Learning Dependent Dictionary Representation with Efficient Multiplicative Gaussian Process

Anonymous Author 1
Unknown Institution 1

Anonymous Author 2
Unknown Institution 2

Anonymous Author 3
Unknown Institution 3

Abstract

In dictionary learning for analysis of images, spatial correlation from extracted patches can be leveraged to improve characterization power. We propose a Bayesian framework for dictionary learning, where spatial location dependencies are captured by imposing a multiplicative Gaussian process prior on the latent units representing binary activations. Data augmentation and Kronecker methods allow for efficient Markov chain Monte Carlo sampling. We further extend our model with a sigmoid belief network, linking Gaussian processes and high-level latent binary units to capture inter-dictionary dependencies, while yielding additional computational savings. Applications to image denoising, inpainting and depth-information restoration demonstrate that the proposed model outperforms traditional Bayesian dictionary learning approaches.

1 INTRODUCTION

Learning overcomplete sparse latent representations for signal restoration and characterization has recently led to state-of-the-art results in tasks such as image denoising, inpainting, super-resolution and compressive sensing (Zhou et al., 2009; Yang et al., 2012). In dictionary learning, signals are represented in a latent factor space, where each signal is encoded as a sparse linear combination of dictionary elements (factors). Traditional approaches learn sparse codes from a fixed number of dictionary elements, while minimizing the reconstruction error subject to sparse regularization (Aharon et al., 2006; Mairal et al., 2009a). However, the sparsity level and the dictionary size are usually not known *a priori*. Non-parametric Bayesian approaches have been successful at tackling these challenges (Zhou et al., 2009, 2012; Polatkan et al., 2015), by employing methodologies like the Indian buffet process (IBP) (Ghahramani and Griffiths, 2005). Such Bayesian approaches also provide a principled way to estimate uncertainty and typically yield excellent generalization ability.

Recent work has demonstrated that modeling images as a collection of sub-regions or *patches* is important, because leveraging local image structure is instrumental for representational quality (Mairal et al., 2009b; Zhou et al., 2009). The key idea in this line of work is that similar patches are likely to share dictionary elements. Furthermore, dictionary learning can be greatly improved by imposing that patches close in space are likely to use the same or similar dictionary elements (Zhou et al., 2011). Specifically, dependent Hierarchical Beta Process (dHBP) (Zhou et al., 2011) utilized a dependent IBP to capture spatial correlations, via a 2-D smoothing function based on patch locations; the smoothing function attenuates as the distance between patches increases. Although their approach leads to dramatic performance improvements, the smoothing function has to be specified parametrically, and the posterior of their Bayesian formulation is not locally conjugate, which makes inference challenging.

Gaussian Process (GP) priors (Rasmussen and Williams, 2006) are a natural choice for capturing spatial dependencies. They are appealing because one can use them, in a principled non-parametric manner, to estimate correlation as a function of relative spatial location. However, despite great flexibility, GPs are known to be computationally expensive; in fact, they become prohibitively expensive as the number of observations grows, due to repeated matrix inversions.

In this paper, we propose a framework for dictionary learning where patch-to-patch spatial dependencies are modeled via GP priors linked to binary dictionary element activations. To address the compu-

Preliminary work. Under review by AISTATS 2016. Do not distribute.

tational challenges of GPs, we consider an efficient Kronecker inference method, with multiplicative covariance functions (Gilboa et al., 2015). Furthermore, we utilize deep Sigmoid Belief Networks (SBNs) to impose correlation structure across dictionary elements, which we demonstrate leads to performance improvements in many cases. The GPs may be linked to the binary units at the top layer of a deep SBN, and the number of these top-layer units may be made small relative to the number of dictionary elements. Therefore, there is a substantial computational savings manifested by placing the (small number of) GPs at the top of an SBN, rather than placing GP for each of the (large number of) dictionary elements.

The contributions from our framework are: (*i*) Gaussian process priors that capture spatial dependencies between patches in images; (*ii*) sigmoid belief networks that capture dependencies between dictionary elements; (*iii*) efficient inference due to Kronecker methods for GPs and a fully local conjugate Bayesian formulation based on Pólya-gamma data-augmentation; (*iv*) computational savings by reducing the number of needed GPs, by linking them to the top layer of the SBN; (*v*) results on traditional image tasks and depth restoration, that demonstrate the flexibility of our approach and highlight how learning dependency structure yields superior performance.

Related Work Several studies have employed GPs in tasks such as interpolation (Wachinger et al., 2014a), super-resolution (super-pixel) (Wachinger et al., 2014b; He and Siu, 2011) and denoising (Liu, 2007; Wang et al., 2014). These works are based on the same key insight, that images composed of noisy or *corrupted* patches can be used to produce clean images as output, by leveraging statistical correlations between image patches. For this purpose, sophisticated covariance functions were designed to characterize patch dependencies. Our work is complementary to their work in that we use GPs for location dependencies, not patch dependencies.

Other work has also explored GPs as priors for dictionary elements. Xing et al. (2012) characterized multichannel hyper-spectral images, where each channel is associated with a distinct wavelength. Their assumption is that dictionary elements from different channels are smooth as a function of wavelength, whereas we assume that dictionary activation is a smooth function of location.

Garrigues and Olshausen (2007) developed a method close to ours; they proposed a sparse coding model where spatial dependencies are imposed via pairwise coupling using an Ising model. However, their model requires a user-defined temperature parameter that uniquely controls the strength of coupling.

