Abstract—Multiscale representations of images have become a standard tool in image analysis. Such representations offer a number of advantages over fixed-scale methods, including the potential for improved performance in denoising, compression, and the ability to represent distinct but complementary information that exists at various scales. A variety of multiresolution transforms exist, including both orthogonal decompositions such as wavelets as well as nonorthogonal, overcomplete representations. Recently, techniques for finding adaptive, sparse representations have yielded state-of-the-art results when applied to traditional image processing problems. Attempts at developing multiscale versions of these so-called dictionary learning models have yielded modest but encouraging results. However, none of these techniques has sought to combine a rigorous statistical formulation of the multiscale dictionary learning problem and the ability to share atoms across scales. We present a model for multiscale dictionary learning that overcomes some of the drawbacks of previous approaches by first decomposing an input into a pyramid of distinct frequency bands using a recursive filtering scheme, after which we perform dictionary learning and sparse coding on the individual levels of the resulting pyramid. The associated image model allows us to use a single set of adapted dictionary atoms that is shared—and learned—across all scales in the model. The underlying statistical model of our proposed method is fully Bayesian and allows for efficient inference of parameters, including the level of additive noise for denoising applications. We apply the proposed model to several common image processing problems including non-Gaussian and nonstationary denoising of real-world color images.

Index Terms—Dictionary learning, sparse coding, multiscale image processing, Bayesian statistical modeling
# TABLE OF CONTENTS

1. INTRODUCTION 01
2. DICTIONARY LEARNING IN THE LAPLACIAN PYRAMID DOMAIN 02
3. STATISTICAL MODEL 09
   - A. Extending the Noise Model Using Gaussian Mixtures 09
4. RESULTS 12
   - A. Learning Multiscale Image Features 09
   - B. Efficiency of Representation 09
   - C. Gaussian Denoising 09
5. REMOVAL OF NON-STATIONARY GAUSSIAN & NON-GAUSSIAN NOISE 13
6. CONCLUSION 14
7. ACKNOWLEDGMENT 14
8. REFERENCES 14
1. INTRODUCTION

Multiscale methods are now widely used in image processing. Notable examples include wavelets [1], [2], multiscale geometric transformations [3], [4], steerable pyramids [5], quadrature mirror filters [6], the empirical mode decomposition [7], and the Laplacian pyramid [8]. Generally speaking, these methods aim to improve upon the pure spatial frequency analysis techniques of the (two-dimensional) Fourier transform by providing resolution in both spatial frequency and spatial location [2]. Such multiscale transforms have been successfully applied to a number of image processing problems such as denoising, compression, and feature extraction [8]–[14].

Multiscale transforms typically produce decompositions using a fixed set of analysis and synthesis (i.e., basis) functions. As an alternative, dictionary learning and sparse coding methods provide a spectrum of models for learning a set of basis functions that are adapted to the data and capable of providing sparse representations of inputs [15]–[17]. Such sparse and often highly overcomplete learned representations have the capability to provide improved robustness in the presence of noise and have been shown to be useful in a wide range of applications, including denoising, inpainting, and feature extraction for classification [17]–[22]. Such models are however typically defined only at a single spatial scale. The aim of this paper is to address this deficit.

While there have been previous attempts to merge the ideas of multiresolution image representations with dictionary learning and sparse coding, many of these methods have some unfortunate drawbacks. Of particular note is the multiscale K-SVD algorithm of [23], [24], which learns separate dictionaries in the image domain using dictionary atoms of different sizes (i.e., dimensionality) in order to capture increasingly large spatial patterns. Unfortunately, the increase in size entails a concomitant increase in expected running time of the algorithm. Moreover, this approach does not take advantage of the fact that coarse-scale information (at sufficiently low spatial frequency) may be more efficiently represented using basis functions of smaller size by subsampling the image. While this is generally accomplished by an appropriate partitioning of the image into various scales, combined with subsampling (this could allow for a dictionary or dictionaries with atoms of a single size, say, 8 × 8 pixels, to be used to represent information at multiple scales), it is not clear how the approach suggested in [23] and [24] could be adapted to account for this, since it does not admit such an explicit partitioning of spatial frequency information.

The work of [25] is an attempt to improve on this concept by performing dictionary learning directly in the domain of the wavelet transform, taking advantage of the frequency selectivity of the individual levels of a wavelet pyramid to remove redundancy in the learned representations. However, this approach also has drawbacks. In particular, it requires that separate dictionaries be learned for the directional subbands in the wavelet transform of an image (or else risk a lack of representational efficiency by learning a joint dictionary not particularly well suited to any individual orientated subband). There is also the problem of a lack of visual interpretability of learned features due both to the asymmetric filtering process used to create the directional subbands as well as to the aliasing introduced by subsampling of the wavelet coefficients. Nonetheless, this approach has the advantage that it is critically sampled, which improves the efficiency of this algorithm over the model presented in [23], [24].

