NONPARAMETRIC BAYESIAN DICTIONARY LEARNING FOR ANALYSIS OF NOISY AND INCOMPLETE IMAGES

By

Mingyuan Zhou, Haojun Chen

John Paisley, Lu Ren

Lingbo Li, Zhengming Xing,

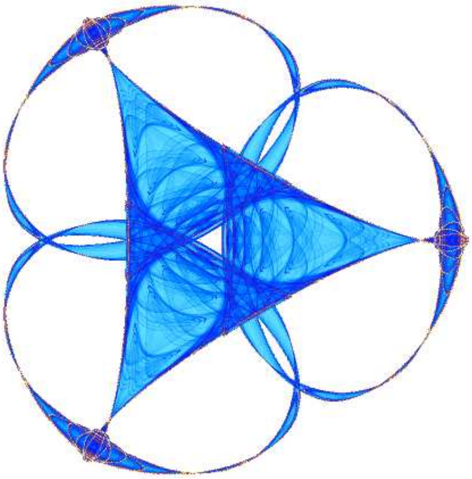
David Dunson, Guillermo Sapiro

and

Lawrence Carin

IMA Preprint Series # 2307

(April 2010)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA
400 Lind Hall
207 Church Street S.E.
Minneapolis, Minnesota 55455–0436

Phone: 612-624-6066 Fax: 612-626-7370
URL: http://www.ima.umn.edu

Nonparametric Bayesian Dictionary Learning for Analysis of Noisy and Incomplete Images

[1]Mingyuan Zhou, [1]Haojun Chen, [1]John Paisley, [1]Lu Ren, [1]Lingbo Li,
[1]Zhengming Xing, [2]David Dunson, [3]Guillermo Sapiro and [1]Lawrence Carin

[1]Department of Electrical and Computer Engineering and [2]Statistics Department
Duke University, Durham, NC 27708-0291, USA
[3]Department of Electrical and Computer Engineering
University of Minnesota, Minneapolis, MN 55455, USA

Abstract

Nonparametric Bayesian methods are considered for recovery of imagery based upon compressive, incomplete and/or noisy measurements. A truncated beta-Bernoulli process is employed to infer an appropriate dictionary for the data under test, and also for image recovery. In the context of compressive sensing, significant improvements in image recovery are manifested using learned dictionaries, relative to using standard orthonormal image expansions. The compressive-measurement projections are also optimized for the learned dictionary. Additionally, we consider simpler (incomplete) measurements, defined by measuring a subset of image pixels, selected uniformly at random; connections are made to matrix completion and union-of-subspace models, providing a link between matrix completion and image processing. Spatial inter-relationships within imagery are exploited through use of the Dirichlet and probit stick-breaking processes. Several example results are presented, with comparisons to other methods in the literature.

I. Introduction

There has been significant recent interest in sparse image representations, in the context of denoising and interpolation [1], [13], [24]–[26], [28], [29], [33], compressive sensing (CS) [5], [12], and classification [40]. All of these applications exploit the fact that images may be sparsely represented in an appropriate dictionary. Most of the denoising, interpolation, and CS literature assumes “off-the-shelf” wavelet and

Work partially supported by DOE, NSF, ONR, NGA, and ARO.

DCT bases/dictionaries [20], but recent research has demonstrated the significant utility of learning an often over-complete dictionary matched to the signals of interest (*e.g.*, images) [1], [3], [12], [13], [24]–[26], [28], [29], [31], [33], [41].

Many of the existing methods for learning dictionaries are based on solving an optimization problem [1], [13], [24]–[26], [28], [29], in which one seeks to match the dictionary to the imagery of interest, while simultaneously encouraging a sparse representation. These methods have demonstrated state-of-the-art performance for denoising, super-resolution, interpolation, and inpainting. However, many existing algorithms for implementing such ideas also have some restrictions. For example, one must often assume access to the noise/residual variance, the size of the dictionary is set *a priori* or fixed via cross-validation type techniques, and a single ("point") estimate is learned.