2 BAYESIAN DICTIONARY LEARNING

Assume observed data $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \in \mathbb{R}^{J\times N}$, where $\boldsymbol{x}_i$ represents data from one of N patches extracted from a single image. In Bayesian dictionary learning, the goal is to learn dictionary elements, $\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_M\} \in \mathbb{R}^{J\times M}$ from data, $\boldsymbol{X}$. The i-th observations, $\boldsymbol{x}_i$, is represented as

$$\begin{aligned} \boldsymbol{x}_i &= \boldsymbol{D}(\boldsymbol{w}_i \odot \boldsymbol{z}_i) + \boldsymbol{\varepsilon}_i\,, & \boldsymbol{\varepsilon}_i &\sim \mathcal{N}(0, \sigma_\varepsilon^2 \boldsymbol{I}_J)\,, \\ \boldsymbol{d}_m &\sim \mathcal{N}(0, \boldsymbol{I}_J)\,, & \boldsymbol{w}_i &\sim \mathcal{N}(0, \sigma_w^2 \boldsymbol{I}_M)\,, \end{aligned} \tag{1}$$

where $\odot$ denotes the element-wise (Hadamard) product, and $\boldsymbol{I}_J$ is a $J \times J$ identity matrix. Vectors $\boldsymbol{w}_i = \{w_{i1}, \ldots, w_{iM}\} \in \mathbb{R}^M$ and $\boldsymbol{z}_i = \{z_{i1}, \ldots, z_{iM}\} \in \{0,1\}^M$ represent weights and *binary activations*, respectively. Specifically, $\boldsymbol{z}_i$, encodes the presence or absence of dictionary elements for the i-th sample. $\boldsymbol{\varepsilon}_i$ is *i.i.d.* additive Gaussian noise. To avoid ambiguities in the relative scaling of $\boldsymbol{D}$ and $\boldsymbol{w}_i$, $\boldsymbol{d}_m$ is constrained to have unit variance. Hyperpriors are placed on the variances of $\boldsymbol{w}_i$ and $\boldsymbol{\varepsilon}_i$, via inverse-Gamma distributions on σ_w^2 and σ_ε^2, which enables us to share dispersion information across dictionary elements.

One possible approach to encourage sparsity in the binary activations, $\boldsymbol{z}_i$, is to use a Bernoulli-beta specification (Zhou et al., 2009)

$$z_{im} \sim \text{Bernoulli}(\pi_{im})\,,\ \pi_{im} \sim \text{Beta}\left(\frac{\eta}{M}, \eta\left(1 - \frac{1}{M}\right)\right)\,, \tag{2}$$

where η controls the sparsity of $\{\boldsymbol{z}_i\}_{i=1}^N$. In practice, π_{im} is marginalized out during inference. Note that the parameters of the beta distribution are set to match a beta process formulation, *i.e.*, its parameters are scaled with M. It has been previously shown that when M is large, (2) corresponds to an finite approximation of the beta process (Zhou et al., 2009).

3 DICTIONARY LEARNING WITH GAUSSIAN PROCESSES

Zhou et al. (2011) demonstrated that it is reasonable to assume that patches located near each other are likely to be represented in terms of the same or similar dictionary elements, and thus binary activations of nearby patches are likely to be consistent with spatial dependencies. We incorporate such prior belief into (1) by means of a GP on a 2-D spatial field. Our approach differs from that by Zhou et al. (2011), in that GPs allow estimation of binary activation dependencies (by connecting the GP output to a logistic link function). We call our method GP-FA (Gaussian Process Factor Analysis), for short. The construction for the m-th binary activation, z_{im}, in (2) becomes

$$z_{im} \sim \text{ Bernoulli}(\sigma(y_{im}))\,, \tag{3}$$

$$y_{im} = f_m(\boldsymbol{l}_i)\,, \qquad f_m(\cdot) \sim \mathcal{GP}(b_m, k_m(\boldsymbol{l}_i, \cdot))\,, \tag{4}$$

$$b_m \sim \mathcal{N}(\lambda_m, \sigma_b^2)\,, \qquad \lambda_m \sim \mathcal{N}_{-}(0, \sigma_\lambda^2)\,, \tag{5}$$

where $\sigma(\cdot)$ denotes the sigmoid function, y_{im} is the value of function $f_m(\cdot)$ evaluated at the 2-D spatial coordinates of the i-th patch, $\boldsymbol{l}_i = \{l_i^{(1)}, l_i^{(2)}\}$. The function $f_m(\cdot)$ is drawn from a GP with constant mean function, $\mu_m(\cdot) = b_m$, and multiplicative covariance function, $k_m(\boldsymbol{l}_i, \cdot)$, defined as $k_m(\boldsymbol{l}_i, \cdot) = k_m^{(1)}(l_i^{(1)}, \cdot) \otimes k_m^{(2)}(l_i^{(2)}, \cdot)$.

For the mean, b_m, we specify a Gaussian prior with mean and variance, λ_m and σ_b^2, respectively. To encourage sparsity in the activations, z_{im}, we bias function instances, y_{im}, towards negative values using a zero-mean Gaussian distribution truncated above zero (*i.e.*, negative support) with variance σ_λ^2. Since this prior is shared by all factors, it encourages sparsity globally. Further, the hierarchy in (5) is convenient from a practical stand point, because it yields local conjugacy.

For the covariance function, $k_m(\cdot, \cdot)$, in our implementation we consider the widely used squared exponential (SE) function. Specifically, the covariance function for axis $s = \{1, 2\}$, is defined as

$$k_m^{(s)}(l^{(s)}, l^{(s')}; \boldsymbol{\Theta}_m) = (\sigma_f^2)_m \exp\{-(l^{(s)} - l^{(s')})^2/\theta_m\}\,,$$

where $\boldsymbol{\Theta}_m = \{(\sigma_f^2)_m, \theta_m\}$ is the set of parameters for the m-th dictionary element, $(\sigma_f^2)_m$ is the signal variance and θ_m is the characteristic length scale (Rasmussen and Williams, 2006).

Note that in our covariance-function specification, we are assuming that the Gaussian process is isotropic in different spatial axes, s, provided that different dimensions share the same characteristic length scale, θ_m. This assumption may seem strong, but works well in practice (He and Siu, 2011). Since our covariance function is multiplicative and isotropic, the similarity between any two patches centered at $\boldsymbol{l}_i$ and $\boldsymbol{l}_{i'}$ is based on the Euclidean distance between their centers.