Other previous work has focused on creating an adaptive wavelet transform whose “wavelet” functions (i.e., band-pass filters) are learned from the data using a convolutional sparse coding model [26]. One downside of this approach is that while it overcomes the necessity of specifying a transform in advance (since the transform itself is learned), the model becomes overcomplete when using more than three learned band-pass filters. This means that if only a small number of bands is chosen to limit excessive computational burden, the learned filters are unlikely to be highly selective for particular image structures in the way that a learned dictionary trained on image blocks can be.
The work presented here is an attempt to overcome some of these limitations and to create an adapted, multiscale dictionary that is shared across all spatial scales in the representation of an image and whose construction has the same asymptotic computational complexity as single-scale dictionary learning algorithms. In particular, we present a fully Bayesian statistical approach to multiscale dictionary learning and sparse coding in the transform domain of the Laplacian pyramid [8] that is a principled extension of single-scale methods and that allows sharing of a single dictionary across scales. The model we present is an example of transform-domain dictionary learning, as suggested in [27], though our goal is specifically to address the joint problems of learning a multiscale dictionary and sharing this dictionary across scales, rather than considering arbitrary transformations. We demonstrate the usefulness of this model for image representations and for denoising grayscale images, and we present a number of extensions to the model that achieve superior performance in denoising realworld color images over established methods.
To formalize our model, we consider the problem of dictionary learning in the Laplacian pyramid. The Laplacian pyramid transform of an image is defined by the following recursive filtering scheme: letting \( G_0 = I \), where \( I \) is the input image, the \( l \)th scale in the Laplacian pyramid is defined as:

\[
L_l = G_l - [G_{l+1} \uparrow 2] \odot h,
\]

for

\[
G_l = [G_{l-1} \odot h] \downarrow 2,
\]

where \( h \) is a lowpass filter, \( \odot \) denotes (two-dimensional) convolution, and \( \uparrow, \downarrow 2 \) denote up- and downsampling by 2, respectively (in this case, with respect to both the rows and the columns). The dimensions of each successive level of the pyramid \( L_l \) are half as large as the previous level’s, and each level captures spatial frequency information in the image in a band whose center frequency is half that of the previous level’s. The original image \( I \) can be reconstructed simply by inverting the operations given above (i.e., expanding each \( G_{l+1} \) and adding it to \( L_l \)). The “Barbara” test image, along with its Laplacian pyramid transform, are depicted in Figure 1.

Because of the subsampling that takes place to generate the individual levels of the pyramid, this representation achieves an approximate normalization of the spatial scales that exist within an image. Consequently, we might hypothesize that because similar structures exist at various scales in natural images, a single adapted dictionary would be appropriate for representing information in the Laplacian pyramid transform of an image. Furthermore, because of the relationship that exists between scales, the Laplacian pyramid is uniquely suited for learning such a multiscale dictionary. Each level of the Laplacian pyramid \( L_l \) is made up of a coefficient image, and we consider the \( \sqrt{n_x} \times \sqrt{n_y} \)-pixel blocks that make up each level in a transform with \( N \) scales. Let \( x^{lp} \in \mathbb{R}^{n_x} \) be a vectorized block of transform coefficients at some scale \( l \) centered at pixel position \( p = (i, j) \). The goal of sparse coding is to find a representation \( w^{lp} \in \mathbb{R}^K \) in terms of a dictionary \( D \in \mathbb{R}^{n_x \times K} \) that (approximately) minimizes a cost function similar to the following:

\[
\|x^{lp} - Dw^{lp}\|_2^2 \text{ sub} \text{ to } \|w^{lp}\|_0 \leq T,
\]

where the 0-pseudonorm counts the number of nonzero values in \( w^{lp} \) and \( T \) is some integral threshold chosen in advance [15]. Other forms of sparse coding that emphasize different characteristics of \( w^{lp} \) are also common [15], [16]. At the same time, we adapt \( D \) to all the blocks \( x^{lp} \) for \( l = 0, \ldots, (N - 1) \) across the levels of the pyramid, thus obtaining a learned representation that accounts for image structure across all scales in the pyramid. Having coded all the blocks \( x^{lp} \) across the pyramid, we may then invert the transform to obtain a new image (e.g., for denoising purposes). If we choose to consider overlapping coefficient blocks, then we simply average the contribution of each block’s coordinates to the corresponding level of the pyramid.
As in a wavelet or steerable pyramid representation of an image, it is clear that residual correlations exist among transform coefficients in the Laplacian pyramid representation [10], [28], [29]. Our goal is to model these with an adapted dictionary that is suited to represent image information across scales. Consequently, instead of coding individual image blocks, we seek to find sparse representations of blocks of transform coefficients.

Since the Laplacian pyramid is not a unitary transform, it is important to consider the effect that minimizing the function in Equation (1) in the transform domain has on the representation error in the image domain. Because the Laplacian pyramid is an invertible linear transformation, we can represent both its forward and inverse transforms in terms of operators $A$ (the forward or analysis operator) and $S$ (the inverse or synthesis operator). Let $\tilde{I}_A$ represent the (vectorized) version of the pyramid in the transform domain formed using the coded blocks $Dw^{l,p}$ across all scales and positions. Thus, given $\tilde{I}_A$ and $S$, we form our reconstructed image $\tilde{I}$ as

$$\tilde{I} = S \tilde{I}_A = I + \epsilon,$$

where $I$ is a vectorized version of the image $I$. The vector $\epsilon$ thus represents the (additive) error of the approximate image $\tilde{I}$ relative to the original. Similarly, we can consider the approximation error directly in the transform domain by applying the forward operator to $I$

$$\epsilon_A = AI - \tilde{I}_A.$$

Our goal is to understand the relationship between $\epsilon$ and $\epsilon_A$. The norm of the induced error in the image domain is

$$\| \epsilon \|_2 = \| S \epsilon_A \|_2 \leq \| S \|_2 \| \epsilon_A \|_2,$$

where $\| S \|_2$ indicates the operator 2-norm of $S$. Without placing further assumptions on $S$ or on $\epsilon$, we cannot improve this bound. However, if we model the transform-domain error $\epsilon_A$ as a zero-mean Gaussian random vector with isotropic covariance matrix, we can make a more concrete statement about the expected error in the image domain. In particular, consider two zero-mean Gaussian random vectors $\epsilon_{A,1} \sim N(0, \sigma_1 I)$ and $\epsilon_{A,2} \sim N(0, \sigma_2 I)$, where $\sigma_1 < \sigma_2$ and $I$ is the identity matrix. Consequently, $E[\| \epsilon_{A,1} \|_2^2] < E[\| \epsilon_{A,2} \|_2^2]$, where $E[\cdot]$ is the expected value operator. Under this assumption, it is straightforward to show that

$$E[\| S \epsilon_{A,1} \|_2^2] < E[\| S \epsilon_{A,2} \|_2^2].$$

Thus, in expectation, larger errors in the transform domain equate with larger errors in the image (analysis) domain and vice versa.