To mitigate the aforementioned limitations, dictionary learning has recently been cast as a factor-analysis problem, with the factor loadings corresponding to the dictionary elements (atoms). Utilizing nonparametric Bayesian methods like the beta process (BP) [30], [38], [42] and the Indian buffet process (IBP) [18], [21], one may for example infer the number of factors (dictionary elements) needed to fit the data itself. Further, one may place a prior on the noise or residual variance, with this inferred from the data [30], [42]. An approximation to the full posterior may be manifested via Gibbs sampling, yielding an ensemble of dictionary representations. Recent research has demonstrated that an ensemble of representations can be better than a single expansion [14], with such an ensemble naturally manifested by statistical models as the one here described. Overall, the here proposed Bayesian framework provides a complementary and alternative framework with respect to the more standard variational formulations. These can also be interpreted via statistical models with solutions obtained via MAP estimation, *e.g.*, see [32] for an overview and new interpretation of this. Such probabilistic interpretations use models different than the ones here exploited, and as mentioned above, have to estimate critical parameters and produce single solutions.

In image analysis there is often additional information that may be exploited when learning dictionaries, with this well suited for Bayesian priors. For example, most natural images may be segmented, and it is probable that dictionary usage will be similar for regions within a particular segment class. To address this idea, we extend the model by employing a probit stick-breaking process (PSBP), with this a generalization of the Dirichlet process (DP) stick-breaking representation [36]. Related clustering techniques have proven successful in image processing [27]. The model clusters the image patches, with each cluster corresponding to a segment type; the PSBP encourages proximate and similar patches to be included within the same segment type, thereby performing image segmentation and dictionary learning simultaneously.

As discussed when presenting results, the proposed method is a natural tool for denoising images, applicable when the noise statistics are nonstationary. The nonstationary noise variance is inferred within the analysis. The principal focus of this paper, however, is on applying the algorithms to new compressive measurement techniques that have been developed recently. Specifically, we consider dictionary learning in the context of compressive sensing (CS) [5], [10], in which the measurements correspond to projections of typical image pixels. We consider dictionary learning performed "offline" based on representative (training) images, with the learned dictionary applied within CS image recovery. We also consider the case for which the underlying dictionary is learned simultaneously with inversion (reconstruction), with this related to "blind" CS [17]. Finally, we design the CS projection matrix to be matched to the learned dictionary (when this is done offline), and demonstrate as in [12] that in practice this yields performance gains relative to conventional random CS projection matrices.

While CS is of interest for its potential to reduce the number of required measurements, it has the disadvantage of requiring the development of new classes of cameras. Such cameras are revolutionary and interesting [11], [37], but there have been decades of previous research performed on development of pixel-based cameras, and it would be desirable if such cameras could be modified simply to perform compressive measurements. In that context, we note that there has been recent interest in the field of matrix completion, in which performance guarantees have been derived that are similar to those associated with CS [6]. One may view the pixel values of an image as a matrix of data, and if one samples the pixels uniformly at random, recovery of the missing pixels corresponds to the matrix-completion problem. However, the matrix-completion literature is based on the assumption that the matrix of interest is low rank [6], [22], [35]. Because the underlying dictionaries associated with natural images are typically over-complete, the assumption of a single low-rank matrix of pixel values is often inappropriate.

While a direct application of matrix-completion technology to this problem is then inappropriate, it may be modified simply such that it is useful. Specifically, because natural images manifest segments and self-similarity, one may view the dictionary-learning framework within a union-of-subspaces setting [15], [23]. Each subspace, defined by a subset of the dictionary, represents a class of *local* structure within an image, and each subspace and associated data may be viewed as a low-rank matrix. The BP, DP and PSBP models discussed above are employed to perform joint clustering and recovery of missing pixels, with comparisons made to related non-Bayesian approaches.

