Utilizing Variational Optimization to Learn Markov Random Fields

Marshall F. Tappen
University of Central Florida
Orlando, FL
mtappen@eecs.ucf.edu

Abstract

Markov Random Field, or MRF, models are a powerful tool for modeling images. While much progress has been made in algorithms for inference in MRFs, learning the parameters of an MRF is still a challenging problem. In this paper, we show how variational optimization can be used to learn the parameters of an MRF. This method for learning, which we refer to as Variational Mode Learning, finds the MRF parameters by minimizing a loss function that penalizes the difference between ground-truth images and an approximate, variational solution to the MRF. In particular, we focus on learning parameters for the Field of Experts model of Roth and Black. In addition to demonstrating the effectiveness of this method, we show that a model based on derivative filters performs quite similarly to the Field of Experts model. This suggests that the Field of Experts model, which is difficult to interpret, can be understood as imposing piecewise continuity on the image.

1. Introduction

Markov Random Field (MRF) models are a powerful tool for modeling images because they allow long-term interactions between pixels to be expressed through a combination of local interactions. While these models have a long history in computer vision [14, 12], their popularity has been spurred by development of efficient, powerful algorithms for inference and energy minimization in these models [5, 23, 21]. While MRFs are often combined with classifiers and estimators that have been built using training examples, the parameters of the MRF are often hand-designed according to some heuristic, such as the system in [15]. These models are hand-designed because the inference step often required for learning MRF parameters is infeasible for the types of models often used in vision applications.

In this paper, we propose a new method for learning the parameters of continuous-valued MRF models. We focus on models that are similar to the Field of Experts model proposed by Roth and Black [13]. Rather than maximizing the likelihood of the training data, the MRF parameters are found by minimizing a loss function that measures the difference between the ground-truth image and optimal image under the MRF model. A unique aspect of our approach is that by using an approximation to this optimal image, the loss can be differentiated with respect to the parameters. This approximate mode of the MRF, which is central to the success of our method, is computed using variational optimization – leading us to refer to this method as Variational Mode Learning.

Using Variational Mode Learning, we train two different MRF models to denoise images. One model is identical to the Field of Experts model, while the other model is designed around a fixed set of derivative filters. The second model is useful because it can be intuitively interpreted as enforcing piecewise continuity on the image. The denoised images returned by this second model are extremely similar to the images produced using the parameters from [13]. These similarities indicate that the Field of Experts model can likewise be understood as imposing higher-order continuity on images.

Below, Section 1.1 describes related approaches for learning MRF parameters, while Section 2 discusses background material on MRFs. Section 4 describes the variational optimization at the core of the learning algorithm. The actual learning algorithm is discussed in Section 5. Finally, Section 6 describes the experiments and discusses how the Field of Experts model can be understood in terms of continuity.

1.1. Related Work

The problem of learning the parameters of discrete-valued MRF is considered in both the work of Levin and Weiss [11] and Anguelov et al. [2]. Both papers learn parameters for labeling pixels in an image. In this paper, we instead focus on continuous-valued models. Continuous-valued models are more appropriate for modeling images because the computational complexity of inference and other tasks is independent of the dynamic range of the image. Besides the computational issues involved with the number of states needed per pixel to model images, the underlying approximations used in these discrete-models limit the types of relationships that can be expressed between pixels. The bounds used in [11], developed by Wainwright et al. [22], are limited, practically, to modeling relationships between pairs of pixels. The learning technique used in [2] is limited to learning parameters in MRFs with a specific, limited type of relationship between pixels, which is referred to as a generalized Potts model.

In [13], Roth and Black learn the parameters of a continuous-valued MRF model by using a sampling strat-

egy based on a criterion known as the contrastive divergence [8]. Using this method, the parameters are updated using each training point to draw one sample from the current distribution. There are both practical and theoretical disadvantages to this sampling-based optimization. Theoretically, it is still unknown whether the contrastive divergence algorithm converges – although there is strong belief that it does [6]. From a practical point of view, the method is only able to compute approximate gradients, which may be problematic for optimization methods that are sensitive to errors in the gradient calculation.

Viewed in the context of these other methods, the Variational Mode Learning strategy described in Section 5 has several particularly attractive properties:

1. In Variational Mode Learning, exact derivative calculations are used to optimize the MRF parameters. One benefit of using exact gradients is that convergence can be guaranteed by choosing the appropriate non-linear optimization algorithm.

2. Like the contrastive divergence scheme used in [13], it is possible to learn the parameters for models that consider the relationship between relatively large groups of pixels.

3. As Section 5 will show, this strategy is based on linear algebra routines that are available in most programming environments. This makes implementation of Variational Mode Learning relatively convenient. Our entire implementation of the training routines consisted of roughly 350 lines of MATLAB code. A sample implementation will be available at `http://www.eecs.ucf.edu/~mtappen`.

2. Markov Random Field Models and Energy Functions

An MRF can be written as a Gibbs distribution:

$$p(\mathbf{x};\theta) = \frac{1}{Z}\exp\left(-\sum_{k=1}^{N_C} V_k(\mathbf{x}_{(k)};\theta)\right) \tag{1}$$

where the sum is over all N_C cliques, k, in the graph. Each function $V_k(\mathbf{x}_{(k)};\theta)$ is a clique potential function associated with clique $\mathbf{x}_{(k)}$. The potential functions also have a set of parameters, θ. The term $\frac{1}{Z}$, also known as the partition function, is a normalization constant that ensures the density integrates to 1.