We these concepts in mind, we can approach the problem of multiscale dictionary learning in the Laplacian pyramid domain. In the following, we define the model used for learning both sparse representations for the coefficient blocks $x^{l,p}$ as well as the dictionary $D$. We take a fully Bayesian approach to this problem, rather than looking for locally optimal point solutions to Equation (1). Although typically this would achieve faster convergence, it often places a number of restrictive assumptions on the representations (e.g., that each representation $w^{l,p}$ should have exactly $T$ nonzero values) in addition to requiring a *priori* knowledge of important parameters for a number of applications, (e.g., noise level in denoising). The Bayesian perspective enables us to learn these parameters, as well as the model’s complexity, directly from the data. Note that in all of our experiments below, we use a 5-tap “binomial filter” with coefficients $[\sqrt{\frac{1}{16}} 1 4 6 4 1]$ to generate the Laplacian pyramid representation of each input. We use the same filter for approximation and interpolation [8], [30].
Equation (1) specifies the relationship between the learned dictionary atoms, (sparse) weights, and the blocks of transform coefficients $x_{l,p}$. In order to find optimal values for model parameters (or, more specifically, to determine posterior distributions of the model parameters), we appeal to a Bayesian statistical model for the dictionary learning problem known as beta process factor analysis (BPFA) [17]. Recently, Bayesian statistical models for sparse factor analysis have been successfully applied to a number of image processing problems [17], [18], [31], [32], demonstrating state-of-the-art results in problems such as denoising and inpainting.

In the BPFA model, we represent each coefficient block $x_{l,p}$ as

$$x_{l,p} = D(z_{l,p} \odot w_{l,p}) + \#_{l,p}, \quad (2)$$

Where as before $D$ represents the dictionary and $w_{l,p}$ is a set of weights that describes the association of the inputs with $D$. We use $\odot$ to denote the Hadamard product, and $z_{l,p}$ is a vector of isotropic Gaussian errors. We have introduced a set of binary feature variables $z_{l,p}$ that induce sparsity in the representation $z_{l,p} \odot w_{l,p}$ of each input by forcing some values of this product exactly to zero. In the traditional factor analysis setting, the learned columns of $D$ describe the covariance structure that exists in the data, and the variables $w_{l,p}$ have an interpretation as a set of latent factors or causes that generate the observed data $x_{l,p}$ via $D$. A similar interpretation exists here, except that only a subset of the columns of $D$ define the covariance structure for a particular datapoint.

We specify the statistical model in terms of the prior distributions we place on model parameters (as in [17]). Our model may be specified – omitting from the lefthand side any additional parameters on which each parameter is conditioned for notational simplicity – as

$$l_{l,p} \sim N(0, \tau^{-2} I_{n0}), \quad w_{l,p} \sim N(0, \tau^{-2} w), \quad d_k \sim N(0, \tau^{-2} D), \quad \#_k \sim \text{Bernoulli}(\#_k), \quad \#_k \sim \text{Beta}$$

where $N(m, v)$ denotes the Gaussian probability density function with mean $m$ and (co)variance $v$, $d_k$ denotes the columns of $D$ and $I_{n0}$ denotes the order-$n_0$ identity matrix. The hyperparameters $a = 1K$ and $b = (K−1) K$ for the prior distribution on feature probabilities $\#_k$ are chosen to induce sparsity in feature usage (by placing most of the mass of the prior distribution of $\#_k$ near zero). This Beta-Bernoulli prior distribution over binary latent features corresponds to a truncated version of a stochastic process in which $K$ is assumed to be unbounded and encourages features to be shared among inputs [17], [33]. In this process, the choices of $a$ and $b$ affect both the amount of sharing that occurs and the overall sparsity level [17], [33], [34]. Note further that, because we include the binary variables $z_{l,p}$, the model is capable of enforcing sparsity by setting certain feature values exactly to zero.

We also place uninformative Gamma prior distributions on the precisions (inverse variances) in the model:

$$\tau^{-2} \sim \text{Gamma}(s!, r!), \quad \tau^{-2} w \sim \text{Gamma}(sw, rw), \quad \tau^{-2} D \sim \text{Gamma}(sD, rD),$$

### 3. STATISTICAL MODEL
where the variables $s$ and $r$ above are the shape and rate hyperparameters of the respective Gamma distributions. In our experiments, unless otherwise noted, we set each of the hyperparameters for all Gamma distributions to $10^{-6}$. Note that we may choose from a number of other possible prior distributions for the feature weights $w_{i,j,p}$ (e.g., Laplace distribution [35]) or for the noise model for $l_i,j,p$. Some of these will be described below as they relate to our experiments.

