial \theta_j} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) = &\langle \frac{\partial}{\partial \theta_j} \sum_{b \in \mathrm{B}} V_b(\bar{b}) \rangle_{p(\boldsymbol{H}|\boldsymbol{\theta})} \\ &- \langle \frac{\partial}{\partial \theta_j} \sum_{b \in \mathrm{B}} V_b(\bar{b}) \rangle_{p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})} \end{aligned} \tag{4.6}$$

where

$$\begin{aligned}2 \frac{\partial}{\partial \mu[j]^\star} \sum_{b \in \mathrm{B}} V_b(\bar{b}) = &2 \Big(\alpha_v \Big(\sum_i h[i, j-1] - M\mu[j-1] \Big) \\ &+ \alpha_v \Big(\sum_i h[i, j+1] - M\mu[j+1] \Big) \\ &- 2\alpha_v \Big(\sum_i h[i,j] - M\mu[j] \Big) \\ &- \alpha \Big(h[1,j] - \mu[j] \Big) \Big) \end{aligned} \tag{4.7}$$

$$\frac{\partial}{\partial \alpha} \sum_{b \in \mathrm{B}} V_b(\bar{b}) = \sum_j |h[1,j] - \mu[j]|^2 \tag{4.8}$$

$$\frac{\partial}{\partial \alpha_v} \sum_{b \in \mathrm{B}} V_b(\bar{b}) = \sum_{i,j} |(h[i,j] - \mu[j]) - (h[i,j+1] - \mu[j+1])|^2 \tag{4.9}$$

$$\frac{\partial}{\partial \alpha_h} \sum_{b \in \mathrm{B}} V_b(\bar{b}) = \sum_{i,j} |(h[i,j] - (h[i+1,j])|^2 \tag{4.10}$$

and $\frac{\partial}{\partial \mu[j]^\star}$ denotes the Wirtinger derivative with respect to $\mu[j]^\star$. Note that the partial derivatives of $\sum_{b \in \mathrm{B}} V_b(\bar{b})$ with respect to the real and imaginary component of $\mu[j]$ coincide with the real and imaginary component of $2\frac{\partial}{\partial \mu[j]^\star} \sum_{b \in \mathrm{B}} V_b(\bar{b})$, respectively. Clearly in order to actually calculate the gradient at $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}^{(k)}$ we first need to know the moments $\langle h[i,j] \rangle$, $\mathrm{Cov}(h[i,j])$, $\mathrm{Cov}([h[i,j], h[i,j+1]])$ and $\mathrm{Cov}([h[i,j], h[i+1,j]])$ with respect to $p(\boldsymbol{H}|\hat{\boldsymbol{\theta}}^{(k)})$ and also with respect to $p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})$. This can be achieved by use of the sum-product algorithm [9]. Note that the random variables in $\boldsymbol{H}$ are jointly Gaussian and hence the messages that the sum-product algorithm passes along the edges of a factor graph are Gaussian distributions as well. Gaussian messages are parameterized by a mean vector and a covariance matrix and so the required moments are readily computed by the operation of the sum-product algorithm. The factor graph on which the sum-product algorithm operates for the calculation of the moments with respect to $p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})$ is shown in Figure 3.3.

The convenient factorization of $p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})$ that the MRF framework provides reduces the complexity of the marginalization tremendously. The factor graph corresponding to $p(\boldsymbol{H}|\hat{\boldsymbol{\theta}}^{(k)})$ coincides with the one in Figure 3.3 when the functions T_i are eliminated. The remainder of this section is dedicated to the implementation of the sum-product algorithm for the calculation of the moments. The factor graph corresponding to $p(\boldsymbol{H}|\hat{\boldsymbol{\theta}}^{(k)})$ is contained in the factor

graph corresponding to $p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})$ and hence the message passing for $p(\boldsymbol{H}|\hat{\boldsymbol{\theta}}^{(k)})$ is just a special case of the one for $p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})$. For that reason it suffices to derive the message passing rules for the factor graph associated with $p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})$.

The message passing rules required for calculation of the moments $\langle h[i,j] \rangle$ and $\mathrm{Cov}(h[i,j])$ are equivalent to the ones in the previous chapter. In order to obtain the moments $\mathrm{Cov}([h[i,j], h[i,j+1]])$ and $\mathrm{Cov}([h[i,j], h[i+1,j]])$ as well, we need to modify our current factor graph setup slightly.