Pólya-gamma augmentation Gaussian process priors linked to binary data as in (4) have been traditionally used for classification tasks. In such a scenario, the Laplace approximation or Expectation Propagation (EP) are typically employed to approximate the non-Gaussian posterior resulting from non-Gaussian likelihoods (Rasmussen and Williams, 2006). MCMC approaches have been proposed as well (Neal, 1997), but they are known to be inefficient because the posterior distribution has to describe highly correlated variables. Here we focus on Gibbs sampling, leveraging the Pólya-Gamma (PG) data augmentation scheme of Polson et al. (2013). In contrast to probit-based augmentation, PG augmentation has been shown to be efficient with sophisticated posteriors (Gan et al., 2015), while enjoying theoretical guarantees in terms of unbiased estimates of posterior expectations (Choi et al., 2013). Briefly, and of relevance to our method, if the auxiliary variable γ is draw from Pólya-gamma distribution, *i.e.*, $\gamma \sim \mathcal{PG}(1, 0)$, the following identity holds for any ψ

$$\frac{e^\psi}{1 + e^\psi} = \frac{1}{2} e^{\frac{\psi}{2}} \int_0^\infty e^{-\frac{\gamma \psi^2}{2}} p(\gamma) d\gamma\,.$$

This identity enables one to write the joint distribution for $\boldsymbol{z}_m = \{z_{1m}, \ldots, z_{Nm}\}$, $\boldsymbol{y}_m = \{y_{1m}, \ldots, y_{Nm}\}$ and $\boldsymbol{\gamma}_m = \{\gamma_{1m}, \ldots, \gamma_{Nm}\}$ as

$$\begin{aligned} p(\boldsymbol{z}_m, \boldsymbol{y}_m, \boldsymbol{\gamma}_m | b_m) &\propto p(\boldsymbol{y}_m | b_m) p_0(\boldsymbol{\gamma}_m) \\ &\textstyle\prod_i \exp\left\{\left(z_{im} - \frac{1}{2}\right) y_{im} - \frac{1}{2}\gamma_{im} y_{im}^2\right\}\,, \end{aligned} \tag{6}$$

which is convenient because it gives rise to closed-form conditional posteriors for $\boldsymbol{z}_m$, $\boldsymbol{y}_m$ and $\boldsymbol{\gamma}_m$.

Gibbs updates for each y_{im} can be obtained by conditioning on the remaining $\boldsymbol{y}_{\backslash im} \triangleq \boldsymbol{y}_m \backslash y_{im}$. In the following discussion, we use the notation $\boldsymbol{K}$ to denote the $N \times N$ Gram matrix of the Gaussian process obtained by evaluating $k_m(\boldsymbol{l}_i, \cdot)$ at $\{\boldsymbol{l}_i\}_{i=1}^N$, and we omit the dictionary index m for clarity. From (6) we obtain

$$\begin{aligned} & y_i | - \sim \mathcal{N}(\mu_*, \sigma_*)\,, \\ & \mu_* = \left(\frac{\boldsymbol{k}_{i,\backslash i} \boldsymbol{K}_{\backslash i, \backslash i}^{-1} \boldsymbol{y}_{\backslash i}^T}{k_{i,i} - \boldsymbol{k}_{i,\backslash i} \boldsymbol{K}_{\backslash i, \backslash i}^{-1} \boldsymbol{k}_{i,\backslash i}^T} + z_i - \frac{1}{2} - \gamma_i b\right) \sigma_*^2\,, \\ & \sigma_*^2 = \left(\frac{1}{k_{i,i} - \boldsymbol{k}_{i,\backslash i} \boldsymbol{K}_{\backslash i, \backslash i}^{-1} \boldsymbol{K}_{i,\backslash i}^T} + \gamma_i\right)^{-1}\,, \\ & \boldsymbol{K} = \begin{bmatrix} k_{i,i} & \boldsymbol{k}_{i,\backslash i} \\ \boldsymbol{k}_{i,\backslash i}^T & \boldsymbol{K}_{\backslash i, \backslash i} \end{bmatrix}\,, \end{aligned} \tag{7}$$

where "$-$" denotes all conditioning parameters. Note that both $\boldsymbol{K}$ and $\boldsymbol{y}$ have been permuted to keep $k_{i,i}$ on the left-top corner of the matrix, for notational convenience. We adopt a patch-by-patch approach to sequentially sample all patches indexed by i. It is possible to sample sub-regions of adjacent patches simultaneously from a blocked multivariate Gaussian. However, sub-region size should be carefully selected or estimated from data. As an alternative to sampling, we could use fast variational methods for sparse GPs instead (Titsias, 2009; Hensman et al., 2013). We leave these possibilities as interesting future work.

The conditional posterior for binary activations, z_{im}, is dependent on both dictionary factorization and Gaussian process prior, thus we can write

$$z_{im}|- \sim \text{ Bernoulli}\left(p^*_{im}/(1+p^*_{im})\right) , \tag{8}$$
$$p^*_{im} = \exp\left\{ \frac{1}{\sigma_\varepsilon^2}\sum_{j=1}^{J} \left(x_{ij} - \sum_{m'\neq m} d_{jm'}s_{im'}\right) d_{jm}w_{im} - \frac{1}{2\sigma_\varepsilon^2}(d_{jm}w_{im})^2 + y_{im} + b_m \right\} .$$

Kronecker method Unfortunately, the approach in (7) is costly, scaling as $\mathcal{O}(N^3)$ time and $\mathcal{O}(N^2)$ memory per patch, due to matrix inversion. As a result, it becomes prohibitive even when processing relatively small images, say where $N=(256-8+1)^2$, for a 256×256 image and patch size 256×256. Recently, efficient GP methods were proposed for both Gaussian (Gilboa et al., 2015) and non-Gaussian (Flaxman et al., 2015) likelihoods, by exploiting the Kronecker structure of multiplicative GPs. In this paper, we adopt the fast inference method of Gilboa et al. (2015), where the computational cost can be effectively reduced to $\mathcal{O}(N^{3/2})$ time and $\mathcal{O}(N)$ memory per patch. Specifically, by defining $\mathbf{\Gamma} = \begin{bmatrix} \gamma^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$, from the block matrix inversion lemma (Petersen and Pedersen, 2012) we can write

$$(\boldsymbol{K}+\mathbf{\Gamma})^{-1} = \begin{bmatrix} k_{i,i}+\gamma^{-1} & \boldsymbol{k}_{i,\backslash i} \\ \boldsymbol{k}_{i,\backslash i}^T & \boldsymbol{K}_{\backslash i,\backslash i} \end{bmatrix}^{-1} \stackrel{\gamma\to 0}{\approx} \begin{bmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{K}_{\backslash i,\backslash i}^{-1} \end{bmatrix} . \tag{9}$$

