Learning in Gaussian Markov Random Fields

Thomas J. Riedl

Abstract—

I. INTRODUCTION

Many problems in Signal Processing can be cast in the framework of state estimation, in which we have state variables whose values are not directly accessible and variables whose values are available. Variables of the latter kind are also referred to as observations in this context. Usually there exists a statistical relationship between the state variables and the observations such that we can infer estimates of the states from the observations. In many cases prior knowledge about the states is also available (usually in form of a probability distribution 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 with unknown parameters. These parameters are then also subject to estimation.

In this paper 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 propose an efficient algorithm to estimate the unknown parameters. In a nutshell our algorithm can be interpreted as an approximation to the well known expectation maximization (EM) algorithm.

II. MARKOV RANDOM FIELD THEORY

Markov Random Fields (MRF) are a natural extension to the concept of Markov Chains. A MRF is described by a undirected graph. The vertices in a MRF stand for random variables and the edges impose statistical constraints on these random variables. Specifically, based on standard MRF theory, the indexed set of random variables $\mathbf{H} = \{ h[i,j] : 0 \leq i \leq M - 1, 0 \leq j \leq L - 1 \}$ is assumed to satisfy the following two conditions:

\begin{align}
 p(h[i,j]) &> 0 \\
 p(h[i,j]|\mathbf{H}_{\backslash h[i,j]}) = p(h[i,j]|\mathbf{N}[i,j])
\end{align}

where $\mathbf{H}_{\backslash h[i,j]}$ is the entire set of random variables $\mathbf{H}$ without the element $h[i,j]$ and $\mathbf{N}[i,j]$ represents the set of $h[i,j]$’s neighboring vertices. It is well known which consequences this setup has on the joint distribution of the random variables in $\mathbf{H}$. Before we can elaborate on these however, we need to introduce the concept of a clique. A subset of $\mathbf{H}$ is called a clique if it is a singleton or if every pair of elements $h[i,j]$ in that subset is neighbors in the corresponding graph. The lattice shaped MRF considered in this paper is depicted in Figure 1. We easily identify a single clique for each vertex $h[i,j]$ and also identify cliques of the form $\{h[i,j], h[i,j-1]\}$ or $\{h[i,j], h[i-1,j]\}$ for each pair of adjacent vertices.

According to the Hammersley-Clifford theorem \[\] an MRF has…

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure1.png}
\caption{Figure 1. lattice shaped Markov Random Field}
\end{figure}

\begin{equation}
 p(\mathbf{H}) \sim \exp(-\sum_{b \in \mathbf{B}} V_b(\bar{\mathbf{b}}))
\end{equation}

where $K$ is a temperature parameter chosen to be unity in this paper. The argument of the exponential function includes a sum of clique potentials $V_b(\bar{\mathbf{b}})$ over all the possible cliques $B$, with $\bar{\mathbf{b}}$ denoting a vector composed of the set of vertices $h[i,j]$ within the clique $b$. The clique potentials $V_b(\bar{\mathbf{b}})$ are simply defined to be nonnegative functions of their arguments. In this paper we take the potential of pairwise cliques as the square of adjacent differences:

\begin{equation}
 V_b(\bar{\mathbf{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{equation}

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 a random variable $h[i,j]$ with index $i = 0$ form a gaussian distribution

\begin{equation}
 V_b(\bar{\mathbf{b}}) = \alpha_{[j_b]} |h[i_b,j_b] - \mu[j_b]|^2
\end{equation}

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

\begin{equation}
 p(\mathbf{H}|\theta) = Z(\theta)^{-1} \prod_{b \in \mathbf{B}_1} \exp(-\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{equation}

\begin{equation}
 \cdot \prod_{b \in \mathbf{B}_2} \exp(-\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{equation}

where the vector $\theta$ 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 $\mathbf{B}_1$ comprises all single cliques that correspond to random variables $h[i,j]$ with...
index \(i = 0\) and the set \(B_2\) contains all the pairwise cliques. \(Z(\theta)\) is a normalization constant and is also referred to as the partition function in this context.