In many vision applications, the clique potential functions often represent some sort of smoothness assumption. Assuming that the distribution is over images and that x can be structured as an image, many smoothness assumptions can be represented as

$$V_k(\mathbf{x};\theta) = \rho(\theta^T\mathbf{x}_k) \tag{2}$$

where the vector x_k contains the variables in clique k and θ is a vector of coefficients. The function $\rho(\cdot)$ is some error function, such as $\rho(x) = x^2$ [3].

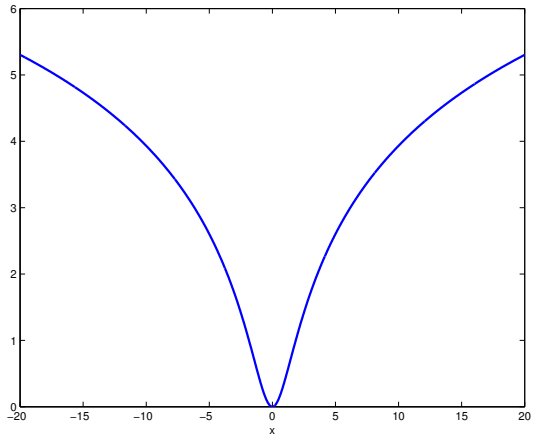


Figure 1. This graph shows a plot of the Lorentzian error function. As the error rises, the rate of increase in the error function decreases. This makes this function robust to large outlier errors.

This model can represent many popular smoothness assumptions. For example, if $\rho(x) = x^2$ and $\theta^T\mathbf{x}_k$ is equivalent to computing the difference between neighboring pixels, Equation 2 is equivalent to the membrane model from [14, 19]. Making $\theta^T\mathbf{x}_k$ equivalent to computing second derivatives would instead represent the thin plate model.

Recently, researchers have demonstrated improved results on applications such as super-resolution [17], CCD demosaicing [17], image in-painting [13], and image denoising [13] by setting $\rho(\cdot)$ to be a robust error function. Examples of robust error functions include the truncated quadratic function and the Lorentzian error function [3] which is

$$\rho(x) = \log\left(1 + \frac{1}{2}x^2\right) \tag{3}$$

A unique feature of robust error functions is that, unlike the quadratic error function, the magnitude of the derivative of the error function does not increase as the error rises. As shown in Figure 1, the rate of increase in the Lorentzian error function actually decreases as the error rises. This property makes this function robust to large outlier errors.

The Field of Experts model uses the Lorentzian error function to define the clique potentials. The model is defined by a set of linear filters, $f_1 \ldots f_{N_f}$, and a set of weights, $\alpha_1 \ldots \alpha_{N_f}$, that are associated with the filters. These filters and weights define a probability density over images, $\mathbf{x}$:

$$p(\mathbf{x}) = \frac{1}{Z}\exp\left(-\sum_{i=1}^{N_f}\alpha_i\sum_{r,c}\rho\left((\mathbf{x} * f_i)(r,c)\right)\right) \tag{4}$$

where $(\mathbf{x} * f_i)(r,c)$ denotes the pixel at location (r,c) in the image produced by convolving $\mathbf{x}$ with the filter f_i. The inner sum is over all locations in each filtered image.

In [13], the density in Equation 4 is used as a prior. Depending on the task, it is combined with different likelihood functions to form posterior distributions over images. For

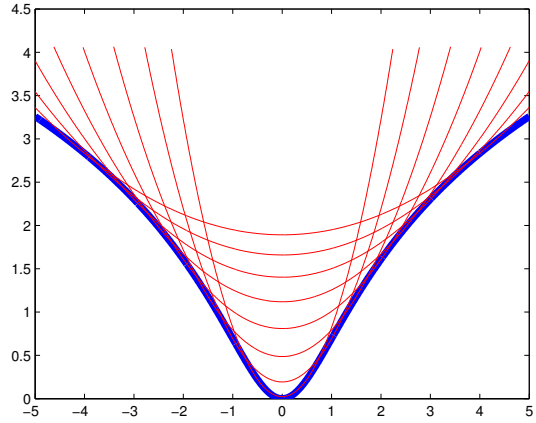


Figure 2. This graph shows the Lorentzian error function, plotted in blue, with a set of quadratic upper bounds, plotted in red. Each quadratic function corresponds to a particular value of the variational parameter λ. It can be shown that the Lorentzian error function is the infimum of the set of these quadratic functions.

example, when denoising images, the Field of Experts prior is combined with a Gaussian likelihood function:

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z}\exp \quad \left(-\beta \sum_{r,c} \left(\mathbf{x}(r,c) - \mathbf{y}(r,c)\right)^2 - \sum_{i=1}^{N_f} \alpha_i \sum_{r,c} \rho\left((\mathbf{x} * f_i)(r,c)\right)\right) \tag{5}$$

where $\mathbf{y}$ is a noisy observation of $\mathbf{x}$. The term β is a scalar weight. Denoised estimates of $\mathbf{x}$ can be found by maximizing either Equation 5 or the log of Equation 5. Throughout this paper, we will focus on models with the same form as 5, where the relationship between the observation, $\mathbf{y}$, and the latent image, $\mathbf{x}$, is Gaussian. We will also assume that $\beta = 1$.