The approximation in (9) allows us to perform a single inversion on the full Gram matrix, $\boldsymbol{K}$, instead of $\boldsymbol{K}_{\backslash i,\backslash i}$. This is desirable because $\boldsymbol{K}$ can be represented as $\boldsymbol{K}=\boldsymbol{K}_1\otimes\boldsymbol{K}_2$. Existing Kronecker methods can be applied via Preconditioned Conjugate Gradient (PCG) (Shewchuk, 1994), by solving the following linear system of equations

$$\boldsymbol{P}\left(\boldsymbol{K}+\mathbf{\Gamma}\right)^{-1}\boldsymbol{P}^T\boldsymbol{x} = \boldsymbol{P}\boldsymbol{y}^T , \tag{10}$$

where $\boldsymbol{P}=\mathbf{\Gamma}^{-\frac{1}{2}}$ is the pre-conditioner matrix. The key idea behind (10) relies on the fast computation of $\boldsymbol{\alpha}=(\otimes_{d=1}^D\boldsymbol{A}_d)\boldsymbol{b}$. Complete details about this algorithm are provided in the Supplementary Material. To further reduce computational cost, we ignore locations with negligible correlation *w.r.t* the current location, *i.e.*, $\rho(i,i')<10^{-6}$. This enables us to consider only a relatively small number of neighbor patches within radius R to the current location. This is a consequence of the light tails of the SE covariance function. In practice, we found that the neighborhood radius, R, is determined by the length scale of the GP, θ, and is usually less than 20. Thanks to this approximation, the computation cost per patch is further reduced to $\mathcal{O}(R^3)$ time.

Automatic relevance determination To estimate the parameters of the covariance functions, $\{\mathbf{\Theta}_m\}_{m=1}^M$, we use *maximum a posterior* (MAP) estimation for $(\sigma_f^2)_m$ and θ_m, corresponding to dictionary element m. This is done by maximizing the conditional log-posterior function, L_m. Omitting constant terms we can write L_m as

$$\begin{aligned} L_m &= \log\textstyle\prod_{i=1}^N p(y_{im}|(\sigma_f^2)_m,\theta_m)p_0((\sigma_f^2)_m,\theta_m) \\ &= -\frac{1}{2}\log|\boldsymbol{K}| - \frac{1}{2}\boldsymbol{y}_m^T\boldsymbol{K}^{-1}\boldsymbol{y}_m + \log p_0\left((\sigma_f^2)_m,\theta_m\right) , \end{aligned}$$

where p_0 is the prior for $(\sigma_f^2)_m$ and θ_m, specified as $\log\mathcal{N}(0,1)\times\log\mathcal{N}(0,1)$. Provided that we estimate individual characteristic length scales, θ_m for each factor, m, in a MAP context, our approach can be seen as an instance of automatic relevance determination (Neal, 1996). Again, the Kronecker product trick can be employed for fast inference via Cholesky decompositions, denoted here as $\text{chol}(\cdot)$. The entire computation for one patch can be done in $\mathcal{O}(N^{3/2})$ time and $\mathcal{O}(N)$ memory. We can write

$$\begin{aligned} L_m = &-\frac{\sqrt{N}}{2}\log|\boldsymbol{K}_m^{(1)}| - \frac{\sqrt{N}}{2}\log|\boldsymbol{K}_m^{(2)}| - \frac{1}{2}\text{Tr}(\boldsymbol{v}^T\boldsymbol{v}) \\ &+ \log\ p_0\left((\sigma_f^2)_m,\theta_m\right) , \end{aligned}$$

where $\boldsymbol{v}=y_m\{(L_1^T)^{-1}\otimes(L_2^T)^{-1}\}$ and $L_1=\text{chol}(\boldsymbol{K}_m^{(1)})$ and $L_2=\text{chol}(\boldsymbol{K}_m^{(2)})$. Note that parameters, $\{\mathbf{\Theta}_m\}_{m=1}^M$, are factor-wise independent, thus can be updated in parallel.

Dictionary size determination So far we have only considered a predefined dictionary size M. It is possible, to incorporate a non-parametric determination of dictionary size, by borrowing ideas from the sampling scheme used in beta processes (Thibaux and Jordan, 2007). Starting from an arbitrary-sized binary activation matrix, $\{\boldsymbol{z}_i\}_{i=1}^N$, the number of new dictionary elements associated with sample i can be drawn from $\text{Poisson}(\alpha/N)$. Conversely, if binary activations for a dictionary element are all zero, we delete the corresponding column of $\boldsymbol{D}$. Despite of the lack of theoretical guarantees due to the non-exchangeability implied by the GP prior, in practice, this heuristic approach can automatically resize the dictionary, until an equilibrium is reached.

Missing pixels In tasks such as inpainting, we are given images with missing pixels. As in Zhou et al. (2009), missing pixel values can be integrated out, thus inference can be performed *w.r.t.* observed pixels only. Further, we can impute missing values by treating them as latent variables to be estimated jointly with all the other parameters of the model, via closed-from conditional predictive distributions.

4 DICTIONARY LEARNING WITH GP-SBN

We leverage sigmoid belief networks (SBNs) as an alternative way of linking binary activations in (1) with the binary output from the GP in (3). As shown in Figure 1, instead of placing a GP prior on each of the M dictionary elements as in GP-FA, we use GPs to impose spatial dependency on the hidden units of an SBN. We denote this model as GP-SBN-FA, for short. Building upon recent work on SBNs (Gan et al., 2015), we consider an SBN with L binary units, which can be written as

$$\begin{aligned} \boldsymbol{z}_i &\sim \text{Bernoulli}(\sigma(\boldsymbol{V}\boldsymbol{h}_i + \boldsymbol{b}))\,, \\ \boldsymbol{h}_i &\sim \text{Bernoulli}(\sigma(\boldsymbol{y}_i))\,, \end{aligned} \tag{11}$$