Inference of parameters in this model is accomplished by iteratively sampling from the conditional distributions of the model parameters using Gibbs sampling, as outlined in [17], [36]. We can then use a reliable sample of these distributions to reconstruct the inputs $l_i$ or treat the learned latent representations $z_{i,j,p} w_{i,j,p}$ as feature vectors. Note that in our experiments we typically do not include the low-pass residual in the dictionary learning process, as the information it contains is an approximation to the original input, rather than a difference of approximations, and it is not clear that it should be represented using the same sparse set of learned features. We use Gibbs sampling inference [36], [37] to infer posterior distributions over model parameters, and the relevant sampling equations can be found in the supplementary materials.

Note that since the dynamic range of the frequency bands of the Laplacian pyramid varies, we typically impose a separate weight variance for each scale in the model. That is, the assumption we make about the prior distribution of the individual latent feature values is that

$$w_{i,j,p} \sim N(0, \sigma^2 w_{i,j,p})$$

where $\sigma^2 w_{i,j}$ is a scale-specific weight variance.

Furthermore, in the context of denoising, we must account for varying levels of noise induced by the forward transform (because it is not a unitary operator). Thus we typically also impose separate error variances for each scale in the model, so that

$$l_i,j,p \sim N(0, \sigma^2 l_{i,j,p})$$

Because the number of scales is typically small (usually four in our experiments), there is very little extra cost associated with performing inference for these additional parameters, and the results are typically superior when including these assumptions in the model specification. In all our experiments below, we use this formulation unless otherwise noted.

Finally, we note that although there are obvious advantages to taking a statistical (i.e., model-based) approach to dictionary learning, the concept of transform-domain dictionary learning imposes no particular constraints on the algorithm used to learn $D$ and the sparse representations $z_{i,j,p} w_{i,j,p}$. One obvious advantage of the approach outlined here is that it allows the noise level(s) to be automatically inferred, rather than specified in advance (see e.g., [17]). Additionally, the tools of statistical inference allow us to extend this model in straightforward ways to add representational richness.

### A. Extending the Noise Model Using Gaussian Mixtures

In real-world imaging applications, noise statistics often do not match the simple assumptions commonly made by denoising models. In particular, the covariance structure of observed noise is often not isotropic Gaussian nor stationary in space or time (see e.g., the images in Figure 5). Consequently, models that assume isotropic Gaussian errors tend to overfit the observed noise in images. Furthermore, multiresolution image transformations provide a means of separating noise into approximately distinct frequency bands, enabling a more natural handling of noise with non-white spectral characteristics than single-scale dictionary learning methods. By taking a Bayesian approach to dictionary learning, we can naturally add complexity to the multiscale dictionary learning model that allows for modeling non-stationary and non-Gaussian noise structure in images.
In particular, instead of assuming that our observations are corrupted with isotropic Gaussian noise, we assume that \( l, j, p \) follows a Gaussian mixture distribution. By doing so, we are able to automatically “cluster” the individual coefficient blocks according to their noise characteristics. Our model changes in that we now assume that

\[
l, p | c_l, p \sim N(0, \sigma^2, c_l, p) \\
\sigma^2 | k \sim \text{Gamma}(s, r) \\
c_l, p \sim \text{Discrete}(1, \ldots, K) \\
(1, \ldots, K) \sim \text{Dirichlet}
\]

where \( c_l, p \in \{1, \ldots, K\} \) indicates the mixture component to which each coefficient block is assigned (assuming \( K \) components).

Conditioned on the assignments \( c_l, p \), the model is capable of capturing non-stationary noise that is assumed to be isotropic and Gaussian within a block, but that (possibly) varies block by block. This enables the modeling of noise sources that vary in space (or channel or scale), while at the same time providing robustness of direct estimation of those parameters from the data. This represents a particular advantage over non-statistically based dictionary learning and sparse coding algorithms since the Bayesian framework allows us to avoid specifying the noise parameters a priori, even when we assume more than a single noise component.

Inference of the additional parameters is similar to that used in the previous model, since, conditioned on the component assignments, the variables have the same posterior distributions, except that only a subset of the data will affect the inference of each of the mixture model-specific parameters (see e.g., [37]). Gibbs sampling equations for the mixture model parameters can be found in the supplementary materials. For denoising applications, we found that setting \( K \) to a large value (e.g., between 5 and 10) produced good results (see below). We further note that, while we do not explicitly take advantage of this fact in our denoising experiments, the Bayesian framework allows us to also place a nonparametric prior on the component assignment probability vector \( \pi \) known as the Dirichlet process [38]. In doing so, we are able to not only infer the noise levels \( \sigma^2, k \) but also the number of components \( K \).

Preliminary experiments showed very good performance in associating noticeable visual artifacts and noise with higher noise components using the Dirichlet process prior. However, this approach adds significant computational complexity to the inference phase, and the finite model defined above approximates a Dirichlet process prior when we set \( K \) large enough so that some components will not be used. Once we have inferred parameters, we can simply remove any unused components from the model. In practice, we found that this produced results which were essentially as good as the Dirichlet process mixture model. In other work [39], we show that the Dirichlet process mixture has distinct advantages over the finite mixture model in other applications beyond denoising.
4. RESULTS

The results of applying the multiscale dictionary learning model (MSDL) to analyzing and representing digital images as well as to artificial and real-world denoising problems are presented below. We compare the proposed multiscale model to single-scale BPFA, K-SVD, wavelet, and steerable pyramid representations of images in our denoising experiments.

A. Learning Multiscale Image Features

We begin the discussion of results by examining the learned multiscale representations of digital images. Figure 1 shows the “Barbara” (512 × 512 pixel) test image [17], along with its Laplacian pyramid representation (with four scales), and one sample of the learned multiscale dictionary. As can be seen, the learned dictionary atoms capture oriented structures in the image, and 251 out of 256 features were used at least once at each scale to represent the image.