A factor graph contains nodes of two types: function nodes and variable nodes. It is always possible to cluster nodes of the same type without changing the global function being represented by a factor graph. In our case we are only interested in the marginal distribution of a pair of variable nodes and hence only consider clusters of two nodes, but the concept of clustering variable nodes is easily generalized to larger clusters. Assume x and y are two variable nodes in some factor graph and we want to calculate their joint distribution. Then we could just combine these two variables to a new variable representing the cluster (x, y), change the factor graph accordingly and marginalize for the new node (x, y). To be more precise, if x and y are two nodes to be combined, simply remove these two nodes and the edges connected to them, introduce the new node (x, y) and reconnect nodes, that were connected with x or y before, with the new node (x, y). Note that functions that had x or y as an argument in the original factor graph are now functions of (x, y). From elementary factor graph theory, we know that the marginal distribution of (x, y) is then obtained as the product of all messages received at (x, y).

Now, in order to obtain the moments $\mathrm{Cov}([h[i,j], h[i,j+1]])$ and $\mathrm{Cov}([h[i,j], h[i+1,j]])$, we cluster the corresponding pairs of nodes $h[i,j]$. Figure 4.1 shows what effect the clustering of $(h[i,j], h[i,j+1])$ has on the factor graph in Figure 3.4. The update rule for messages of the third kind as listed

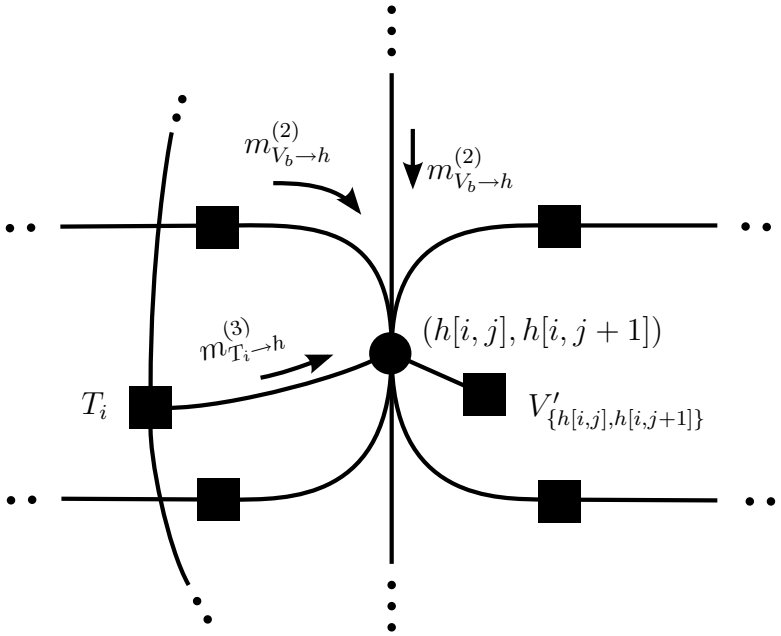

Figure 4.1: Factor graph

above must then be modified as follows.

Update rule for messages of the third kind

$$\begin{aligned}
\boldsymbol{\Sigma}^{-1}_{m_{T_i \to (h[i,j],h[i,j+1])}} &= \begin{bmatrix} x[i-j] \\ x[i-j-1] \end{bmatrix}^{\star} \begin{bmatrix} x[i-j] \\ x[i-j-1] \end{bmatrix}^{\mathrm{T}} \\
&\quad \cdot \sigma_n^{-2}(1+z)^{-1} \qquad (4.11) \\
z &= \sigma_n^{-2} \sum_{l=0, l \neq j, j+1}^{L-1} |x[i-l]|^2 \sigma^2_{m_{h[i,l] \to T_i}} \\
\boldsymbol{\mu}_{m_{T_i \to (h[i,j],h[i,j+1])}} &= \begin{bmatrix} 1 \\ 0 \end{bmatrix} x[i-j]^{-1} \\
&\quad \cdot (y[i] - \sum_{l=0, l \neq j, j+1}^{L-1} x[i-l] \mu_{m_{h[i,l] \to T_i}}) \qquad (4.12)
\end{aligned}$$

4.2.3 Analysis

As mentioned above it is well known that the EM algorithm yields a nondecreasing sequence of likelihoods $p(\boldsymbol{y}|\hat{\boldsymbol{\theta}}^{(k)})$. This property remains true even if the maximization step is replaced by a gradient ascent step as proposed in this paper. A proof of this result can be found in Appendix B. Note, however, that the above algorithm only approximates this gradient. As our factor graph does contain cycles, the sum-product algorithm only approximately calculates the moments required for setting up the gradient [17].