3. Upper Bounds and Variational Methods

Given the noisy observation, $\mathbf{y}$, the denoised estimate of $\mathbf{x}$ can be found by maximizing Equation 5. This section introduces the basic ideas behind a variational method for optimizing a model like Equation 5 to find the denoised estimate of $\mathbf{x}$. This variational optimization method, described in Section 4, will form the basis of the learning algorithm described in Section 5.

In [9], Jordan et al. describe how $\log(x)$ can be upper-bounded by a family of linear functions. The Lorentzian error function cannot be upper-bounded by a linear function, but it can be bounded by a quadratic function. For brevity, we will drop the $\frac{1}{2}$ term from the Lorentzian function for the rest of this paper and focus on the function

$$\rho(x) = \log(1 + x^2)$$

We define each quadratic upper bound as a quadratic function in x with coefficients determined by λ:

$$h(x, \lambda) = a(\lambda)x^2 + b(\lambda)x + d(\lambda) \tag{6}$$

The coefficent functions $a(\lambda)$, $b(\lambda)$, and $d(\lambda)$ are chosen by imposing the following constraints on $h(\cdot)$:

$$\begin{aligned} h(\lambda_0, \lambda_0) &= \log(1 + \lambda_0^2) \\ h'(\lambda_0, \lambda_0) &= \tfrac{2\lambda_0}{1+\lambda_0^2} \\ h'(0, \lambda_0) &= 0 \end{aligned} \tag{7}$$

The first constraint implies that the value of the quadratic function determined by λ_0 must be equal to the value of the Lorentzian function at λ_0. The second constraint requires that the quadratic bound have the same slope as the Lorentzian at λ_0, while the third constraint requires that the approximation has its minimum at 0. Effectively, λ is a variational parameter that indexes a set of quadratic upper-bounds on the Lorentzian error function. Each of these upper bounds touches the Lorentzian at two locations. Figure 2 shows these upper bounds for different values of λ.

As Figure 2 shows, the Lorentzian error function can also be written as a minimization over the variational parameter λ:

$$\log(1 + x^2) = \min_{\lambda} \left(\frac{x^2}{1 + \lambda^2} + \log(1 + \lambda^2) - \frac{\lambda^2}{1 + \lambda^2} \right) \tag{8}$$

This can be verified by differentiating

$$\frac{x^2}{1 + \lambda^2} + \log(1 + \lambda^2) - \frac{\lambda^2}{1 + \lambda^2} \tag{9}$$

with respect to λ. The derivative will be equal to zero at $\lambda = -x, 0$ and x. Checking the second derivative reveals that the second derivative is only positive at $\lambda = -x$ and $\lambda = x$. Plugging either of these values into Equation 9 verifies Equation 8. Families of quadratic upper-bounds for other robust penalty functions are described in [7] and [3].

4. Variational Optimization

Using this variational formulation of $\log(1 + x^2)$, the task of finding the image, $\mathbf{x}$, that minimizes the negative log-posterior of the model can now be rewritten as:

$$\min_{\mathbf{x},\lambda} \sum_{i=1}^{N_f} \alpha_i \sum_{r,c} \frac{(\mathbf{x} * f_i)(r,c)^2}{1 + \lambda_{i,r,c}^2} + \log(1 + \lambda_{i,r,c}^2) - \frac{\lambda_{i,r,c}^2}{1 + \lambda_{i,r,c}^2} + \sum_{r,c} \left(\mathbf{x}(r,c) - \mathbf{y}(r,c)\right)^2 \tag{10}$$

Here $\lambda_{i,x,y}$ denotes the unique variational parameter associated with every term in the exponent of Equation 5.

This suggests a two-step coordinate descent algorithm for finding the image, $\mathbf{x}$, that minimizes the variational negative log-likelihood in Equation 10.

1. Treat all values of λ as constants and minimize Equation 10 in terms of $\mathbf{x}$.

   The value of $\mathbf{x}$ that minimizes Equation 10 can be computed in closed-form using the pseudoinverse. More details are given in Section 4.1.

2. Treat $\mathbf{x}$ as a constant and minimize Equation 10 with respect to each λ by setting

$$\lambda_{i,r,c} \leftarrow (\mathbf{x} * f_i)(r,c) \tag{11}$$

for all filters and pixel responses. This step minimizes Equation 10 because, using Equation 8, each term

$$\frac{(\mathbf{x} * f_i)(r,c)^2}{1+\lambda_{i,r,c}^2} + \log(1+\lambda_{i,r,c}^2) - \frac{\lambda_{i,r,c}^2}{1+\lambda_{i,r,c}^2}$$

will be minimized when it equals $\log(1 + (\mathbf{x} * f_i)(r,c)^2)$. Setting each $\lambda_{i,r,c}$ to $(\mathbf{x} * f_i)(r,c)$ makes each of these terms equal to $\log\left(1 + (\mathbf{x} * f_i)(r,c)^2\right)$.

In each of these two steps, the optimal solution for one set of variables is computed, thus guaranteeing that the energy will never increase. The initial value of $\mathbf{x}$ can be any image, but the observed image, $\mathbf{y}$, is a logical initialization.

This variational optimization method is similar to continuation methods and the Graduated Non-Convexity (GNC) method [4, 12]. In the GNC method, a non-convex energy function is approximated with a set of approximate energy functions. The first approximation is convex, then succeeding approximations are adjusted to more closely match the energy function. Unlike the variational optimization described above, the successive approximations are not necessarily convex.