where $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{iM})^T$ has a GP prior, as in (4), and $\boldsymbol{h}_i \in \{0,1\}^L$ is a vector of L binary units. The weight matrix, $\boldsymbol{V} \in \mathbb{R}^{M\times L}$, contains L features encoding M dictionary elements correlations. We place a three-parameter beta normal prior on the weight matrix, $\boldsymbol{V}$, which has demonstrated good mixing performance (Gan et al., 2015). Further, we let $\boldsymbol{b} \sim \mathcal{N}(0, \boldsymbol{I}_M)$, for simplicity. Closed-form conditional posteriors for $\{\boldsymbol{V}, \boldsymbol{b}\}$ via Gibbs sampling are available via Pólya-gamma data augmentation (Gan et al., 2015). The conditional posterior for $\boldsymbol{h}_i$ is very similar to that for z_{im} in (8). See the Supplementary Material about Gibbs updates for $\{\boldsymbol{V}, \boldsymbol{b}\}$ and $\{\boldsymbol{h}_i\}_{i=1}^N$. In this work we only consider one-layer SBNs as in (11). However, adding layers to form deep architectures is straightforward, as previously described by Gan et al. (2015). Use of deep SBNs may result in more computational savings, as the number of top-layer deep SBN units can be small, reducing the number of needed GPs. With this said, the single-layer SBN considered here yields excellent results, and computational savings.

From Figure 1 and (11), we see that we are imposing smoothness on the activation of latent features of the SBN, not on the activation of dictionary elements, $\boldsymbol{z}_i$, as in GP-FA. It is likely that the number of SBN features, L, needed to describe correlations across dictionary elements is considerably smaller than the dictionary size, and therefore $M < L$. We have observed that $L = M/2$ works well in practice.

GP-SBN-FA is motivated by two key ideas. First, the SBN accounts for inter-dictionary dependencies via $\boldsymbol{V}$, thus it learns the correlation structure of dictionary elements *w.r.t.* the probability of binary activations. Second, by placing GP priors directly on the hidden units of the SBN, we need to specify fewer GPs, thus reducing the overall computational cost.

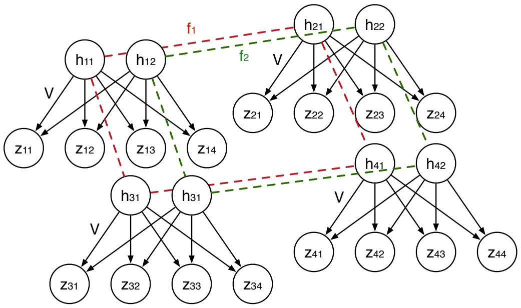

Figure 1: GP-SBN-FA setup. Dashed lines represent GP fields ($f_1(\cdot)$ and $f_2(\cdot)$). Instead of directly impose GP priors on $\{\boldsymbol{z}_i\}_{i=1}^N$, GP-SBN-FA assigns GP priors to the hidden units of the SBN, $\{\boldsymbol{h}_i\}_{i=1}^N$, that further impose correlation structure on dictionary elements, $\boldsymbol{D}$. The weights $\boldsymbol{V}$ are shared across patches.

Previous work has shown that one-layer SBNs with infinite number of hidden units can be explained as GPs (Neal, 1996). However, stacking GPs in multi-layer configurations such as that of Damianou and Lawrence (2013) can be prohibitive due to the high cost of GP inference. Our GP-SBN-FA can be seen as a way to combine the flexibility of GPs with the computational efficiency of SBNs, in a model where GPs are only used where they are most needed, in our case to capture spatial dependencies.

5 DEPTH INFORMATION RESTORATION

Modern consumer depth cameras (Zhang, 2012) capture color and depth (RGB-D) information of a scene. However, depth maps captured by these devices are often noisy and miss values at some pixels, especially around object boundaries. Meanwhile, color images are often noisy (Lu et al., 2014a). Our GP approach can be naturally extended to characterize dependencies between different channels, for simultaneously imputing missing depth values and denoising RGB-D channels. While the method described above characterizes the spatial-dependency within a single channel, here we generalize it to multiple channels, to exploit channel dependencies in RGB-D images. To the best of our knowledge, this is the first Bayesian approach for the joint learning of multi-channel images applied to RGB-D image restoration.

Our method can be applied to this scenario by considering a 3-D covariance function for the GP. We extend the covariates for patch i as $\boldsymbol{l}_i = \{l_i^{(1)}, l_i^{(2)}, l_i^{(c)}\}$, where

$l_i^{(c)} = \{\mathrm{R}, \mathrm{G}, \mathrm{B}, \mathrm{D}\}$ indicates the channel of patch i. Prior domain knowledge suggests that depth-to-RGB correlation must be lower than correlation within RGB channels (Lu et al., 2014a). To leverage such prior information, we introduce two additional parameters in our GP formulation. We let θ_{dm} to describe the relative depth-to-RGB dissimilarity. The dissimilarity within RGB is defined as 1 to avoid identifiability issues. We also let θ_{cm} be the characteristic length scale of the third-dimension covariance function, k_m^c. The covariance function, represented in Gram matrix, is defined as $\boldsymbol{K}_m = \boldsymbol{K}_m^{(1)} \otimes \boldsymbol{K}_m^{(2)} \otimes \boldsymbol{K}_m^{(c)}$. Specifically,

$$k_m^{(c)}(l_i^{(c)}, l_i^{(c)}) = (\sigma_f^2)_m \exp\{-\mathrm{dist}(l_i^{(c)}, l_{i'}^{(c)}; \theta_{dm})^2/\theta_{cm}\} \,,$$

$$\mathrm{dist}(l_i^{(c)}, l_{i'}^{(c)}) = \begin{cases} 0, \text{if } l_i^{(c)} = l_{i'}^{(c)} \\ \theta_{dm}, \text{if } l_i^{(c)} = \{\mathrm{D}\} \text{ and } l_{i'}^{(c)} \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\} \\ \theta_{dm}, \text{if } l_i^{(c)} \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\} \text{ and } l_{i'}^{(c)} = \{\mathrm{D}\} \\ 1, \text{if } c \neq l_{i'}^{(c)} \text{ and } l_i^{(c)}, l_{i'}^{(c)} \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\} \end{cases}$$

Conceptually, when determining whether a certain dictionary is activated on a certain patch, the model seeks for both dictionary activation structure within current channel and across different channels. This is desirable because dictionary elements are shared among all channels, thus dictionary elements appearing in one channel are likely to appear at the same location in other channels.