The contribution of several multiscale features selected from the dictionary is shown in Figure 2. In the top portion of the figure are depicted enlarged versions of the eight features with highest posterior probability $\pi_k$ in representing the image in Figure 1. In order to understand the contribution that each feature makes to the ultimate image representation, we can consider reconstructing the image with an individual feature active across the scales in the pyramid. Using only this one feature, we then invert the transformation to produce an imagedomain representation of the contribution of the individual feature. Beneath each feature in Figure 2 is shown its imagedomain representation. We then display at the bottom of the figure the sum of the contributions of the eight most probable features. Clearly, the model is capable even with this small amount of features of representing a significant amount of the detail in the image at multiple scales.

B. Efficiency of Representation

The primary goal of dictionary learning is to capture correlated local structures in signals. We may define representational efficiency through a measure of the image information contained in one nonzero coefficient in the set of representations $\{z_l,p \# w_l,p\}$ across all scales and positions. Multiscale dictionary atoms contain information about many pyramid coefficients in the same way that a single-scale dictionary contains information about image pixels. This suggests that encoding an image using the latent variables $\{z_l,p\}$ and $\{w_l,p\}$ will result in a lower reconstruction error than that obtained for the same number of nonzero coefficients from a wavelet or a single-scale dictionary learning-based representation. Our results suggest that while both single- and multiscale dictionary learning methods provide consistent improvements over standard orthogonal wavelet representations, the extent of these improvements is highly dependent on the image structure as well as the learned dictionary. This is particularly true in the case of single-scale K-SVD representations.
The stochastic nature of the MSDL model makes it difficult to create optimal representations with fixed numbers of nonzero coefficients. That said, we can induce greater sparsity in the learned representations by varying the hyperparameters $a$ and $b$ for the distribution of the $\lambda_k$ variables. Using a Beta$(a, b)$ distribution to model the $\lambda_k$, the expected value of each $\lambda_k$ is $a/(a+b)$, and conditioned on $\lambda_k$, $\mathbb{E}[z_{l,p} | \lambda_k] = \lambda_k$. Thus, the total expected number of nonzero values in any binary feature vector $z_{l,p}$ is

$$\lambda_k \mathbb{E}[z_{l,p} | \lambda_k] = \frac{\lambda_k}{a + b} = K \cdot \frac{a}{a + b}.$$ 

Consequently, varying the relative sizes of $a$ and $b$ gives a way to control (in expectation) the sparsity in the model by placing a more significant prior assumption that each $\lambda_k$ is likely to be very small, typically resulting in fewer nonzero entries in the $z_{l,p}$. This does not meet the exact requirement of choosing $M$ nonzero values a priori, but it is the closest analogue in this particular type of model. Nonetheless, we show that increasing the size of $b$ relative to $a$ does in fact produce sparser solutions, and the solutions degrade in quality in a manner consistent with wavelet coefficient pruning.

In order to provide a consistent means of comparison between images and to analogize the notion of compression using a fixed basis with that of compression using a learned dictionary, we first learned a dictionary to represent the test images using an independent training set of natural images from the Berkeley Segmentation Dataset [40]. We selected 512 $8 \times 8$-pixel coefficient blocks from each scale of a four-scale Laplacian pyramid representation of each of the 200 training images in this dataset to train the MSDL dictionary. A set of randomly extracted patches from the images was used to train a single-scale K-SVD dictionary. The number of features in both models was set to 512. We then inferred $D$ for 50 burn-in and 25 collection iterations (75 total iterations in the K-SVD case). For MSDL, the collection iterations were averaged to compute the final dictionary $D$, while the last iterate was used after training with the K-SVD algorithm. We then used these trained dictionaries to represent the test images in our experiments, and compared the representational efficiency of the (sparse) coded images to that of an orthogonal wavelet representation generated from a Daubechies-4 wavelet. Coefficients were pruned based on absolute value globally across scales. Note that for the dictionary learning models, we considered non-overlapping blocks.

The MSDL and wavelet representations each used three detail scales, and we kept the low-pass information “as-is” (that is, it was not coded). For the K-SVD representation, we followed common practice (see e.g., [41]) and subtracted the local mean intensity from each block before coding. For the K-SVD comparisons, these local means served as the low-pass information we compared against in the following. Because we are interested in understanding how much information is contained in the coefficients we encode, we considered only a fixed number of nonzero coefficients in excess of the low-pass information. Our task was then to determine the “gain” that this number of coefficients provided.

Figure 3 shows the graph of the gain-over-low-pass (for PSNR) in dB as a function of the number of nonzero coefficients in each of the models for the indicated test images. In all of our experiments, we set $a = 1$ and varied $b$ using the values $1 \times 10^4$, $1 \times 10^5$, $2.5 \times 10^5$, $1 \times 10^6$, $2 \times 10^6$, and $5 \times 10^6$. For each image, all relevant parameters in the MSDL model were sampled for 1000 burn-in iterations followed by 250 collection iterations. These were averaged to compute both the output representation as well as to determine the number of
nonzero coefficients NC for each value of b. This quantity was computed as the average number of nonzero coefficients over the collection samples. We then selected the same number of coefficients in the wavelet representations [25]. For the K-SVD representation, we selected \( T = \frac{NC}{NB} \) nonzero coefficients for each block, where NB is the number of blocks in the test image.