4.1. Minimizing with Respect to the Image

In Step 1 above, Equation 10 is minimized with respect to the current image estimate, $\mathbf{x}$. This section describes the basic steps for performing this minimization for the basic Field of Experts model described in Equation 5.

Performing this minimization is described most easily using matrix notation. Treating all λ as constants, Equation 10 can be rewritten using matrix notation by defining a set of convolution matrices $F_1 \dots F_{N_f}$. These matrices are defined such that multiplying an image, $\mathbf{x}$, that has been unwrapped into a vector, with a matrix F_j will return the unwrapped version of $\mathbf{x} * f_j$. Essentially, each matrix F_j represents convolution with filter f_j. After dropping the constant terms in Equation 10, the quadratic minimization over $\mathbf{x}$ can be rewritten as

$$F^T W(\lambda) F, \text{ where } F = \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_{N_f} \end{bmatrix} \tag{12}$$

The matrix $W(\lambda)$ is a diagonal matrix with the entry on the diagonal in row r', $w_{r'}$ set to

$$w_{r'} = \frac{\alpha_i}{1+\lambda_{i,r,c}^2} \tag{13}$$

where i, r, and c, are the image indices of the constraint corresponding to row r'.

The quadratic term in Equation 10 can be incorporated by appending the identity matrix to F and solving the system $(F^T W(\lambda) F)^{-1} F^T W \hat{\mathbf{y}}$, where $\hat{\mathbf{y}}$ is a vector created by unwrapping the observation and prepending the correct number of zeros. The matrix $W(\lambda)$ is likewise extended with ones along the diagonal.

Thus, Step 1 can be rewritten as

$$\mathbf{x} \leftarrow \left(F^T W(\lambda) F\right)^{-1} F^T W(\lambda) \hat{\mathbf{y}} \tag{14}$$

5. Variational Mode Learning for MRF Parameters

Having shown how the Lorentzian MRF model can be optimized using a variational technique, this section describes how to use this optimization technique to learn the MRF parameters. We refer to this method for learning parameters as Variational Mode Learning because the parameters are found by minimizing the loss of an approximate mode of the MRF. This mode is found through variational optimization. To motivate the necessity of this method, Section 5.1 discusses how traditional maximum-likelihood learning is infeasible for many MRFs.

5.1. Using the Likelihood to Learn MRF Parameters

Traditionally, the parameters of the MRF model are found by maximizing the log-likelihood of the training images, $\mathbf{t}^1 \dots \mathbf{t}^{\mathbf{N_x}}$:

$$\log p(\mathbf{t}_{1\dots N_x}; \theta) = \sum_{m=1}^{N_x} - \left(\sum_{k=1}^{N_C} V_k(\mathbf{t}_{(k)}^m, \theta) \right) - \log Z \tag{15}$$

where we have used $p(\mathbf{t}_{1\dots N_T})$ to denote $P(\mathbf{t}_1 \dots \mathbf{t}_{N_x})$.

The derivative of Equation 15 with respect to a parameter θ_j, $\frac{\partial \log P(\mathbf{t}_{1\dots N_T}; \theta)}{\partial \theta_j}$, can be expressed as using an expectation:

$$- \left(\sum_{m=1}^{N_x} \sum_{k=1}^{N_C} \frac{\partial V_c(\mathbf{t}_{(k)}^m, \theta)}{\partial \theta_j} - \mathbb{E}_{p(\mathbf{x};\theta)} \left[\frac{\partial V(\mathbf{x}_{(k)}^m, \theta)}{\partial \theta_j} \right] \right) \tag{16}$$

The expectation, denoted $\mathbb{E}[\cdot]$ is computed using $p(\mathbf{x}; \theta)$, which is the probability density over images defined by the current parameters, θ.

Unfortunately, computing this expectation is infeasible for MRF models with loops in the graph defining the MRF. The Field of Experts model has loops, but Roth and Black were able to achieve good results by instead using a sampling strategy to minimize the contrastive divergence [8]. However, as discussed in Section 1.1, it is unknown whether this method converges, and the gradients computed are not exact.

5.2. Using a Loss Function Instead

As recent work has pointed out, the difficulties of learning parameters using the log-likelihood can be avoided by focusing on the mode of the MRF's distribution [1, 10, 18]. Instead of maximizing the likelihood of the training data, the parameters can be adjusted to make the most-likely point in the MRF as similar as possible to the ground-truth image. Given a training image, $\mathbf{t}$, and a loss function $L(\mathbf{x}, \mathbf{t})$ that penalizes the difference between $\mathbf{t}$ and $\mathbf{x}$, the

parameters θ can be found by using the following optimization:

$$\begin{aligned} &\min_{\theta} L(\mathbf{x}, \mathbf{t}) \\ &\text{s.t. } \mathbf{x} = \arg\min_{\mathbf{x}} -\log p(\mathbf{x}|\mathbf{y};\theta) \end{aligned} \tag{17}$$

where $\mathbf{y}$ is the observation. Assuming a differentiable loss function, this would be straightforward to optimize with gradient descent techniques if

$$\arg\min_{\mathbf{x}} -\log p(\mathbf{x}|\mathbf{y};\theta)$$

were also differentiable with respect to θ. While this arg min expression cannot be differentiated, it can be replaced with an approximate solution.