4.3 Simulation Results

We chose to evaluate the performance of the proposed estimator on synthetic data as this enables us to compare the obtained parameter estimate against the actual value of the parameter. So we drew realizations of the state variables from the probability distribution $p(\boldsymbol{H}|\theta)$, observed some noisy observations $\boldsymbol{y}$ and finally employed the proposed algorithm on these observations to obtain an estimate $\hat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$. For our simulation we parameterized the MRF model by setting the parameters α, α_v, α_h and $\boldsymbol{\mu}$ to 100, 100, 1000 and $[0\,0.6\,0.4\,1\,0.2]$, respectively. Figure 4.2 shows how the absolute estimation error of one of the parameters approaches zero over the iterations. The convergence speed of the gradient ascent algorithm depends on the step size and so does the algorithm presented here.

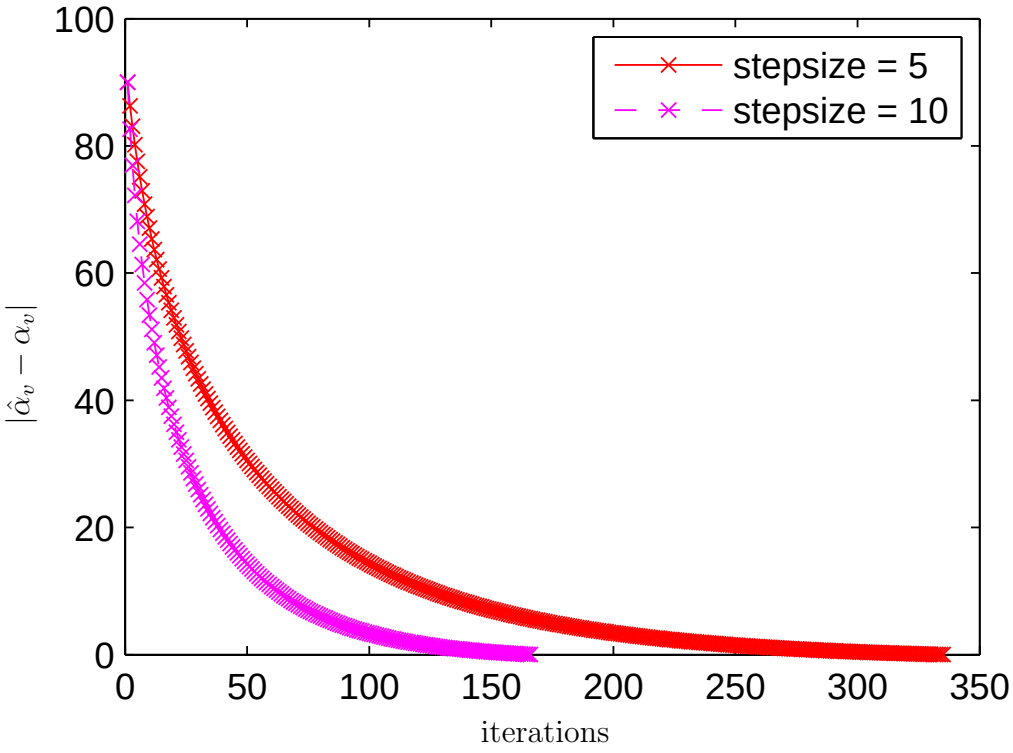

Figure 4.2: The absolute estimation error of α_v over iterations

APPENDIX A

Derivation of the Third Update Rule

For a more readable derivation of the third update rule we will denote $\mu_{m_{h[i,u]\to T_i}}$ and $\sigma^{-2}_{m_{h[i,u]\to T_i}}$ by μ_u and σ_u^{-2}, respectively.