The remainder of the paper is organized as follows. In Section II we review the classes of problems being considered. The beta-Bernoulli process is discussed in Section III, with relationships made with previous work in this area, including those based on the Indian buffet process. The Dirichlet and probit stick-breaking processes are discussed in Section IV, and several example results are presented in Section V. Conclusions and a discussion of future work are provided in Section VI, and details of the inference equations are summarized in the Appendix.

II. Problems Under Study

A. *Denoising and compressive sensing*

We consider data samples that may be expressed in the form

$$\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i \tag{1}$$

where $\boldsymbol{x}_i \in \mathbb{R}^P$, $\boldsymbol{\epsilon}_i \in \mathbb{R}^P$, and $\boldsymbol{w}_i \in \mathbb{R}^K$. The columns of the matrix $\mathbf{D} \in \mathbb{R}^{P \times K}$ represent the K components of a dictionary with which $\boldsymbol{x}_i$ is expanded. For our problem, the $\boldsymbol{x}_i$ will correspond to $B \times B$ (overlapped) pixel patches in an image [1], [13], [24], [25], [28], [42]. The *set* of vectors $\{\boldsymbol{x}_i\}_{i=1,N}$ may be extracted from an image(s) of interest.

For the denoising problem, the vectors $\boldsymbol{\epsilon}_i$ may represent sensor noise, in addition to (ideally small) residual from representation of the underlying signal as $\mathbf{D}\boldsymbol{w}_i$. To perform denoising, we place restrictions on the vectors $\boldsymbol{w}_i$, such that $\mathbf{D}\boldsymbol{w}_i$ by itself does not exactly represent $\boldsymbol{x}_i$. A popular such restriction is that $\boldsymbol{w}_i$ should be sparse, motivated by the idea that any particular $\boldsymbol{x}_i$ may often be represented in terms of a small subset of representative dictionary elements, from the full dictionary defined by the columns of $\mathbf{D}$. There are several methods that have been developed recently to impose such a sparse representation, including ℓ_1-based relaxation algorithms [24], [25], iterative algorithms [1], [13], and Bayesian methods [42]. One advantage of a Bayesian approach is that the noise/residual statistics may be nonstationary (with unknown noise statistics). Specifically, in addition to placing a sparseness-promoting prior on $\boldsymbol{w}_i$, we may also impose a prior on the components of $\boldsymbol{\epsilon}_i$. From the estimated posterior density function on model parameters, each component of $\boldsymbol{\epsilon}_i$, corresponding to the ith $B \times B$ image patch, has its own variance. Given $\{\boldsymbol{x}_i\}_{i=1,N}$, our goal may be to simultaneously infer $\mathbf{D}$ and $\{\boldsymbol{w}_i\}_{i=1,N}$ (and implicitly $\boldsymbol{\epsilon}_i$), and then the denoised version of $\boldsymbol{x}_i$ is represented as $\mathbf{D}\boldsymbol{w}_i$.

In many applications the total number of pixels $N \cdot P$ may be large. However, it is well known that compression algorithms may be used on $\{\boldsymbol{x}_i\}_{i=1,N}$ *after* the measurements have been performed, to significantly reduce the quantity of data that need be stored or communicated. This compression indicates that while the data dimensionality $N \cdot P$ may be large, the underlying information content may be relatively low. This has motivated the field of compressive sensing [5], [10], [11], [37], in which the total number of measurements performed may be much less than $N \cdot P$. Toward this end, researchers have proposed *projection* measurements of the form

$$\boldsymbol{y}_i = \boldsymbol{\Sigma} \boldsymbol{x}_i \tag{2}$$

where $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times P}$ and $\boldsymbol{y}_i \in \mathbb{R}^n$, ideally with $n \ll P$. The projection matrix $\boldsymbol{\Sigma}$ has traditionally been constituted randomly [5], [10], with a binary or real alphabet (and $\boldsymbol{\Sigma}$ may also be a function of the specific patch, and generalized as $\boldsymbol{\Sigma}_i$). It is desirable that matrices $\boldsymbol{\Sigma}$ and $\mathbf{D}$ be as incoherent as possible.