Let $\mathbf{x}^{N_I}(\theta)$ define the result returned by running n iterations of the variational optimization steps defined in Section 4. Instead of trying to find the solution of the non-linear program in Equation 4, we propose solving the following problem:

$$\min_{\theta} L(\mathbf{x}^{N_I}(\theta), \mathbf{t}) \tag{18}$$

Essentially, we have replaced the maximum of $p(\mathbf{x}|\mathbf{y};\theta)$ with the approximate maximum computed using N_I iterations of the variational optimization method described in Section 4. As the next section will show, this approximate maximum can be differentiated analytically, thus allowing the parameters to be optimized with gradient-descent techniques.

This approach to estimating the MRF parameters is similar to that used by Geman and Reynolds [7]. Although [7] uses a different approach and assumptions to estimate the model parameters, both that approach and our approach estimate parameters based on the minima found through coordinate descent.

5.3. Differentiating the Approximate Solution

The key to optimizing Equation 18 is differentiating $\mathbf{x}^n(\theta)$ with respect to θ. These derivatives can be computed using this identity from matrix calculus:

$$\frac{\partial A^{-1}}{\partial x} = -A^{-1}\frac{\partial A}{\partial x}A^{-1} \tag{19}$$

The derivative of the loss, which we will shorten to L, computed with respect to a single parameter, θ_j is

$$\frac{\partial L}{\partial \theta_j} = \frac{\partial L}{\partial \mathbf{x}^n}\frac{\partial \mathbf{x}^n}{\partial \theta_j} \tag{20}$$

where $\frac{\partial L}{\partial \mathbf{x}^n}$ is a $1 \times N_p$ vector and $\frac{\partial \mathbf{x}^n}{\partial \theta_j}$ is a $N_p \times 1$ vector.

Using the chain rule, $\frac{\partial \mathbf{x}^n}{\theta_j}$ can be computed. Recall, from Section 4.1, that at each iteration

$$\mathbf{x} \leftarrow \left(F^T W(\lambda) F\right)^{-1} F^T W(\lambda)\hat{\mathbf{y}} \tag{21}$$

The diagonal weight matrix $W(\lambda)$ is computed using the variational parameters, which are in turn computed from the previous value of $\mathbf{x}$.

Hence, $\mathbf{x}^n(\theta)$ can be rewritten as

$$\mathbf{x}^n(\theta) \leftarrow \left(F^T W(\mathbf{x}^{n-1}(\theta),\theta) F\right)^{-1} F^T W(\mathbf{x}^{n-1}(\theta),\theta)\hat{\mathbf{y}} \tag{22}$$

The one tricky aspect of differentiating Equation 22 is that $\mathbf{x}^n(\theta)$ depends on θ in two ways. First, the parameters, θ, directly influence the weighting matrix, $W(\mathbf{x}^{n-1}(\theta),\theta)$ and the filter matrices $F_1 \dots F_{N_f}$, if the filters are part of the parameters. The second way that θ affects $\mathbf{x}^n(\theta)$ is that $W(\mathbf{x}^{n-1}(\theta),\theta)$ also depends on $\mathbf{x}^{n-1}(\theta)$.

For clarity in the derivations below, we will introduce a new term $\mathbf{x}^n(\mathbf{x}_*^{n-1},\theta)$. When using this notation, we treat $\mathbf{x}^{n-1}$ as a constant that does not depend on θ. Likewise, $\mathbf{x}^n(\mathbf{x}^{n-1},\theta_*)$ will denote that θ is being treated as a constant.

Using this notation, $\frac{\partial L}{\partial \theta_j}$ can be written as

$$\frac{\partial L}{\partial \theta_j} = \frac{\partial L}{\partial \mathbf{x}^n}\frac{\partial \mathbf{x}^n(\mathbf{x}_*^{n-1},\theta)}{\partial \theta_j} + \frac{\partial L}{\partial \mathbf{x}^n}\frac{\partial \mathbf{x}^n(\mathbf{x}^{n-1},\theta_*)}{\partial \theta_j} \tag{23}$$

The second term in Equation 23 can be computed using the Jacobian matrix:

$$\frac{\partial L}{\partial \mathbf{x}^n}\frac{\partial \mathbf{x}^n(\mathbf{x}^{n-1},\theta_*)}{\partial \theta_j} = \frac{\partial L}{\partial \mathbf{x}^n}J_{\mathbf{x}}^n(\mathbf{x}^{n-1})\frac{\partial X^{n-1}}{\theta_j} \tag{24}$$

where $J_{\mathbf{x}}^n(\mathbf{x}^{n-1})$ is the Jacobian matrix relating $\mathbf{x}^n$ and $\mathbf{x}^{n-1}$.

Due to space limitations, we will not describe detailed derivations for computing quantities such as $\frac{\partial \mathbf{x}^n(\mathbf{x}_*^{n-1},\theta)}{\partial \theta_j}$, though derivations for similar quantities can be found in [16]. A sample MATLAB implementation of these steps will be available at `http://www.eecs.ucf.edu/~mtappen`.