The plots in Figure 3 show that when compared with the wavelet representation, the MSDL model produced representations that achieved a larger gain in reconstruction quality for an equivalent number of coefficients across all experimental conditions. Interestingly, the shapes of the curves in each case are close to identical, suggesting that the transform coefficient selection (and pruning) that happens in the MSDL as we vary the strength of the prior assumption on the \( \ell_k \) is similar to the effect of wavelet coefficient pruning. Compared with the K-SVD representation, the MSDL model always yielded better performance for large numbers of nonzero coefficients, but for the “Lena” and “House” images, yielded worse performance for small numbers of nonzero coefficients. One explanation for this is the differing spatial frequency distributions among the test images which may have been more efficiently represented using the trained dictionary. A further reason that the K-SVD algorithm is able to outperform MSDL on some of the test images when the number of nonzero coefficients is small is that the dictionaries we learned were intended to be general, rather than highly specific to an individual image or class of images. This may explain the smaller improvement we demonstrate here over the gains shown using wavelet dictionary learning in [25].

As stated above, the BPFA statistical model does not admit a straightforward a priori restriction on the cardinality of the individual \( z_{l,p} \), nor on the collection taken as a whole. What the model does allow is some specification of the prior belief about how sparse a representation will be, and thus what the expected number of nonzero binary feature values for any particular input will be [33]. The work of [42] suggests an extension of the underlying stochastic process in the BPFA model that enables placing a restriction on the number of nonzero entries in each \( z_{l,p} \). This extension would analogize the notion of matching pursuits on individual image blocks to the Bayesian setting [15], [43]–[45]. However, the connection between global matching pursuits (or simple thresholding pruning schemes) on wavelet coefficients and global (i.e., joint) selection of nonzero entries of the collection of \( z_{l,p} \) is still lacking. This is a natural direction for future investigation.

C. Gaussian Denoising

A simple method for image denoising is easily derived from the assumptions of the model given above. Since the noise term in the model is capable of capturing both approximation error and noise, the minimum mean-square error estimate of the data is simply the posterior mean \( \mathcal{D}(z_i | w_i) \). Thus, an approximation to the true data is given by ignoring the \( \ell_{p} \) term, and so we reconstruct an approximation of the original underlying inputs by computing \( \mathcal{D}(z_i | w_i) \). In order to estimate the denoised image, we then apply the inverse transformation to the pyramid represented by the individual coefficient blocks.

We performed a number of experiments on a set of standard test images [17], [41] to assess the denoising performance of the MSDL model on denoising images corrupted with artificially added Gaussian noise. We compare the proposed method to the BPFA and K-SVD single-scale dictionary learning denoising algorithms, wavelet soft thresholding [46], and the steerable pyramid-based Gaussian scale mixture (GSM) denoising algorithm of [29]. In each experiment the dictionary learning-based models were initialized with 256 features drawn from their prior distribution (in the case of MSDL/BPFA). In the case of K-SVD we used an overcomplete DCT basis with 256 elements. In all cases, the patch size (and thus the dimensionality of the dictionary atoms) was set to 8 \( \times \) 8 pixels, and the Gibbs samplers were run for 350 burn-in iterations, after which 100 collection samples were averaged to produce the results in Table I. The K-SVD algorithm was similarly run for 350 iterations, after which 100 iterations were averaged to produce the final result.\(^{1}\) For the dictionary learning-based algorithms, we considered both non-overlapping blocks (as in [25]) and blocks that overlapped by four pixels. The difference in performance based on these

\(^{1}\)We found that this procedure worked better than using the last iterate alone to reconstruct the image, due to the fact that the dictionary continued to be adapted throughout all iterations of the algorithm. While more iterations of inference could have been used, in practice there was little difference in performance.
choices can be seen in Table I, where the results represent the average over five independent experiments. We now discuss the results in detail.

For each test image, we corrupted the pixels with independent isotropic Gaussian noise with the standard deviations indicated in Table I. For the experiments using the K-SVD algorithm, the dictionary was learned “in-place” on the noisy images [25], [41]. Furthermore, for K-SVD, the noise standard deviation was assumed known and was used to choose the error cutoff criterion in the OMP sparse coding step [15], [41], [47]. This criterion was set to the default value of 1.15 times the assumed noise level. For wavelet soft thresholding, we supplied the true noise level as the assumed noise level of the algorithm, and similarly for the GSM-based algorithm. When using the wavelet-based noise level estimation procedure of [48], the results for wavelet soft thresholding were similar, were typically slightly worse for the GSM-based model, and were significantly worse for the K-SVD algorithm, though they improved when overlap between the blocks was added. However, for K-SVD, the results were still typically up to 2 dB worse when the noise level was learned from the data.

The results in Table I indicate that for the lowest noise level, the algorithms perform similarly, but with an obvious quantitative disadvantage of using only non-overlapping blocks in the dictionary learning-based approaches. However, as Figure 4 indicates, this does not necessarily mean that the visual performance is significantly worse (or worse at all). Indeed, although the GSM-based algorithm displayed the best quantitative performance across all of our experiments, the MSDL model showed essentially equivalent qualitative performance to this algorithm, even when using non-overlapping blocks.

![Figure 4: Representative result of denoising images corrupted with added Gaussian noise using noisy version of the “Boat” test image (for the top row, $\sigma_v = 5$; for the bottom row, $\sigma_v = 10$). In these examples, the dictionary learning-based approaches used non-overlapping image blocks. (a,e) Result using proposed method. (b,f) Result using BPFA. (c,g) Result using K-SVD. (d,h) Result using Gaussian scale mixture algorithm of [29]. Note that for the higher noise level (bottom row), the single-scale BPFA/K-SVD algorithms produced results with significant blocking artifacts, which can to some extent also be seen for the case $\sigma_v = 5$. The result using the proposed MSDL model shows no presence of such artifacts, and visual results on par with the GSM-based denoising algorithm, despite learning the noise level automatically from the data.](image-url)
The MSDL model also showed significantly improved performance over BPFA and K-SVD when the noise level was higher, often showing a 1–2 dB improvement in peak signal-to-noise ratio over the other models when using non-overlapping blocks. MSDL and BPFA typically showed a modest but not significant improvement over K-SVD when overlapping blocks were used. The better results using non-overlapping blocks were partly a consequence of the significant blocking artifacts produced when using the single-scale dictionary learning algorithms. This can be seen in the bottom row of Figure 4. It is interesting to note that the MSDL algorithm does not suffer from this problem. This is due to the fact that rather than subtracting the local mean intensity from a specific block, the filtering process used to generate the pyramid subtracts a smooth approximation of the local mean intensity.