We note that the depth channel may exhibit smoother transitions when activating dictionary elements compared to color channels (Lu et al., 2014a). This behavior can be characterized by a more-sophisticated covariance function, where spatial functions for patch pair $\{i, i'\}$, *i.e.*, $k_m^{(1)}(\boldsymbol{l}_i, \boldsymbol{l}_{i'})$ and $k_m^{(2)}(\boldsymbol{l}_i, \boldsymbol{l}_{i'})$ are no longer independent of the value of the channel covariates, $\{l_i^{(c)}, l_{i'}^{(c)}\}$. One particular drawback of such a setup is that the Kronecker method that yields fast inference is not feasible anymore. In view of this, we resort to a simple solution where kernels are independent of each other. Like in the previous section, parameters θ_{dm} and θ_{cm} are inferred using MAP estimation.

6 EXPERIMENTS

We present experiments on two sets of images. The results on gray-scale images for denoising and inpainting tasks highlight how characterization of spatial structure improves results. Results on RGB-D images demonstrates that our GP based approach can improve the restoration of multi-channel images, by capturing channel dependencies.

$\sigma = 25$					
Method	C.man	House	Pepper	Lena	Barbara
BPFA	28.41	31.92	29.36	31.25	28.83
GP-FA	28.70	32.22	29.65	31.42	29.11
GP-SBN-FA	**28.99**	**32.23**	**29.78**	**31.51**	**29.18**
Method	Boats	F.print	Man	Couple	Hill
BPFA	29.25	27.44	29.06	28.89	29.29
GP-FA	29.49	**27.55**	**29.27**	29.04	29.49
GP-SBN-FA	**29.56**	27.54	29.23	**29.15**	**29.52**
$\sigma = 50$					
Method	C.man	House	Pepper	Lena	Barbara
BPFA	24.31	27.62	25.41	27.59	25.14
GP-FA	24.66	28.12	**25.71**	27.80	**25.44**
GP-SBN-FA	**24.66**	**28.15**	25.67	**27.83**	25.39
Method	Boats	F.print	Man	Couple	Hill
BPFA	25.72	23.80	25.95	25.37	26.25
GP-FA	25.99	**23.91**	**26.22**	**25.51**	**26.48**
GP-SBN-FA	**26.03**	23.89	26.18	25.45	26.45

Table 1: Denoising results for 2 noise levels $\sigma = \{25, 50\}$. Performance is measured as PSNR in dB.

6.1 2-D Grayscale Images

Denoising We analyzed 10 gray-scale images typically used for demonstration of image denoising. We added isotropic *i.i.d.* Gaussian noise, $\mathcal{N}(0, \sigma)$, to each pixel with $\sigma = 25$ and 50. As input to our model, each image was partitioned into 8×8 patches with sliding distance of one pixel, *i.e.*, the distance between centers of neighbor patches is one pixel. We ran 500 MCMC iterations with random initialization and kept the last 50 samples for image reconstruction (averaging over these collection samples). The hyper-parameters controlling Gaussian distribution variances, *i.e.*, σ_w, σ_ε, σ_b and σ_λ, were all set to 0.1. As suggested in Zhou et al. (2009), the hyper-parameters for the Gamma distributions were set to 10^{-6}. Dictionary sizes in both GP-FA and GP-SBN-FA are initially set to 128. In GP-SBN-FA, we use a one-layer SBN with the number of binary units set to half the size of the dictionary (reducing the number of GPs by half). For each MCMC iteration, computations were parallelized *w.r.t.* dictionary elements using a desktop GPU. We use Peak Signal-to-Noise Ratio (PSNR) to measure the recovery performance of original images. Compared with BPFA (Zhou et al., 2009), as shown in Table 1, GP-SBN-FA yields the best results for most images under different noise regimes. The performance of dHBP (Zhou et al., 2011) is similar to BPFA but no better than GP-FA or GP-SBN-FA, thus not shown.

Inpainting We performed image interpolation on the same gray-scale images from the previous experiment, where a portion of pixels was set to missing at random. The dictionary size is set as either 256 or 512, to match image size. Other hyper-parameters are set as in the denoising task. We consider two observed data ratios, 20% and 50% (observed pixels selected

Method	C.man	House	Pepper	Lena	Barbara
BPFA	28.90	38.02	32.58	36.94	33.17
dHBP	**29.89**	38.83	32.90	37.14	**36.03**
GP-FA	29.03	38.53	32.84	**37.18**	33.18
GP-SBN-FA	28.98	**38.89**	**33.04**	37.01	33.33
Method	Boats	F.print	Man	Couple	Hill
BPFA	33.78	33.53	33.29	**35.56**	34.23
dHBP	33.92	32.70	33.72	33.54	34.14
GP-FA	**34.16**	**34.08**	**33.83**	34.63	**34.46**
GP-SBN-FA	33.98	33.89	33.54	33.60	34.31

Table 2: Inpainting results, 50% observed data. Performance is measured as PSNR in dB.

uniformly at random). 500 MCMC iterations were used and 50 samples were collected for reconstruction. In this task, we compared with dHBP and BPFA, results and experiment settings for dHBP were obtained from Zhou et al. (2011). The results on 50% observed data are shown in Table 2. GP-FA and GP-SBN-FA can generally yield better PSNR than dHBP when the proportion of observed data is relatively high, 50%. When this proportion drops to 20%, dHBP tends to outperform our approach (see the Supplementary Material). We hypothesize that lower observed proportions may lead to poor estimation of the GP posterior, where a predefined filtering function with domain knowledge such as that of dHBP may be favorable.

The learned dictionary elements and binary activations for dictionary elements, obtained from 50% observed data and GP-FA, are shown in Figure 2. For GP-SBN-FA, the binary units of SBN are also shown in Figure 3. Both GP-FA and GP-SBN-FA effectively capture spatial dependencies by incorporating GP priors. The binary activation patterns of GP-SBN-FA for each dictionary, as seen in Figure 3(a), seem to be more similar with each other, compared to those from GP-FA, see Figure 2(a). One explanation is that the imposed SBN architecture encourages blocks of dictionary elements to simultaneously turn on or turn off. The results showing inter-dictionary dependency are provided in the Supplementary Material. Such an inter-dictionary dependency assumption is useful if the dictionary elements are heavily correlated.