5.4. Computing the Derivatives Efficiently

Equation 24 suggests an efficient algorithm, reminiscent of the back-propagation algorithm for neural networks, for computing $\frac{\partial L}{\partial \theta_j}$:

1. Initialize $\frac{\partial L}{\partial \theta_j}$ to all zeros.
2. Initialize $\frac{\partial L}{\partial \mathbf{x}^{N_I}}$, where N_I is the number of variational optimization steps used to compute $\mathbf{x}^{N_I}$. The optimization is started with the initial estimate $\mathbf{x}^0$.
3. For $n = N_I \dots 0$
   (a) $\frac{\partial L}{\partial \theta_j} \leftarrow \frac{\partial L}{\partial \theta_j} + \frac{\partial L}{\partial \mathbf{x}^n}\frac{\partial \mathbf{x}^n(\mathbf{x}_*^{n-1},\theta)}{\partial \theta_j}$
   (b) If $n > 0$, $\frac{\partial L}{\partial \mathbf{x}^{n-1}} \leftarrow \frac{\partial L}{\partial \mathbf{x}^n}J_{\mathbf{x}}^n(\mathbf{x}^{n-1})$

6. Learning Parameters for Denoising

In this section, we apply the learning algorithm described in the previous section to find MRF parameters for denoising images. Variational Mode Learning was used to find the parameters for two different models. Section 6.1 describes the results for a model that uses a fixed set of derivative filters, while Section 6.3 describes the results when learning a model identical to the Field of Experts model.

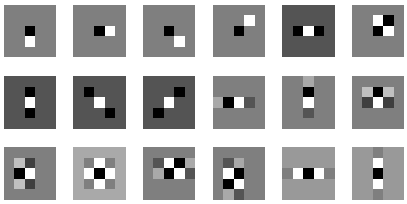


Figure 3. The set of filters used for the model referred to in this paper as the Robust Derivative (RD) model. These filters include a complete set of first, second, and third derivatives.

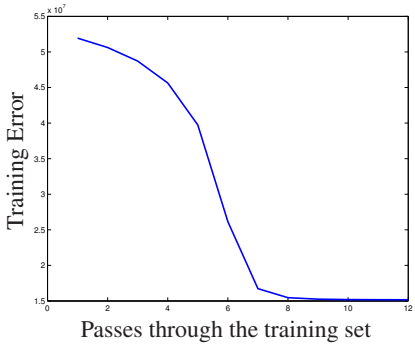


Figure 4. This plot shows the training error after each pass of the stochastic gradient descent algorithm through the data. The error stopped decreasing significantly after eight passes.

Besides demonstrating the effectiveness of the algorithm, the results from the model with a fixed set of filters, which we will refer to as the Robust Derivative model, will also provide a new way of describing the underlying mechanism that the Field of Experts model relies on. In [13], Roth and Black point out that filters in the Field of Experts model lack "the clearly interpretable structure of the filters learned using the standard PoE [Products of Experts] model." Section 6.2 discusses how the similarity between the results from this model and the Field of Experts model suggests that the Field of Experts model can be understood as imposing piecewise continuity on the image.

6.1. The Robust Derivative Model

Similar to Equation 5, we define the conditional density, $p(\mathbf{x}|\mathbf{y};\alpha,\beta)$, of the latent image $\mathbf{x}$ given the observation $\mathbf{y}$, using parameters α and β), to have the form

$$p(\mathbf{x}|\mathbf{y};\alpha,\beta) \propto \exp\left(-\sum_{r,c}\left(\mathbf{x}(r,c)-\mathbf{y}(r,c)\right)^2 - \sum_{i=1}^{N_f}\alpha_i \sum_{r,c}\rho\left((\mathbf{x}*f_i)(r,c),\beta_i\right)\right) \tag{25}$$

where $\rho(x,\beta)$ is

$$\rho(x,\beta) = \log\left(1+(\beta x)^2\right) \tag{26}$$

In this model, the filters, $f_1 \ldots f_{N_f}$ are not listed as parameters of the model. Instead, we fix the filters to be a set of derivative filters. These filters are shown in Figure 3. These filters include a complete set of first, second, and third derivatives – which leads to the name Robust Derivative model.