The recovery of $\boldsymbol{x}_i$ from $\boldsymbol{y}_i$ is an ill-posed problem unless restrictions are placed on $\boldsymbol{x}_i$. We may exploit the same class of restrictions used in the denoising problem; specifically, the observed data satisfy $\boldsymbol{y}_i = \boldsymbol{\Phi} \boldsymbol{w}_i + \boldsymbol{\nu}_i$, with $\boldsymbol{\Phi} = \boldsymbol{\Sigma}\mathbf{D}$ and $\boldsymbol{\nu}_i = \boldsymbol{\Sigma}\boldsymbol{\epsilon}_i$, and with sparse $\boldsymbol{w}_i$. Note that the sparseness constraint implies that $\{\boldsymbol{w}_i\}_{i=1,N}$ (and hence $\{\boldsymbol{x}_i\}_{i=1,N}$) occupy *nonlinear* subspaces of $\mathbb{R}^P$.

In most applications of compressive sensing $\mathbf{D}$ is assumed known, corresponding to an orthonormal basis ($e.g.$, wavelets or a DCT) [5], [10], [20]. However, such bases are not necessarily well matched to natural imagery, and it is desirable to consider design of dictionaries $\mathbf{D}$ for this purpose [12]. One may even consider recovering $\{\boldsymbol{x}_i\}_{i=1,N}$ from $\{\boldsymbol{y}_i\}_{i=1,N}$ while simultaneously inferring $\mathbf{D}$. Thus, we again have a dictionary-learning problem, which may be coupled with optimization of the CS matrix $\boldsymbol{\Sigma}$, such that it is matched to $\mathbf{D}$ (defined by a low coherence between the rows of $\boldsymbol{\Sigma}$ and columns of $\mathbf{D}$ [5], [10], [12], [20]).

B. Matrix completion and pixel recovery

Consider a matrix $\mathbf{M} \in \mathbb{R}^{n_1 \times n_2}$ of rank r, and assume we only observe $m \ll n_1 \cdot n_2$ components of this matrix, with the observed components selected uniformly at random. Let Ω represent a set defining the entries of $\mathbf{M}$ for which we have data. Consider the convex program

$$\text{minimize} \quad \|\mathbf{Z}\|_* \tag{3}$$

$$\text{subject to} \quad P_\Omega(\mathbf{Z}) = P_\Omega(\mathbf{M}) \tag{4}$$

where $P_\Omega(\cdot)$ defines the vector of samples of the associated matrix that are in the set Ω. The nuclear norm $\|\mathbf{Z}\|_* = \sum_i \gamma_i(\mathbf{Z})$, where $\gamma_i(\mathbf{Z})$ represents the ith singular value of $\mathbf{Z}$. Defining $n = \max(n_1, n_2)$, Candès and Tao [6] showed that with probability exceeding $1 - n^{-3}$, if $m \geq C\mu^2 nr(\log n)^6$ then $\mathbf{Z} = \mathbf{M}$, thereby recovering the missing entries of $\mathbf{M}$ ([8] considers related issues). The parameter μ represents the coherence, a measure of how "spread out" the singular vectors of $\mathbf{M}$ are, and ideally the coherence is near one. For this result to be useful, we require $r \ll n_1$ and $r \ll n_2$.

The assumption that $\mathbf{M}$ is low rank, and the use of this in recovering missing matrix entries, implies that the columns (and rows) of $\mathbf{M}$ reside in an r-dimensional *linear* subspace. Specifically,

$$\mathbf{M} = \sum_{i=1}^{r} \lambda_i \boldsymbol{u}_i \boldsymbol{v}_i^T \tag{5}$$

where $\lambda_i \in \mathbb{R}$, $\boldsymbol{u}_i \in \mathbb{R}^{n_1}$, and $\boldsymbol{v}_i \in \mathbb{R}^{n_2}$ represent, respectively, the ith singular value, left singular vector, and right singular vector. Hence, each column of $\mathbf{M}$ is assumed to reside in a *linear* subspace defined by $\{\boldsymbol{u}_i\}_{i=1,r}$.