All the experiments were conducted on a single machine with two 2.7 GHz processors and 12 GB RAM. By taking advantage of GPU and partial C++ implementation, the running time of GP-FA and GP-SBN-FA is comparable with dHBP. For a 256×256 image, one single iteration of GP-FA takes 96 seconds, while GP-SBN-FA takes 68 seconds. When doubling the number of MCMC iterations, the average PSNR for our method increases by approximately 0.15 dB for both GP-FA and GP-SBN-FA in the inpainting tasks, suggesting that taking more MCMC iterations may marginally improve results.

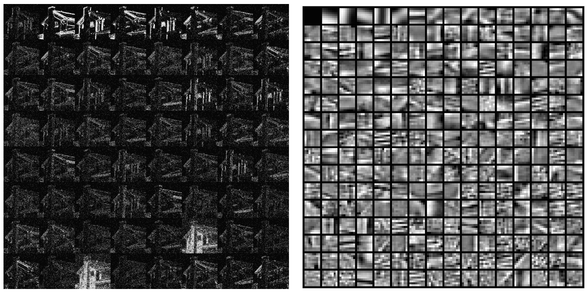
(a) (b)

Figure 2: Inpainting from GP-FA. a) Binary activations of dictionary elements learned. Each block represents a (reshaped) dictionary element. Bright points indicate dictionary activation for a given patch. First 64 dictionary elements are shown. Average sparsity level is around 5%. b) Dictionary elements learned.

6.2 Depth Restoration

We applied our methods to the 30 images of the Middlebury stereo dataset (Scharstein and Szeliski, 2002; Lu et al., 2014a). The provided RGB images are noisy, and the depth-information for each image has a portion of missing pixels, 14% in average. The task is to jointly recover the corrupted pixels in the depth map and to denoise RGB-D channels. We compared our methods with BPFA and dHBP. These models can directly process RGB data by collapsing channels. However, our model leverages the information across channels independently. Thus, instead of extracting patches that consist of all RGB-D channels, *i.e.*, each patch having $8 \times 8 \times 4$ pixels, we extract patches within each channel individually, 8×8 pixels, to impose milder assumptions on the dependency structure over channels (see Supplementary Material for details). For fair comparison, we also tested BPFA and dHBP under this patch extraction strategy. The learned dictionary elements are shared across channels. The proposed patch extraction approach leads to a $\sim$ 1dB improvement in PSNR for BPFA and dHBP. We used 500 burn-in samples for our methods, and kept 50 MCMC collection samples for image reconstruction. For BPFA and dHBP, we use default settings for the hyper-parameters (see Supplementary Material for details), and perform 64 sequential MCMC iterations with incomplete data (Zhou et al., 2009), followed by 300 MCMC iterations. An overall comparison of depth channel interpolation task is shown in Figure 4. In general, GP-FA is marginally better than GP-SBN-FA, as in about 75% images it performs better than GP-SBN-FA. However, GP-SBN-FA is approximately 25% faster than GP-FA. GP-FA and GP-SBN-FA are consistently better than dHBP and BPFA in all images. A detailed comparison for each image is provided in

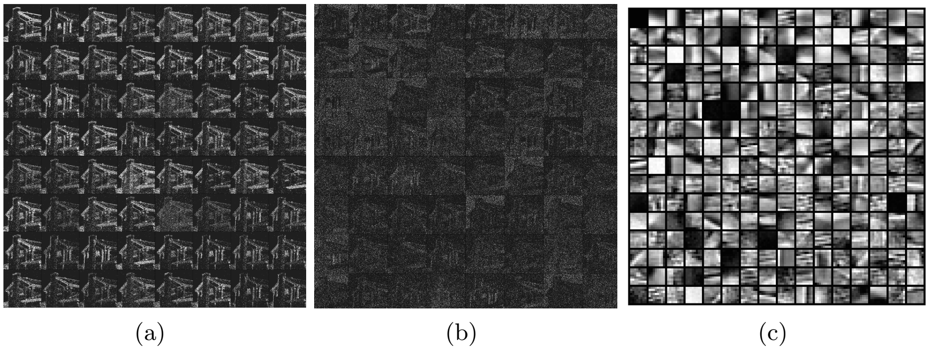

Figure 3: Inpainting from GP-SBN-FA. a) Binary activations of dictionary elements learned. First 64 dictionary elements are shown. b) Binary hidden units of SBN. c) Dictionary elements learned.

the Supplementary Material. Figure 5 shows for one image, restored image and restored depth-information.

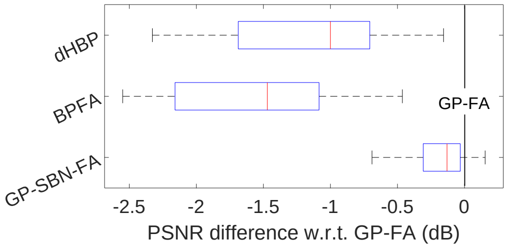

Figure 4: Summarized results of depth-information restoration comparison. Each method was compared with GP-FA, computing differences in PSNR for each image *w.r.t.* GP-FA. Boxplots summarize the distribution of such differences over all images. The red horizontal line denotes the median PSNR difference.

Provided that the original implementation of dHBP only applies to 2-D spatial filters, we performed another experiment on dHBP, where channel information was excluded from the covariance function, *i.e.*, we removed $k_m^{(c)}$ from their smoothing function. We verified that resulting PSNRs with and without channel information are about the same (see Supplementary Material for details).

One key observation about our approach is that by removing the third covariance function, $k_m^{(c)}$, the resulting average PSNR decreases by about 0.5 dB. This suggests that imputation on the depth channel can effectively borrow information from color channels via the GP prior. Another phenomenon is that for cases where the local smoothness assumption about the data does not hold, FA-GP and dHBP do not perform well. We also noticed that FA-GP yields good imputation results particularly when the image has repeated patterns. This may be explained by the fact that GPs can capture periodic behaviors, whereas smoothing kernel functions decaying over distance are likely to fail. We also found that the binary activation patterns of RGB channels are similar to each other, while the activate dictionary elements in the depth channel exhibit weaker similarities with color channels (see Supplementary Material for detailed results).