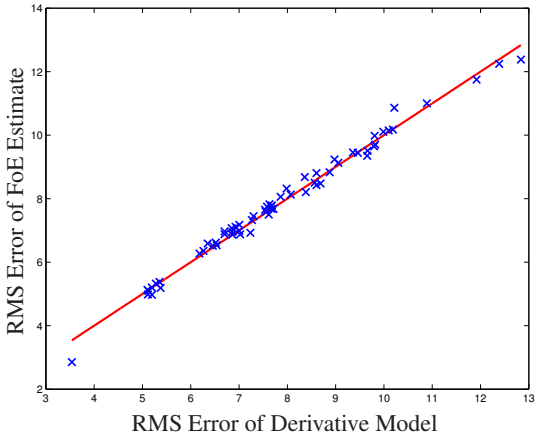


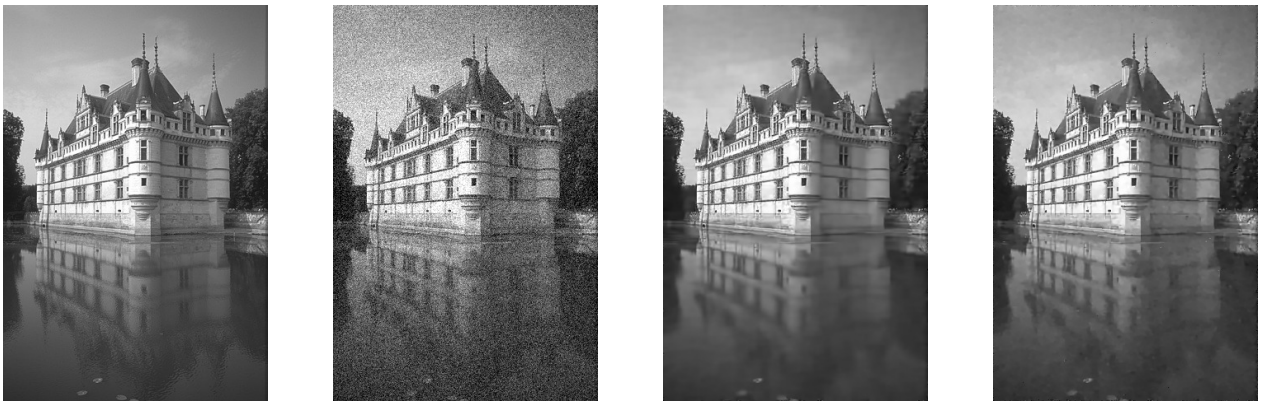
Figure 5. This scatterplot shows the RMS error in each of the 68 test images when denoised with either the Field of Experts (FoE) model or the Robust Derivative model described in this paper. The Robust Derivative model outperforms the FoE model for all images where the point is above the red line. As this figure shows, the results between the two models are comparable.

The parameters in the Robust Derivative model are the weights associated with each filter, $\alpha_1 \ldots \alpha_{N_f}$, and the β parameters. As shown in Figure 1, the Lorentzian error function is convex near $x = 0$ and becomes non-convex as x increases. In essence, each β parameter controls the range of filter responses that fall in the convex portion. As β becomes smaller, the model will behave more like a Gaussian MRF. As β grows the MRF's behavior becomes more non-Gaussian.

The parameters α and β were found by minimizing the loss function in Equation 18. Specifically, we minimized the squared-difference between the ground-truth images and the training images. For training, we used 400 27×27 image patches. The noisy observations were created by adding noise with a standard deviation, σ, of 15. Rather than using the steepest-descent algorithm to optimize the loss function, we instead used the stochastic gradient descent algorithm. When optimizing with stochastic gradient descent, the parameter vector is updated by computing the gradient for a small subset of the data. This allows updates to be computed much more quickly. Vishwanathan et al. have found the stochastic gradient descent algorithm to work well for training Conditional Random Fields [20]. In the implementation used for the results in this section, each update was computed using two training examples.

Sixteen iterations of the variational optimization scheme were used when computing the image estimates during training. In other words, the loss was computed using $\mathbf{x}^{16}(\theta)$. As Figure 4 shows, the training stopped decreasing after eight iterations. Training the model was very efficient – eight iterations took 3.6 hours on a *single* 2.6GHz Pentium IV workstation.

To test the quality of the model, we tested with the same images used in [13]. For comparison, we also denoised the images using the Field of Experts implementation provided by Roth and Black. When running the Field of Experts code, we used a step-size of 0.6 and 5000 iterations. We



(a) Original (b) Noisy Observation (c) Results using FoE Model (d) Results using Derivative Model

Figure 6. An example of denoised images produced by the Robust Derivative model described in this paper and the Field of Experts model from [13]. The results are quite similar, though the Robust Derivative model tends to capture slightly more high-frequency texture. This similarity suggests that the Field of Experts model is using the same heuristics and cues as the Robust Derivative model, which can be understood as imposing piecewise continuity on the image.

(a) Original (b) Noisy Observation (c) Results using FoE Model (d) Results using Derivative Model

Figure 7. These images show a second example comparing the results from the Field of Experts model and the Robust Derivative model. Again, the results are very similar, though the results from the Robust Derivative model are slightly sharper.

used a large number of iterations to get as close as possible to a true local maximum under the model. This enabled us to compare how well the two models capture the statistics of natural images. Denoised estimates were produced in a similar fashion using the Robust Derivative model.

Figure 5 shows a comparison of the RMS error in the estimate produced by each model over all the training images. In this figure, the Robust Derivative model produces better estimates for all those images where the plotted point falls above the red line. As can be seen in Figure 5, the two models perform quite similarly.

Examining the images in Figures 6 and 7 shows that the results are also qualitatively very similar. The primary difference is that the Robust Derivative model tends to bring out more of the fine detail in the images.

6.2. Using the Robust Derivative Model to Understand the Field of Experts Model

The similarity in the denoised estimates produced by the two models is significant because it provides insight into the computational mechanism underlying the Field of Experts model. The Robust Derivative model, from Equation 25, can also be expressed as an outlier process [3]. An outlier process can be expressed as

$$J(\mathbf{x}) = \sum_{<x_i,x_j>} z_{<i,j>}(x_i - x_j)^2 + \Psi(z_{<i,j>}) \qquad (27)$$

where the sum is over all pairs of neighboring pixels in $\mathbf{x}$. The quadratic term $(x_i - x_j)^2$ will cause outlier pairs to dominate the cost, but their influence can be trimmed by the variable z, where $0 \leq z \leq 1$. Each variable $z_{<i,j>}$ can be thought of as an indicator variable that decides whether or not to impose the quadratic penalty. The penalty function $\Psi(z_{<i,j>})$ prevents all $z_{<i,j>}$ from going to zero.