First we rewrite $T_i(\boldsymbol{H})$ as

$$\begin{aligned} T_i &= \exp(-|y[i] - \sum_{l=0}^{L-1} h[i,l]x[i-l]|^2/\sigma_n^2) \\ &= \exp\left(-\sigma_n^{-2} \begin{pmatrix} y[i] \\ \boldsymbol{h} \end{pmatrix}^{\mathrm{H}} \begin{pmatrix} 1 \\ -\boldsymbol{x}^\star \end{pmatrix} \begin{pmatrix} 1 \\ -\boldsymbol{x}^\star \end{pmatrix}^{\mathrm{H}} \begin{pmatrix} y[i] \\ \boldsymbol{h} \end{pmatrix}\right) \end{aligned} \tag{A.1}$$

where $\boldsymbol{x}^{\mathrm{T}} = \begin{pmatrix} x[i], & \ldots, & x[i-L+1] \end{pmatrix}$ and $\boldsymbol{h}^{\mathrm{T}} = \begin{pmatrix} h[i,0], & \ldots, & h[i,L-1] \end{pmatrix}$ and define

$$\begin{aligned} f_i(\boldsymbol{h}) =& \sigma_n^{-2} \begin{pmatrix} y[i] \\ \boldsymbol{h} \end{pmatrix}^{\mathrm{H}} \begin{pmatrix} 1 \\ -\boldsymbol{x}^\star \end{pmatrix} \begin{pmatrix} 1 \\ -\boldsymbol{x}^\star \end{pmatrix}^{\mathrm{H}} \begin{pmatrix} y[i] \\ \boldsymbol{h} \end{pmatrix} \\ &+ (\boldsymbol{h}-\boldsymbol{\mu})^{\mathrm{H}} \boldsymbol{\Sigma} (\boldsymbol{h}-\boldsymbol{\mu}) \\ =& \boldsymbol{h}^{\mathrm{H}} \boldsymbol{L} \boldsymbol{h} - \boldsymbol{h}^{\mathrm{H}} (\sigma_n^{-2} \boldsymbol{x}^\star y[i] + \boldsymbol{\Sigma}\boldsymbol{\mu}) \\ &+ (\sigma_n^{-2} \boldsymbol{x}^{\mathrm{T}} y[i]^\star + \boldsymbol{\mu}^{\mathrm{H}} \boldsymbol{\Sigma}) \boldsymbol{h} + C_1 \end{aligned} \tag{A.2}$$

where

$$\begin{aligned}\boldsymbol{\mu} &= \begin{pmatrix} \mu_0, & \ldots, & \mu_{j-1}, & 0, & \mu_{j+1}, & \ldots, & \mu_{L-1} \end{pmatrix} \\ \boldsymbol{\Sigma} &= \operatorname{diag}\begin{pmatrix} \sigma_0^{-2}, & \ldots, & \sigma_{j-1}^{-2}, & 0, & \sigma_{j+1}^{-2}, & \ldots, & \sigma_{L-1}^{-2} \end{pmatrix} \\ \boldsymbol{L} &= \sigma_n^{-2}\boldsymbol{x}^\star\boldsymbol{x}^{\mathrm{T}} + \boldsymbol{\Sigma}\end{aligned} \tag{A.3}$$

and C_1 is a constant. To calculate the message coming from the function T_i, we need to marginalize $\exp(-f_i(\boldsymbol{h}))$ for $h[i,j]$. As $\exp(-f_i(\boldsymbol{h}))$ is proportional to a jointly Gaussian distribution in $\boldsymbol{h}$ with inverse covariance matrix $\boldsymbol{L}$, the marginalization reduces to the calculation of the mean $\hat{\mu}$ and variance $\hat{\sigma}^2$ of $h[i,j]$ with respect to this distribution. These two parameters $\hat{\mu}$ and $\hat{\sigma}^2$ then fully characterize the message coming from the function T_i. The main idea in the following derivation is the permutation of the matrix $\boldsymbol{L}$ in such a way that we can exploit the block matrix inversion lemma for the inversion of $\boldsymbol{L}$. The variance $\hat{\sigma}^2$ we seek will then simply be an element on the diagonal of the inverse of $\boldsymbol{L}$.