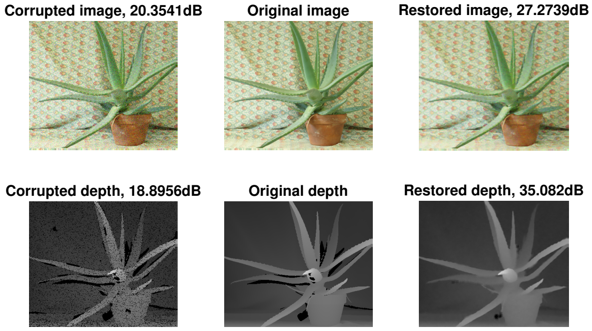
Corrupted image, 20.3541dB
Original image
Restored image, 27.2739dB
Corrupted depth, 18.8956dB
Original depth
Restored depth, 35.082dB

Figure 5: Depth restoration results using GP-FA. The task is to jointly denoise RGB channels and interpolate the depth channel. Upper panels shows recovery from a noisy RGB image. Lower panel shows interpolation of depth-information.

7 DISCUSSION

We have presented a dictionary learning model that captures spatial correlation of dictionary activation patterns in a principled non-parametric way. Binary activation vectors indicating the presence or absence of each dictionary element are established either via a Gaussian process field followed by logistic link functions, or a Gaussian process field followed by a sigmoid belief network. Pólya-gamma augmentation and Kronecker methods were described for efficient MCMC inference. Experiments on real-world images demonstrated that our approach performs better than related Bayesian dictionary learning models in inpainting, denoising and depth restoration tasks.

References

M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

H. M. Choi, J. P. Hobert, et al. The polya-gamma gibbs sampler for bayesian logistic regression is uniformly ergodic. *Electronic Journal of Statistics*, 7: 2054–2064, 2013.

A. C. Damianou and N. D. Lawrence. Deep Gaussian Processes. Nov. 2013.

S. Flaxman, A. G. Wilson, D. Neill, H. Nickisch, and A. J. Smola. Fast Kronecker Inference in Gaussian Processes with non-Gaussian Likelihoods. pages 607–616, 2015.

Z. Gan, R. Henao, D. Carlson, and L. Carin. Learning Deep Sigmoid Belief Networks with Data Augmentation. 2015.

P. Garrigues and B. A. Olshausen. Learning Horizontal Connections in a Sparse Coding Model of Natural Images. pages 505–512, 2007.

Z. Ghahramani and T. L. Griffiths. Infinite latent feature models and the indian buffet process. In *NIPS*, pages 475–482, 2005.

E. Gilboa, Y. Saatci, and J. P. Cunningham. Scaling Multidimensional Inference for Structured Gaussian Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):424–436, 2015.

H. He and W.-C. Siu. Single image super-resolution using Gaussian process regression. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 449–456. IEEE, 2011.

J. Hensman, N. Fusi, and N. D. Lawrence. Gaussian processes for big data. *arxiv*, 2013.

P. J. Liu. *Using Gaussian process regression to denoise images and remove artefacts from microarray data.* PhD thesis, University of Toronto, 2007.

S. Lu, X. Ren, and F. Liu. Depth Enhancement via Low-Rank Matrix Completion. pages 3390–3397, 2014a.

S. Lu, X. Ren, and F. Liu. Depth Enhancement via Low-Rank Matrix Completion. pages 3390–3397, 2014b.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *ICML*, 2009a.

J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *ICML*, pages 689–696, 2009b.

R. M. Neal. *Bayesian Learning for Neural Networks.* Springer, Sept. 1996.

R. M. Neal. Monte Carlo Implementation of Gaussian Process Models for Bayesian Regression and Classification. *arxiv*, Jan. 1997.

K. B. Petersen and M. S. Pedersen. The matrix cookbook, nov 2012.

G. Polatkan, M. Zhou, L. Carin, D. Blei, and I. Daubechies. A Bayesian Nonparametric Approach to Image Super-Resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2): 346–358, Feb. 2015.

N. G. Polson, J. G. Scott, and J. Windle. Bayesian Inference for Logistic Models Using Pólya–Gamma Latent Variables. *JASA*, 108(504):1339–1349, Aug. 2013.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* MIT Press, Jan. 2006.

D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 2002.

J. R. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain, 1994.

R. Thibaux and M. I. Jordan. Hierarchical beta processes and the indian buffet process. In *AISTATS*, pages 564–571, 2007.

M. K. Titsias. Variational learning of inducing variables in sparse gaussian processes. In *AISTATS*, pages 567–574, 2009.

C. Wachinger, P. Golland, M. Reuter, and W. Wells. Gaussian process interpolation for uncertainty estimation in image registration. In *MICCAI*, pages 267–274. Springer, 2014a.

C. Wachinger, P. Golland, M. Reuter, and W. M. Wells III. Gaussian Process Interpolation for Uncertainty Estimation in Image Registration. volume 17, pages 267–274, 2014b.

S. Wang, L. Zhang, and R. Urtasun. Transductive gaussian processes for image denoising. In *ICCP*, pages 1–8. IEEE, 2014.

Z. Xing, M. Zhou, A. Castrodad, G. Sapiro, and L. Carin. Dictionary Learning for Noisy and Incomplete Hyperspectral Images. *SIAM J. Imaging Sciences*, 5(1):33–56, 2012.

J. Yang, Z. Wang, Z. Lin, S. Cohen, and T. Huang. Coupled dictionary training for image super-resolution. *Image Processing, IEEE Transactions on*, 21(8):3467–3478, 2012.

Z. Zhang. Microsoft Kinect Sensor and Its Effect. *IEEE MultiMedia*, 19(2):4–10, 2012.

M. Zhou, H. Chen, L. Ren, G. Sapiro, L. Carin, and J. W. Paisley. Non-Parametric Bayesian Dictionary Learning for Sparse Image Representations. pages 2295–2303, 2009.

M. Zhou, H. Yang, and G. Sapiro. Dependent hierarchical beta process for image interpolation and denoising. *AISTATS*, 2011.

M. Zhou, H. Chen, J. W. Paisley, L. Ren, L. Li, Z. Xing, D. B. Dunson, G. Sapiro, and L. Carin. Nonparametric Bayesian Dictionary Learning for Analysis of Noisy and Incomplete Images. *IEEE Transactions on Image Processing*, 21(1):130–144, 2012.