### III. Parameter Estimation

#### A. Problem setup

As mentioned in the introduction we consider an incomplete data problem where some of the variables are hidden and others are observable. The hidden variables are modeled by the proposed MRF model. The observations \(y[i]\) and the hidden variables \(H\) are assumed to have the following relation

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

where the \(x[i, j]\) denote some known non-zero complex numbers, \(w[i]\) is additive white gaussian noise with zero mean and variance \(\sigma^2_n\) and \(M\) is the total number of observations. So we have the following statistical relationship between the hidden variables and the observations

\[
p(y|H, \theta) = (\pi\sigma^2_n)^{-M} \prod_{i=0}^{M-1} \exp(-\sigma^{-2}_n |y[i] - \sum_{j=0}^{L-1} h[i, j] x[i, j]|^2)^2 = T_i(H)
\]

(7)

where \(\theta\) now also comprises the noise variance \(\sigma^2_n\). Our goal is to estimate the parameters \(\theta\) and we do so in a Maximum Likelihood (ML) fashion.

\[
\hat{\theta} = \arg\max_{\theta} p(y|\theta) = \arg\max_{\theta} \int_H p(y, H|\theta)
\]

where it can easily be checked that

\[
p(y, H|\theta) = p(y|H, \theta)p(H|\theta)
\]

The contribution of this paper 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 [1]. A short review of this algorithm is in order:

1. Make some initial guess \(\theta^{(0)}\)
2. Expectation step: calculate

\[
Q(\theta, \hat{\theta}^{(k)}) = \langle \log p(y, H|\theta) \rangle_{p(H|y, \hat{\theta}^{(k)})}
\]

(10)

3. Maximization step: compute

\[
\hat{\theta}^{(k+1)} = \arg\max_{\theta} Q(\theta, \hat{\theta}^{(k)})
\]

(11)

4. Repeat 10-11 until convergence or until the available time is over.

\(\langle \cdot \rangle_p\) represents expectation with respect to \(p\). This algorithm is proven to yield a nondecreasing sequence \(p(y|\theta^{(k)})\). However, it is well known that due to the interaction between the \(h[i, j]\), the precise calculation of the partition function \(Z(\theta)\) and the integral in 10 is not computationally feasible [2].

#### B. Solution

One can bypass the requirement of exactly knowing the partition function \(Z(\theta)\) by approximating the maximization step above by a gradient ascent step. It should be noted, however, that taking all parameters in \(\theta\) 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 \(\alpha_{[i,j],[l,m]}\) that correspond to a vertical pairwise clique are the same and equal \(\alpha_v\) and similarly that all \(\alpha_{[i,j],[l,m]}\) that correspond to a horizontal pairwise clique are the same and equal \(\alpha_h\). Also we take \(\alpha_{[i]} = \alpha\).

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

\[
\frac{\partial}{\partial \sigma^2_n} Q(\theta, \hat{\theta}^{(k)}) = -M \frac{\partial}{\partial \sigma^2_n} \sum_{i=0}^{M-1} \langle |y[i] - \sum_{j=0}^{L-1} h[i, j] x[i, j]|^2 \rangle_{p(H|y, \hat{\theta}^{(k)})}
\]

(12)

And the partial derivatives with respect to the MRF parameters \(\theta_j\) have the following form

\[
\frac{\partial}{\partial \theta_j} Q(\theta, \hat{\theta}^{(k)}) = \langle \frac{\partial}{\partial \theta_j} \sum_{b \in B} V_b(\hat{\theta}) \rangle_{p(H|y, \hat{\theta}^{(k)})} - \langle \frac{\partial}{\partial \theta_j} \sum_{b \in B} V_b(\hat{\theta}) \rangle_{p(H|y, \hat{\theta}^{(k)})}
\]

(13)

where

\[
\frac{\partial}{\partial \theta_j} \sum_{b \in B} V_b(\hat{\theta}) = 2 \left( \sigma^{-2}_{v,j-1} \langle \sum_i h[i, j - 1] - M \mu[j - 1] \rangle + \sigma^{-2}_{v,j} \langle \sum_i h[i, j + 1] - M \mu[j + 1] \rangle 
\]

\[
- \langle \sigma^{-2}_j + \sigma^{-2}_{v,j-1} + \sigma^{-2}_{v,j} \rangle \langle \sum_i h[i, j] - M \mu[j] \rangle \right)
\]

(14)

\[
\frac{\partial}{\partial \sigma^{-2}_n} \sum_{b \in B} V_b(\hat{\theta}) = \sum_i \langle h[i, j] - \mu_j \rangle^2
\]

(15)

\[
\frac{\partial}{\partial \sigma^{-2}_{v,j}} \sum_{b \in B} V_b(\hat{\theta}) = \sum_i \langle h[i, j] - \mu_j \rangle - \langle h[i, j + 1] - \mu_{j+1} \rangle^2
\]