In [3], Black and Rangarajan show how the energy function in the exponent of Equation 25 can be expressed as an outlier process, similar to Equation 27. Using the outlier-process view, the Robust Derivative model can be viewed as imposing higher-order piecewise continuity on the estimated image. The continuity is piecewise because $z_{<i,j>}$ can go to 1 across edges and no continuity is enforced.

The similarity between the results from the Robust Derivative model and the Field of Experts model suggests that that the Field of Experts model can also be viewed as imposing piecewise, higher-order continuity on the estimated image.

6.3. Learning Derivative Filters

The same learning algorithm can also be used to learn the derivative filters themselves. Using the same method described above, we trained a model identical to the Field of Experts model. In this model, each filter is expressed as the combination of a set of 5×5 basis filters. While

computing the gradient, almost 80% of the computation was in computing the matrix products, $F^T W F$ from Equation 12. The matrices corresponding to these filters are much less sparse than the matrices corresponding to the derivative filters, so computing the gradients took significantly longer.

Evaluated on the training set described above, the denoised estimates found using these filters and weights had an average RMS error of 8.07 and a mean PSNR of 30.25 dB. These results are comparable to those produced using the FoE model trained by Roth and Black. Using the parameters from Roth and Black, the average RMS error was 7.86 and the mean PSNR was 30.49 dB.

When denoising full-size images using these learned filters, we found it necessary to increase the weight of the quadratic term that penalizes the difference between the estimated image and the observed image. Given the much larger set of parameters to optimize over, compared to the Robust Derivative model, the filters are likely overly specialized for denoising 27×27 images. Utilizing computing clusters to optimize the model on larger images would likely lead to significantly improved results.

7. Conclusion

We have introduced a new method for learning the parameters of continuous-valued Markov Random Fields. Variational Mode Learning is relatively easy to implement and is based on computing exact derivatives of a loss function. This enables it to inherit convergence properties from the optimization algorithm used to train the parameters.

We have also shown how the parameters can be found efficiently using stochastic gradient descent and generate results that are comparable to those from the Field of Experts model. In particular, we have used Variational Mode Learning to train a derivative-based model to denoise images. The results from this model are quite similar to those from the Field of Experts model, suggesting that the Field of Experts model can be interpreted as imposing piecewise continuity on the image.

References

[1] Y. Altun, I. Tsochantaridis, and T. Hofmann. Hidden markov support vector machines. In *The Twentieth International Conference on Machine Learning (ICML-2003)*, 2003.

[2] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng. Discriminative learning of markov random fields for segmentation of 3d scan data. In *International Conference on Computer Vision and Pattern Recognition (CVPR05)*, San Diego, CA, June 2005.

[3] M. J. Black and A. Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *International Journal of Computer Vision*, 19(1):57–92, July 1996.

[4] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, Massachusetts, 1987.

[5] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[6] M. E. Carreira-Perpignan and G. E. Hinton. On contrastive divergence learning. In *Artificial Intelligence and Statistics (AISTATS)*, Barbados, 2005.

[7] D. Geman and G. Reynolds. Constrained restoration and the recovery of discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(3):367–383, March 1992.

[8] G. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(7):1771–1800, 2002.

[9] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. In M. I. Jordan, editor, *Learning in Graphical Models*. MIT Press, Cambridge, 1999.

[10] Y. LeCun and F. Huang. Loss functions for discriminative training of energy-based models. In *Proc. of the 10-th International Workshop on Artificial Intelligence and Statistics (AIStats'05)*, 2005.

[11] A. Levin and Y. Weiss. Learning to combine bottom-up and top-down segmentation. In *European Conference on Computer Vision (ECCV)*, Graz, Austria, May 2006.

[12] A. Rangarajan, R. Chellappa, and B. S. Manjunath. Markov random fields and neural networks with applications to early vision problems. In I. K. Sethi and A. K. Jain, editors, *Artificial Neural Networks and Statistical Pattern Recognition: Old and New Connections*, pages 155–174. Elsevier Science Press, 1991.

[13] S. Roth and M. Black. Field of experts: A framework for learning image priors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 860–867, 2005.

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

[15] M. F. Tappen, W. T. Freeman, and E. H. Adelson. Recovering intrinsic images from a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9):1459–1472, September 2005.

[16] M. F. Tappen, C. Liu, E. H. Adelson, and W. T. Freeman. Learning gaussian conditional random fields for low-level vision. In *International Conference on Computer Vision and Pattern Recognition (CVPR07)*, 2007.

[17] M. F. Tappen, B. C. Russell, and W. T. Freeman. Efficient graphical models for processing images. In *Proceedings of the 2004 IEEE Conference on Computer Vision and Patt ern Recognition*, volume 2, pages 673–680, 2004.

[18] B. Taskar, S. Lacoste-Julien, and M. Jordan. Structured prediction via the extragradient method. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*. MIT Press, Cambridge, MA, 2006.

[19] D. Terzopoulos. Regularization of inverse visual problems involving discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:413–424, 1986.

[20] S. Vishwanathan, N. Schraudolph, M. Schmidt, and K. Murphy. Accelerated training of conditional random fields with stochastic meta-descent. In *International Conference on Machine Learning (ICML '06)*, 2006.

[21] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Transactions on Information Theory*, 2005.

[22] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, July 2005.

[23] J. Yedidia, W. T. Freeman, and Y. Weiss. Generalized belief propagation. In T. K. Leen, H. G. Diettrich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 689–695, 2001.