Although better performance can be achieved by adding additional overlapping blocks to all three dictionary learning based approaches, this adds significant computational cost. While the additional cost of performing a single sparse coding step may be tolerable – which might be done with a fixed basis representation, e.g., DCT, and thus involves only a single estimation phase – previous work has shown image denoising performance to be best when simultaneously learning the dictionary and performing denoising [17], [41]. Consequently, the quadratic increase in computational complexity by adding overlapping blocks quickly becomes burdensome on all but the smallest images, since a dictionary update and sparse coding step must be performed in each iteration. The MSDL model, on the other hand, is capable of overcoming the obvious visual artifacts that result when using non-overlapping image blocks in single-scale algorithms. At the same time it provides an advantage over the GSM algorithm by enabling the noise level present in the image to be inferred from the data rather than specified in advance, while achieving comparable visual performance.
5. REMOVAL OF NON-STATIONARY GAUSSIAN & NON-GAUSSIAN NOISE

We found that for difficult real-world denoising problems, the noise mixture model extension presented in Section III demonstrated significantly improved performance over the MSDL model with a single noise component and over the other models we tested. In the following experiments, we concentrate on denoising color images that are corrupted with noise and artifacts that are not stationary across space or channel and that often do not possess broad spectral properties (i.e., the noise is often concentrated at particular scales). The images shown in Figure 5 are two examples of such images and were taken with consumer digital cameras, and both possess significant visual artifacts and non-white noise.

Although dictionary learning-based algorithms often represent color images using image blocks that consist of concatenated color channels, we instead adopt a different approach that enables us to make a more consistent comparison with transform-based methods. In particular, we convert each color image to the YCbCr colorspace and operate on the luminance channel only [49]. In order to remove noise from the chroma components, we simply blur them with a 7-tap separable binomial filter. Because the luminance component contains most of the detail of the image, this process does not significantly affect the sharpness of the result (see e.g., [50]).

We use the extended MSDL model with the noise mixture model to formulate a new denoising algorithm for real-world images. The advantage of incorporating the mixture model can be seen by observing the following fact: when strong, highly non-isotropic Gaussian noise is present in an image, dictionary learning-based models, which capture the covariance structure in the data, will tend to overfit this noise, treating it as significant structure. On the other hand, adding the flexibility of the noise mixture model allows us to bias against representing data whose structure is atypical, isolating these areas by associating them with higher noise levels.

Once we have accomplished this, we can use the information provided by the mixture model to intelligently bias the prior distribution of the noise components in the model as well as the prior distribution on the zl,p. Our denoising algorithm proceeds as follows: we first learn the mixture component assignments and the component parameters on an image, after which we fix the component assignments (essentially treating this as a “true” underlying grouping of the patches). Having

![Figure 5: Real-world test images. Note the significant amount of high-frequency visual artifacts (left image) and the presence of both Gaussian-like and narrow-band noise (right image).](image-url)
fixed the assignments, we place a Gamma prior distribution on the component precisions (which we subsequently learn only from the data in each group), with shape parameter $10^{-6}$ and rate parameter chosen to be much larger than before, say between 5 and 25. This has the effect of biasing the prior toward larger noise levels. We also set the hyperparameters of the feature probabilities $\lambda_k$ to $a = 1$ and $b$ to some larger value, say 104. We then learn model parameters using these updated hyperparameters for an additional number of iterations to produce the final result.

This approach was effective for two reasons: first, biasing the noise prior distributions toward larger variances and the feature probabilities’ prior distribution toward encouraging greater sparsity increases the model’s robustness against non-Gaussian artifacts, and second, the spatial variation of the noise level enabled by the mixture model allows us to differentially denoise patches with different noise statistics. In practice, we found that for images with repeating artifacts with significant structure (e.g., the left image in Figure 5), larger values of the Gamma rate parameter and $b$ produced the best results. For images with less structured artifacts, such as the right-hand image in Figure 5, small values were often sufficient. However, in the results we present below, we used $b = 105$ for the left-hand image in Figure 5 and $b = 104$ for the right-hand image in Figure 5. In both cases, we set the Gamma rate parameter during the second phase to 25.

We applied this two-stage approach to learn spatially varying noise levels in the images in Figure 5, comparing the results of applying the MSDL+Mixture model with a mixture model-enhanced version of BPFA, the single-scale K-SVD algorithm, and the GSM-based denoising algorithm. For these real-world images, we estimated the noise level as input to the K-SVD and GSM-based models using the wavelet-based noise level estimation method suggested in [48]. However, we found that optimal visual results were obtained by significantly increasing the noise parameter $\sigma$ to $2.5 \times \sigma$, though naturally this presents a tradeoff between detail preservation and removal of visual artifacts, which we discuss in more detail below. Sections of the result of applying these algorithms to the images in Figure 5 are shown in Figure 6.