(16)

\[
\frac{\partial}{\partial \sigma^{-2}_{h,j}} \sum_{b \in B} V_b(\hat{\theta}) = \sum_i \langle h[i, j] - h[i + 1, j] \rangle^2
\]

(17)

Clearly in order to actually calculate the gradient at \(\theta = \hat{\theta}^{(k)}\) we first need to know the moments \(\langle h[i, j] \rangle, \text{Cov}(h[i, j])\), \(\text{Cov}(h[i, j], h[i, j + 1])\) and \(\text{Cov}(h[i, j], h[i + 1, j])\) with respect to \(p(H|\hat{\theta}^{(k)})\) and also with respect to \(p(H|y, \hat{\theta}^{(k)})\). This can be achieved by use of the sum-product algorithm. Note that the random variables in \(H\) are jointly gaussian and hence the messages that the sum-product algorithm passes along the edges of a factor graph are gaussian as well. Gaussian messages are fully characterized 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
For that reason it suffices to derive the message passing rules for the factor graph associated with $p(H; y, \hat{\theta}^{(k)})$ in Figure 2. The black boxes and circles represent function neighbors a given node as $h$ a variable node $p$ and variable nodes, respectively. The nice factorization of $p(H; y, \hat{\theta}^{(k)})$ that the MRF framework provides reduces the complexity of the marginalization tremendously. The factor graph corresponding to $p(H; \hat{\theta}^{(k)})$ coincides with the one in Figure 2 when the functions $T_i$ are dropped.

The remainder of this section is dedicated to the concrete implementation of the sum-product algorithm for the calculation of the moments we are interested in. The factor graph corresponding to $p(H; \hat{\theta}^{(k)})$ is contained in the factor graph corresponding to $p(H; y, \hat{\theta}^{(k)})$ and hence the message passing on $p(H; \hat{\theta}^{(k)})$ is just a special case of the one on $p(H; y, \hat{\theta}^{(k)})$. For that reason it suffices to derive the message passing rules for the factor graph associated with $p(H; \hat{\theta}^{(k)})$.

Following the notation of [1] we denote messages sent from a variable node $h[i, j] \in H$ to a local function node as $m_{h \rightarrow f}(h)$. Here $f$ represents either one of the functions $T_i$ or one of the potential functions $V_h(\hat{b})$. Furthermore we denote messages sent from a local function node to a variable node as $m_{f \rightarrow h}(h)$. Also, let $n(v)$ denote the set of nodes that neighbors a given node $v$ in a factor graph. Then the message computations performed by the sum-product algorithm may be expressed as follows:

**variable to local function**

$$m_{h \rightarrow f}(h) = \prod_{g \in n(h) \setminus \{f\}} m_{g \rightarrow h}(h) \quad (18)$$

**local function to variable**

$$m_{f \rightarrow h}(h) = \int_{\sim(h)} f(X) \prod_{u \in n(f) \setminus \{h\}} m_{u \rightarrow f}(u) \quad (19)$$

Of course these rules are rather general, but before we derive more specific message computation rules, let us focus on the factor graph shown in figure 2 again. We easily see that the messages that are sent along the edges of our factor graph are basically just of three different 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 different types of messages are illustrated in figure 3, where the kind of the message is superscripted in each case. As mentioned above the messages are gaussian meaning that every message $m$ is completely characterized by its mean $\mu_m$ and variance $\sigma_m^2$. The derivation of update rules for messages of the first two kinds is straight forward, the derivation of an update rule for messages of the third kind is more involved and the interested reader is referred to the appendix. Let us just summarize the results here.

**update rule for messages of the first kind**

$$\mu_{m_{h \rightarrow f}} = \frac{\sum_{g \in n(h) \setminus \{f\}} \sigma^{-2}_g \mu_g}{\sum_{g \in n(h) \setminus \{f\}} \sigma^{-2}_g} \quad (20)$$

$$\sigma^{-2}_{m_{h \rightarrow f}} = \sum_{g \in n(h) \setminus \{f\}} \sigma^{-2}_g$$

where $\mu_g$ and $\sigma^2_g$ are the mean and the variance of the message $m_{g \rightarrow h}(h)$, respectively.