We introduce a suitable permutation matrix $\boldsymbol{P}$[1] and obtain

$$\begin{aligned} f_i(\boldsymbol{h}) =& (\underbrace{\boldsymbol{P}\boldsymbol{h}}_{\tilde{\boldsymbol{h}}})^{\mathrm{H}} \underbrace{\boldsymbol{P}\boldsymbol{L}\boldsymbol{P}}_{\tilde{\boldsymbol{L}}}(\boldsymbol{P}\boldsymbol{h}) \\ & -(\boldsymbol{P}\boldsymbol{h})^{\mathrm{H}}(\sigma_n^{-2}\underbrace{\boldsymbol{P}\boldsymbol{x}^\star}_{\tilde{\boldsymbol{x}}^\star} y_i + \underbrace{\boldsymbol{P}\boldsymbol{\Sigma}\boldsymbol{P}}_{\tilde{\boldsymbol{\Sigma}}}(\underbrace{\boldsymbol{P}\boldsymbol{\mu}}_{\tilde{\boldsymbol{\mu}}})) \\ & -(\sigma_n^{-2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{P}y[i]^\star + (\boldsymbol{P}\boldsymbol{\mu})^{\mathrm{H}}\boldsymbol{P}\boldsymbol{\Sigma}\boldsymbol{P})(\boldsymbol{P}\boldsymbol{h}) + C_1 \\ & =(\tilde{\boldsymbol{h}} - \boldsymbol{m})^{\mathrm{H}}\tilde{\boldsymbol{L}}(\tilde{\boldsymbol{h}} - \boldsymbol{m}) + C_2 \end{aligned} \tag{A.4}$$

[1]Notice that $\boldsymbol{P} = \boldsymbol{P}^{\mathrm{T}}$ and $\boldsymbol{P}\boldsymbol{P} = \boldsymbol{I}$ hold for all permutation matrices.

where

$$
\begin{aligned}
\tilde{\boldsymbol{x}}^{\mathrm{T}} =& \begin{pmatrix} \boldsymbol{x}_{\backslash x[i-j]}^{\mathrm{T}}, & x[i-j] \end{pmatrix} \\
\tilde{\boldsymbol{h}}^{\mathrm{T}} =& \begin{pmatrix} \boldsymbol{h}_{\backslash h[i,j]}^{\mathrm{T}}, & h[i,j] \end{pmatrix} \\
\tilde{\boldsymbol{\Sigma}} =& \begin{pmatrix} \boldsymbol{\Sigma}_{\backslash} & \boldsymbol{0} \\ 0 & 0 \end{pmatrix} \\
\boldsymbol{\Sigma}_{\backslash} =& \begin{pmatrix} \sigma_0^{-2} & & & & & & \\ & \ddots & & & & & \\ & & \sigma_{j-1}^{-2} & & & & \\ & & & \sigma_{j+1}^{-2} & & & \\ & & & & \ddots & & \\ & & & & & \sigma_{L-1}^{-2} \end{pmatrix} \\
\tilde{\boldsymbol{L}} =& \sigma_n^{-2} \tilde{\boldsymbol{x}}^{\star} \tilde{\boldsymbol{x}}^{\mathrm{T}} + \tilde{\boldsymbol{\Sigma}} \\
\boldsymbol{\mu}_{\backslash}^{\mathrm{T}} =& \begin{pmatrix} \mu_0, & \ldots, & \mu_{j-1}, & \mu_{j+1}, & \ldots, & \mu_{L-1} \end{pmatrix} \\
\tilde{\boldsymbol{\mu}}^{\mathrm{T}} =& \begin{pmatrix} \boldsymbol{\mu}_{\backslash}^{\mathrm{T}}, & 0 \end{pmatrix} \\
\boldsymbol{m}^{\mathrm{T}} =& \begin{pmatrix} \boldsymbol{\mu}_{\backslash}^{\mathrm{T}}, & x[i-j]^{-1}(y[i] - \boldsymbol{x}_{\backslash x[i-j]}^{\mathrm{T}} \boldsymbol{\mu}_{\backslash}) \end{pmatrix}
\end{aligned}
\tag{A.5}
$$

(A.6)

and C_2 is some constant. Here we readily see that

$$
\begin{aligned}
\hat{\mu} =& [\boldsymbol{m}]_{[L]} \\
=& x[i-j]^{-1}(y[i] - \boldsymbol{x}_{\backslash x[i-j]}^{\mathrm{T}} \boldsymbol{\mu}_{\backslash})
\end{aligned}
\tag{A.7}
$$