Figure 6 shows that the MSDL+Mixture model has produced results which are visually superior to the other algorithms, with good preservation of image detail while removing noticeably more visual artifacts than the other methods (see also Supplementary Figures 3–10 for a comparison of the entire denoised images). In these results, we used image patches that overlapped by four pixels for the dictionary learning-based algorithms and learned initial mixture model parameters for MSDL/BPFA for 50 iterations, followed by 50 iterations of sampling model parameters with the assignments fixed and 25 collection iterations, which were averaged to produce the final result. For the K-SVD algorithm, we learned parameters for 100 iterations, followed by averaging an additional 25 iterations.

Despite the greater patch overlap, the K-SVD results tended to look blocky and in some cases had a significant loss of detail, though many artifacts were still not successfully removed from the test images. On the other hand, the mixture model based approaches, which allow the model to adapt the noise level block by block, were largely able to remove the noticeable artifacts from the images. The BPFA+Mixture model did however produce results that had a rougher and blockier look, a problem not suffered by the MSDL model with its smooth subtraction of the local mean from the images. Similar to K-SVD, the GSM-based model was unable to remove a number of artifacts from the image (e.g., around the soup can label) and in places where the apparent SNR was high. We also include a comparison in the Supplementary Material between the MSDL+Mixture model and the result of applying the same denoising procedure in the MSDL model (with increased sparsity due to the larger value of the parameter $b$), but without the spatial adaptivity of the mixture model. Once again, we see that the sparsity alone is not enough to successfully remove the same number of artifacts from the images as the MSDL+Mixture model (cf. supplementary Figures 11 and 12).
We can conclude from these results that using a single noise parameter in the K-SVD and GSM models often meant sacrificing detail for denoising ability. On the other hand, by using the mixture model, we can concentrate removal of noise in particular areas that are strongly artifactual. Perhaps surprisingly, these areas were typically identified by the dictionary learning models using the mixture model noise term (see Supplementary Figure 1 for an example of the ability of the model to capture spatially varying noise levels in the test images). It is the Bayesian statistical framework that allows us to naturally add complexity to the model, enabling not only the addition of multiple noise sources, but also the ability to learn optimal values for these parameters directly from the data, unlike in the case of the K-SVD and GSM models.
6. CONCLUSION

We have presented a fully Bayesian model for multiscale dictionary learning and sparse coding and demonstrated its efficacy on a number of image processing problems. This model overcomes some of the limitations of prior work in that this multiscale representation admits a computationally more efficient approach to learning multiscale dictionaries than was suggested in [23], and also makes possible the sharing of a single dictionary across scales. We have furthermore shown that the model is capable of overcoming standard issues with block-based image processing algorithms (e.g., single-scale dictionary learning models), even when using nonoverlapping image blocks. It is exactly the transform that we use – the Laplacian pyramid – that provides this additional advantage. The multiscale representation also produces denoising results that are on par with, and in some cases better than, standard existing methods, while the Bayesian model formulation enables learning important parameters (such as the noise level(s)) directly from the data.

The Laplacian pyramid also appears to be the ideal representation for modeling multiscale features in images. We showed that it is capable of representing structures efficiently across scales and learns features that can be shared across scales while simultaneously capturing structures of different size. The subsampling introduced by the Laplacian pyramid transform performs the normalization for scale that is critical to sharing a dictionary. Furthermore, the lack of directional selectivity of the transform, as well as the minimal aliasing it introduces, ensure a consistency of structures across scales that is not possible in a wavelet representation. Although the applications presented here focused on lower-level processing, the sharing in the model of features across scales suggests that properties of the features may provide useful descriptions of images for higher-level image processing and computer vision tasks such as scene recognition or object detection.

Finally, we note that the above model does not in any significant way depend on the use of images as the signals of interest, so that the model extends naturally to applications in 1D signal processing. Furthermore, the noise mixture model has a number of additional applications, for example for estimating and detecting multiple noise sources in images, denoising images corrupted with structured noise sources, and image forensics applications [39].
7. ACKNOWLEDGMENT

The authors would like to thank E. Simoncelli for his pyramid toolbox [30] and wish also to thank N. Foti and A. Chakrabarti for many useful discussions.

ABOUT LGS INNOVATIONS

LGS Innovations delivers next generation solutions that solve the most complex networking and communications challenges facing the U.S. Federal Government, State and Local Governments, Foreign Governments, and commercial enterprises. LGS offers groundbreaking research and development and builds advanced wireless, optical, and wired products and applications customized for specific mission environments. These solutions provide unique information and security advantages that lead to the operational success of its customers. LGS’ offerings include:

- Campus and building networking solutions for military bases, hospitals, and corporate centers
- Maritime applications for in-port and at sea communications
- Global networks (long-haul communications, including undersea cable)
- Enterprise voice, video, and data networking
- 4G wireless deployable communications for public safety, battlefield, and emergency and first responder communities
- Network engineering, integration, and installation
- Cloud and data center infrastructure
- Video teleconferencing and IPTV suites
- Research and development in advanced multimedia/RF communications, cybersecurity, sensing technologies, and photonics

LGS Innovations is a U.S.-owned company headquartered in Herndon, Virginia, with offices in Colorado, Illinois, Maryland, New Jersey, New Mexico, and North Carolina. Formerly a subsidiary of Alcatel-Lucent, LGS is the exclusive reseller of Alcatel-Lucent products and services to the U.S. Federal Government and any other entity when the end customer is the U.S. Federal Government. LGS maintains strong ties to Bell Labs and its technologies, employing more than 450 scientists and engineers and a total of nearly 700 employees worldwide. To learn more about LGS Innovations, visit www.lgsinnovations.com.
8. REFERENCES