**update rule for messages of the second kind**

$$\mu_{m_{V_h \rightarrow h}} = \mu_u + \mu_j[i_h] - \mu_j[i_u] \quad (21)$$

$$\sigma^{-2}_{m_{V_h \rightarrow h}} = \sigma^{-2}_u + \sigma^{-2}_{\alpha_{i_h, j_h}[i_u, j_u]} \quad (22)$$

where $\mu_u$ and $\sigma^2_u$ are the mean and the variance of the message $m_{u \rightarrow V_h}(u)$, $u \in n(V_h) \setminus \{h\}$, 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**
where  

\[
\sigma_n^{-2} m_{T_i \rightarrow h[i,j]} = |x[i-j]|^2 \sigma_n^{-2} (1 + z)^{-1} 
\]

\[
z = \sigma_n^{-2} \sum_{l=0,l\neq j}^{L-1} |x[i-l]|^2 \sigma_m h[i,l] \rightarrow T_i 
\]

\[
\mu_{m_{T_i \rightarrow h[i,j]}} = x[i-j]^{-1} (y[i] - \sum_{l=0,l\neq j}^{L-1} x[i-l] \mu_{m_{h[i,l] \rightarrow T_i}}) 
\]

The above rules cover the message passing required for calculation of the moments \( \langle h[i,j] \rangle \) and \( \text{Cov}(h[i,j]) \). In order to obtain the moments \( \text{Cov}([h[i,j], h[i,j+1]]) \) and \( \text{Cov}([h[i,j], h[i+1,j]]) \) 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 marginal. 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 \( \text{Cov}([h[i,j], h[i,j+1]]) \) and \( \text{Cov}([h[i,j], h[i+1,j]]) \), we cluster the corresponding pairs of nodes \( h[i,j] \). Figure 4 shows what effect the clustering of \( (h[i,j], h[i,j+1]) \) has on the factor graph in Figure 3. The update rule for messages of the third kind must then be modified as follows.

\[
\Sigma_n^{-1} m_{T_i \rightarrow (h[i,j], h[i,j+1])} = \left[ \begin{array}{c} x[i-j] \\ x[i-j-1] \end{array} \right]^T \left[ \begin{array}{c} x[i-j] \\ x[i-j-1] \end{array} \right] + \sigma_n^{-2} (1 + z)^{-1} \]

\[
z = \sigma_n^{-2} \sum_{l=0,l\neq j+1}^{L-1} |x[i-l]|^2 \sigma_m h[i,l] \rightarrow T_i 
\]

\[
\mu_{m_{T_i \rightarrow (h[i,j], h[i,j+1])}} = \left[ \begin{array}{c} 1 \\ 0 \end{array} \right] x[i-j]^{-1} 
\]

\[
(y[i] - \sum_{l=0,l\neq j+1}^{L-1} x[i-l] \mu_{m_{h[i,l] \rightarrow T_i}}) 
\]

As mentioned above it is well known that the EM algorithm yields a nondecreasing sequence of likelihoods \( p(y|\hat{\theta}(k)) \). 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 the appendix. 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. correctness of means in gaussian message passing...
APPENDIX

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

\[
\ln p(y|\theta) = F(\theta, \hat{\theta}^{(k)}) + KL(\hat{\theta}^{(k)}||\theta)
\]

where

\[
F(\theta, \hat{\theta}^{(k)}) = \int_H p(H|y, \hat{\theta}^{(k)}) \ln \left( \frac{p(y,H|\theta)}{p(H|y, \hat{\theta}^{(k)})} \right)
\]

\[
KL(\hat{\theta}^{(k)}||\theta) = -\int_H p(H|y, \hat{\theta}^{(k)}) \ln \left( \frac{p(H|y, \theta)}{p(H|y, \hat{\theta}^{(k)})} \right)
\]

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

\[
\hat{\theta}^{(k+1)} = \nabla_\theta F(\theta, \hat{\theta}^{(k)}) \bigg|_{\hat{\theta}^{(k)}} \epsilon^{(k)} + \hat{\theta}^{(k)}
\]

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

\[
F(\hat{\theta}^{(k+1)}, \hat{\theta}^{(k)}) \geq F(\hat{\theta}^{(k)}, \hat{\theta}^{(k)})
\]

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

\[
\ln p(y|\hat{\theta}^{(k+1)}) \geq \ln p(y|\hat{\theta}^{(k)})
\]

Note that

\[
\nabla_\theta F(\theta, \hat{\theta}^{(k)}) = \nabla_\theta Q(\theta, \hat{\theta}^{(k)})
\]

and the proof is complete.