We now reconsider the set of vectors $\{\boldsymbol{x}_i\}_{i=1,N}$ discussed above, again with $\boldsymbol{x}_i \in \mathbb{R}^P$ corresponding to patches of pixels from the image(s). Let $\mathbf{X} \in \mathbb{R}^{P \times N}$ have columns defined by the vectors $\{\boldsymbol{x}_i\}_{i=1,N}$. It is assumed that a randomly constituted subset of the components of $\mathbf{X}$ are measured, and our goal is to recover the missing data using ideas analogous to those employed in matrix-completion theory. However, as discussed above, each $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i$, for sparse $\boldsymbol{w}_i$. If we ignore $\boldsymbol{\epsilon}_i$ for now, the columns of $\mathbf{X}$ reside in a *nonlinear* subspace of $\mathbb{R}^P$, due to the fact that all $\boldsymbol{w}_i$ are sparse (not necessarily with the same sparsity patterns). Further, since the dictionary $\mathbf{D}$ is typically over-complete, $\mathbf{X}$ is generally not low-rank. Therefore, linear matrix-completion theory is not applicable to $\mathbf{X}$.

While a *direct* application of this theory may be inappropriate for image-processing problems, the framework may be modified to make it applicable. As one way in which the model may be modified, recall that images tend to possess significant self-similarity [4] and segments, implying that many $B \times B$ patches have similar structure. This suggests that there may be a *clustering* of the $B \times B$ blocks, and that within each cluster the associated data can constitute the columns of a matrix of low-rank, recoverable in the presence of significant missing matrix values. The nonparametric Bayesian methods developed here implement joint clustering of the observed image patches and inference of the missing pixel values.

III. Sparse Factor Analysis with the Beta-Bernoulli Process

When presenting example results, we will consider three problems. For *denoising*, it is assumed we measure $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i$, where $\boldsymbol{\epsilon}_i$ represents measurement noise and model error; for the *compressive-sensing* application we observe $\boldsymbol{y}_i = \boldsymbol{\Sigma}(\mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i) = \boldsymbol{\Phi}\boldsymbol{w}_i + \boldsymbol{\nu}_i$, with $\boldsymbol{\Phi} = \boldsymbol{\Sigma}\mathbf{D}$ and $\boldsymbol{\nu}_i = \boldsymbol{\Phi}\boldsymbol{\epsilon}_i$; and, finally, for the *interpolation* problem we observe $P_\phi(\mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i)$, where $P_\phi(\boldsymbol{x}_i)$ is a vector of elements from $\boldsymbol{x}_i$ contained within the set ϕ, as in (4). For all three problems our objective is to infer the underlying signal $\mathbf{D}\boldsymbol{w}_i$, with $\boldsymbol{w}_i$ assumed sparse; we generally wish to simultaneously infer $\mathbf{D}$ and $\{\boldsymbol{w}_i\}_{i=1,N}$. To address each of these problems, we consider a statistical model for $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i$, placing Bayesian priors on $\mathbf{D}$, $\boldsymbol{w}_i$ and $\boldsymbol{\epsilon}_i$; the way the model is used is modified slightly for each specific application. For example, when considering interpolation, only the observed $P_\phi(\boldsymbol{x}_i)$ are used within the model likelihood function.