By partitioning $\tilde{\boldsymbol{L}}$

$$\tilde{\boldsymbol{L}} = \left(\begin{array}{c|c} \boldsymbol{W} & x[i-j]\boldsymbol{x}^{\star}_{\setminus x[i-j]}\sigma_n^{-2} \\ \hline x[i-j]^{\star}\boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]}\sigma_n^{-2} & x[i-j]^{\star}x[i-j]\sigma_n^{-2} \end{array} \right) \tag{A.8}$$

where

$$\boldsymbol{W} = \sigma_n^{-2}\boldsymbol{x}_{\setminus x[i-j]}{}^{\star}\boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]} + \boldsymbol{\Sigma}_{\setminus} \tag{A.9}$$

and using the block matrix inversion lemma we obtain

$$\tilde{\boldsymbol{L}}^{-1} = \left(\begin{array}{c|c} (\boldsymbol{W} - \sigma_n^{-2}\boldsymbol{x}_{\setminus x[i-j]}{}^{\star}\boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]})^{-1} & \cdot \\ \hline -x[i-j]^{\star}\sigma_n^{-2}\hat{\sigma}^2\boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]}\boldsymbol{W}^{-1} & \hat{\sigma}^2 \end{array} \right) \tag{A.10}$$

where $\hat{\sigma}^{-2} = x[i-j]^{\star}x[i-j]\sigma_n^{-2} - x[i-j]^{\star}\sigma_n^{-2}\boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]}\boldsymbol{W}^{-1}x[i-j]\sigma_n^{-2}\boldsymbol{x}^{\star}_{\setminus x[i-j]}$. Also note that the matrix $\tilde{\boldsymbol{L}}^{-1}$ is hermitian and therefore the upper triangle part of this matrix is redundant. Applying the inversion lemma on $\boldsymbol{W}$ results in

$$\begin{aligned} \boldsymbol{W}^{-1} = &\boldsymbol{\Sigma}_{\setminus}^{-1} - \\ &\boldsymbol{\Sigma}_{\setminus}^{-1}\boldsymbol{x}^{\star}_{\setminus x[i-j]}(\sigma_n^2 + \boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]}\boldsymbol{\Sigma}_{\setminus}^{-1}\boldsymbol{x}^{\star}_{\setminus x[i-j]})^{-1}\boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]}\boldsymbol{\Sigma}_{\setminus}^{-1} \end{aligned} \tag{A.11}$$

and defining $z = \boldsymbol{x}^{\mathrm{T}}_{\setminus x[i-j]}\boldsymbol{\Sigma}_{\setminus}^{-1}\boldsymbol{x}^{\star}_{\setminus x[i-j]}\sigma_n^{-2}$ we get

$$\hat{\sigma}^{-2} = |x[i-j]|^2\sigma_n^{-2}(1+z)^{-1} \tag{A.12}$$

Assuming $\left(\begin{array}{cccccc} \sigma_0^2, & \ldots, & \sigma_{j-1}^2, & \sigma_{j+1}^2, & \ldots, & \sigma_{L-1}^2 \end{array} \right) > 0$, the inverse of $\tilde{\boldsymbol{L}}$ always exists.

APPENDIX B

Proof of Nondecreasing Likelihood Sequence Property

In order to prove that the sequence of likelihoods $p(\boldsymbol{y}|\hat{\boldsymbol{\theta}}^{(k)})$ is nondecreasing even if the maximization step is replaced by a gradient ascent step, we decompose $\ln p(\boldsymbol{y}|\hat{\boldsymbol{\theta}}^{(k)})$ as follows:

$$\ln p(\boldsymbol{y}|\boldsymbol{\theta}) = F(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) + \mathrm{KL}(\hat{\boldsymbol{\theta}}^{(k)}||\boldsymbol{\theta}) \tag{B.1}$$

where

$$F(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) = \int_{\boldsymbol{H}} p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)}) \ln \left(\frac{p(\boldsymbol{y}, \boldsymbol{H}|\boldsymbol{\theta})}{p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})} \right) \tag{B.2}$$

$$\mathrm{KL}(\hat{\boldsymbol{\theta}}^{(k)}||\boldsymbol{\theta}) = -\int_{\boldsymbol{H}} p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)}) \ln \left(\frac{p(\boldsymbol{H}|\boldsymbol{y}, \boldsymbol{\theta})}{p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})} \right) \tag{B.3}$$

and $\mathrm{KL}(\hat{\boldsymbol{\theta}}^{(k)}||\boldsymbol{\theta})$ is easily identified as the Kullback-Leibler divergence between $p(\boldsymbol{H}|\boldsymbol{y}, \hat{\boldsymbol{\theta}}^{(k)})$ and $p(\boldsymbol{H}|\boldsymbol{y}, \boldsymbol{\theta})$. Then choose