A. Beta-Bernoulli process for active-set selection

Let the binary vector $\boldsymbol{z}_i \in \{0,1\}^K$ denote which of the K columns of $\mathbf{D}$ are used for representation of $\boldsymbol{x}_i$ (active set); if a particular component of $\boldsymbol{z}_i$ is equal to one, then the corresponding column of $\mathbf{D}$ is used in the representation of $\boldsymbol{x}_i$. Hence, for the data $\{\boldsymbol{x}_i\}_{i=1,N}$ there is an associated set of latent binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$, and the beta-Bernoulli process provides a convenient prior for these vectors [30], [38], [42]. Specifically, consider the model

$$\begin{aligned} \boldsymbol{z}_i &\sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_k) \\ \boldsymbol{\pi} &\sim \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \end{aligned} \tag{6}$$

where π_k is the kth component of $\boldsymbol{\pi}$, and a and b are model parameters; the impact of these parameters on the model are discussed below.

Considering the limit $K \to \infty$, and after integrating out $\boldsymbol{\pi}$, the draws of $\{\boldsymbol{z}_i\}_{i=1,N}$ may be constituted as follows. For each $\boldsymbol{z}_i$, draw $c_i \sim \text{Poisson}(\frac{a}{b+i-1})$ and define $C_i = \sum_{j=1}^{i} c_j$, with $C_0 = 0$. Let z_{ik} represent the kth component of $\boldsymbol{z}_i$, and $z_{ik} = 0$ for $k > C_i$. For $k = 1, \ldots, C_{i-1}$, $z_{ik} \sim \text{Bernoulli}(\frac{n_{ik}}{b+i-1})$, where $n_{ik} = \sum_{j=1}^{i-1} z_{jk}$ (n_{ik} represents the total number of times the kth component of $\{\boldsymbol{z}_j\}_{j=1,i-1}$ is one). For $k = C_{i-1}+1, \ldots, C_i$, we set $z_{ik} = 1$. Note that as $a/(b+i-1)$ becomes small, with increasing i, it is probable that c_i will be small. Hence, with increasing i, the number of new non-zero components of $\boldsymbol{z}_i$ diminishes. Further, as a consequence of $\text{Bernoulli}(\frac{n_{ik}}{b+i-1})$, when a particular component of the vectors $\{\boldsymbol{z}_j\}_{j=1,i-1}$ is frequently one, it is more probable that it will be one for subsequent $\boldsymbol{z}_j$, $j \geq i$. When $b = 1$ this construction for $\{\boldsymbol{z}_i\}_{i=1,N}$ corresponds to the Indian buffet process [18].

Since $\boldsymbol{z}_i$ defines which columns of $\mathbf{D}$ are used to represent $\boldsymbol{x}_i$, (6) imposes that it is probable that some columns of $\mathbf{D}$ are used repeatedly among the set $\{\boldsymbol{x}_i\}_{i=1,N}$, while other columns of $\mathbf{D}$ may be more specialized to particular $\boldsymbol{x}_i$. As demonstrated below, this has been found to be a good model when $\{\boldsymbol{x}_i\}_{i=1,N}$ are patches of pixels extracted from natural images.

B. Full hierarchical model

The hierarchical form of the model may now be expressed as

$$\begin{aligned}
\boldsymbol{x}_i &= \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i \\
\boldsymbol{w}_i &= \boldsymbol{z}_i \odot \boldsymbol{s}_i \\
\boldsymbol{d}_k &\sim \mathcal{N}(0, P^{-1}\mathbf{I}_P) \\
\boldsymbol{s}_i &\sim \mathcal{N}(0, \gamma_s^{-1}\mathbf{I}_K) \\
\boldsymbol{\epsilon}_i &\sim \mathcal{N}(0, \gamma_\epsilon^{-1}\mathbf{I}_P)
\end{aligned} \tag{7}$$

where $\boldsymbol{d}_k$ represents the kth component (atom) of $\mathbf{D}$, $\circ$ represents the pointwise or Hadamard vector product, $\mathbf{I}_P$ ($\mathbf{I}_K$) represents a $P \times P$ ($K \times K$) identity matrix, and $\{\boldsymbol{z}_i\}_{i=1,N}$ are drawn as in (6). Conjugate hyperpriors $\gamma_s \sim \text{Gamma}(c,d)$ and $\gamma_\epsilon \sim \text{Gamma}(e,f)$ are also imposed. The construction in (7), and with the prior in (6) for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as the beta process factor analysis (BPFA) model.

Note that we impose independent Gaussian *priors* for $\boldsymbol{d}_k$, $\boldsymbol{s}_i$ and $\boldsymbol{\epsilon}_i$ for modeling convenience (conjugacy of consecutive terms in the hierarchical model). However, the inferred *posterior* for these terms is generally *not* independent or Gaussian. The independent priors essentially impose prior information about the *marginals* of the posterior of each component, while the inferred posterior accounts for statistical dependence as reflected in the data. To make connections of this model to more-typical optimization-based approaches [24], [25], note that the negative logarithm of the posterior density function is

$$\begin{aligned}
-\log\ p(\boldsymbol{\Theta}|\mathcal{D},\mathcal{H}) &= \frac{\gamma_\epsilon}{2}\sum_{i=1}^{N}\|\boldsymbol{x}_i - \mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i)\|_2^2 + \frac{P}{2}\sum_{k=1}^{K}\|\boldsymbol{d}_k\|_2^2 + \frac{\gamma_s}{2}\sum_{i=1}^{N}\|\boldsymbol{s}_i\|_2^2 \\
&- \log f_{Beta-Bern}(\{\boldsymbol{z}_i\}_{i=1}^N;\mathcal{H}) - \log\text{Gamma}(\gamma_\epsilon|\mathcal{H}) - \log\text{Gamma}(\gamma_s|\mathcal{H}) + Const.
\end{aligned} \tag{8}$$

where $\boldsymbol{\Theta}$ represents all unknown model parameters, $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1,N}$, $f_{Beta-Bern}(\{\boldsymbol{z}_i\}_{i=1}^N;\mathcal{H})$ represents the beta-Bernoulli process prior in (6), and $\mathcal{H}$ represents model hyper-parameters (*i.e.*, a, b, c, d, e and f). Therefore, the typical ℓ_2 constraints [24], [25] on the dictionary elements $\boldsymbol{d}_k$ and on the non-zero weights $\boldsymbol{s}_i$ correspond here to the Gaussian priors employed in (7). However, rather than an employing an ℓ_1 (Laplacian prior) constraint [24], [25] to impose sparseness on $\boldsymbol{w}_i$, we employ the beta-Bernoulli process and $\boldsymbol{w}_i = \boldsymbol{s}_i \circ \boldsymbol{z}_i$. The beta-Bernoulli process imposes that the binary $\boldsymbol{z}_i$ should be sparse, *and* that there should be a relatively consistent (re)use of dictionary elements across the image, thereby imposing self-similarity. Further, and perhaps most importantly, we do *not* constitute a point estimate, as one would do if a single $\boldsymbol{\Theta}$ was sought to minimize (8). We rather estimate the full posterior density $p(\boldsymbol{\Theta}|\mathcal{D},\mathcal{H})$, implemented via Gibbs sampling. A significant advantage of the hierarchical construction in (7) is that each Gibbs update equation is analytic, with detailed update equations provided in Appendix B. Note that consistent use of atoms is encouraged because the active sets are defined by the binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$, and these are all drawn from a shared probability vector $\boldsymbol{\pi}$; this is distinct from drawing the active sets i.i.d. from a Laplacian prior. Further, the beta-Bernoulli prior imposes that many components of $\boldsymbol{w}_i$ are exactly zero, while with a Laplacian prior many components are small but not exactly zero.