$$\hat{\boldsymbol{\theta}}^{(k+1)} = \nabla_{\boldsymbol{\theta}} F(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)})\Big|_{\hat{\boldsymbol{\theta}}^{(k)}} \epsilon^{(k)} + \hat{\boldsymbol{\theta}}^{(k)} \tag{B.4}$$

with $\epsilon^{(k)} > 0$ small enough. It follows that

$$F(\hat{\boldsymbol{\theta}}^{(k+1)}, \hat{\boldsymbol{\theta}}^{(k)}) \geq F(\hat{\boldsymbol{\theta}}^{(k)}, \hat{\boldsymbol{\theta}}^{(k)}) \tag{B.5}$$

and as $\mathrm{KL}(\hat{\boldsymbol{\theta}}^{(k)}||\hat{\boldsymbol{\theta}}^{(k+1)}) \geq 0$, we have

$$\ln p(\boldsymbol{y}|\hat{\boldsymbol{\theta}}^{(k+1)}) \geq \ln p(\boldsymbol{y}|\hat{\boldsymbol{\theta}}^{(k)}) \tag{B.6}$$

Note that

$$\nabla_{\boldsymbol{\theta}} F(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) = \nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}^{(k)}) \tag{B.7}$$

and the proof is complete.

REFERENCES

[1] J. Proakis and M. Salehi, *Communication Systems Engineering*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, July 2002.

[2] S. Song, A. C. Singer, and K.-M. Sung, "Soft input channel estimation for turbo equalization," *IEEE Transactions on Signal Processing*, vol. 52, no. 10, pp. 2885–2894, October 2004.

[3] K. J. Kim and R. A. Iltis, "Joint detection and channel estimation algorithms for QS-CDMA signals over time-varying channels," *IEEE Transactions on Communications*, vol. 50, no. 5, pp. 845–855, May 2002.

[4] A. Lakhzouri, E. S. Lohan, R. Hamila, and M. Renfors, "Extended Kalman filter channel estimation for line-of-sight detection in WCDMA mobile positioning," *EURASIP Journal on Applied Signal Processing*, vol. 2003, no. 13, pp. 1268–1278, 2003.

[5] W. Ling and L. Ting, "Kalman filter channel estimation based on comb-type pilot in time-varying channel," *IEEE Transactions on Communications*, vol. 50, no. 5, pp. 845–855, May 2002.

[6] A. P. Worthen and W. E. Stark, "Unified design of iterative receivers using factor graphs," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 843–849, February 2001.

[7] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721–741, November 1984.

[8] L. Ying, Z.-P. Liang, D. C. Munson, R. Koetter, and B. J. Frey, "Unwrapping of MR phase images using a Markov random field model," *IEEE Transactions on Medical Imaging*, vol. 25, no. 1, pp. 128–136, January 2006.

[9] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, February 2001.

[10] M. R. Spiegel, *Complex Variables (With an Introduction to Conformal Mapping and Its Applications)*. New York, NY: McGraw Hill Book Company, 1974.

[11] R. Hunger, "An introduction to complex differentials and complex differentiability," Associate Institute for Signal Processing, Technische Universität München, München, Tech. Rep., October 2007.

[12] S. Lang, *Complex Analysis (Graduate Texts in Mathematics)*. New York, NY: Springer-Verlag, 1993.

[13] W. Wirtinger, "Zur formalen theorie der funktionen von mehr komplexen veränderlichen," *Math. Ann.*, vol. 97, pp. 357–375, 1927.

[14] R. Remmert, *Theory of Complex Functions*. New York, NY: Springer-Verlag, 1991.

[15] J. M. Hammersley and P. Clifford, "Markov fields on finite graphs and lattices," unpublished manuscript, 1971.

[16] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *Journal of the Royal Statistical Society*, vol. 36, no. 2, pp. 192–236, 1974.

[17] Y. Weiss and W. T. Freeman, "Correctness of belief propagation in Gaussian graphical models of arbitrary topology," *Neural Computation*, vol. 13, no. 10, pp. 2173–2200, 2001.

[18] J. Zhang, "The mean field theory in EM procedures for Markov random fields," *IEEE Transactions on Signal Processing*, vol. 40, no. 10, pp. 2570–2583, October 1992.

[19] D. Chandler, *Introduction to Modern Statistical Mechanics*. Oxford, UK: Oxford University Press, 1987.

[20] 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, no. 1, pp. 1–38, 1977.