IV. Patch Clustering via Dirichlet and Probit Stick-Breaking Processes

A. *Dirichlet process*

As discussed in Section II-B, in many applications it is expected that the data patches $\{\boldsymbol{x}_i\}_{i=1,N}$ may cluster, and it is of interest to infer this clustering nonparametrically ($i.e.$, to infer the number of clusters and their composition from the data). The imposition of such prior knowledge may improve the quality of the inversion for $\{\boldsymbol{x}_i\}_{i=1,N}$ based upon incomplete measurements. The Dirichlet process (DP) [16] constitutes a popular means of performing such nonparametric clustering. A random draw from a DP, $G \sim \text{DP}(\alpha G_0)$, with precision $\alpha \in \mathbb{R}^+$ and "base" measure G_0, may be constituted via the stick-breaking construction [36]

$$G = \sum_{l=1}^{\infty} \beta_l \delta_{\theta_l^*} \quad , \quad \theta_l^* \sim G_0 \tag{9}$$

where $\beta_l = V_l \prod_{h=1}^{l-1}(1 - V_h)$ and $V_h \sim \text{Beta}(1, \alpha)$. The β_l may be viewed as a sequence of fractional breaks from a "stick" of original length one, where the fraction of stick broken off on break l is V_l. The θ_l^* are model parameters, associated with the lth data cluster. For our problem it has proven effective to set $G_0 = \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K)$ analogous to (6), and hence $G = \sum_{l=1}^{\infty} \beta_l \delta_{\boldsymbol{\pi}_l^*}$. The $\boldsymbol{\pi}_l^*$, drawn from G_0, correspond to distinct probability vectors for using the K dictionary elements (columns of $\mathbf{D}$). For sample i we draw $\boldsymbol{\pi}_i \sim G$, and a separate sparse binary vector $\boldsymbol{z}_i$ is drawn for each sample $\boldsymbol{x}_i$, as $\boldsymbol{z}_i \sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_{ik})$, with π_{ik} the kth component of $\boldsymbol{\pi}_i$. In practice we truncate the infinite sum for G to N_L elements, and impose $V_{N_L} = 1$, such that $\sum_{l=1}^{N_L} \beta_l = 1$. A (conjugate) gamma prior is placed on the DP parameter α.

We may view this DP construction as an "Indian buffet franchise," generalizing the Indian buffet analogy [18]. Specifically, there are N_L Indian buffet restaurants; each restaurant is composed of the same "menu" (columns of $\mathbf{D}$), and is distinguished by different probabilities for selecting menu items. The "customers" $\{\boldsymbol{x}_i\}_{i=1,N}$ cluster based upon which restaurant they go to. The $\{\boldsymbol{\pi}_l^*\}_{l=1,N_L}$ represent the probability of using each column of $\mathbf{D}$ in the respective N_L different buffets. The $\{\boldsymbol{x}_i\}_{i=1,N}$ cluster themselves among the different restaurants in a manner that is consistent with the characteristics of the data, with the model also simultaneously learning the dictionary/menu $\mathbf{D}$. Note that we typically make the truncation N_L large, and the posterior distribution infers the number of clusters actually needed to support the data, as represented by how many β_l are of significant value. The model in (7), with the above DP construction for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as DP-BPFA.

B. Probit stick-breaking process

The DP yields a clustering of $\{\boldsymbol{x}_i\}_{i=1,N}$, but it does not account for our knowledge of the location of each patch within the image. It is natural to expect that if $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$ are proximate then they are likely to be constituted in terms of similar columns of $\mathbf{D}$ [1]. To impose this information, we employ the probit stick-breaking process (PSBP). A *logistic* stick-breaking process is discussed in detail in [34]. We employ the closely related probit version here because it may be easily implemented in a Gibbs sampler. We note that while the method in [34] is related to that discussed below, in [34] the concepts of learned dictionaries and beta-Bernoulli priors were not considered.

We augment the data as $\{\boldsymbol{x}_i, \boldsymbol{r}_i\}_{i=1,N}$, where $\boldsymbol{x}_i$ again represents pixel values from the ith image patch, and $\boldsymbol{r}_i \in \mathbb{R}^2$ represents the two-dimensional location of each patch. We wish to impose that proximate patches are more likely to be composed of the same or similar columns of $\mathbf{D}$. In the PSBP construction, all aspects of (7) are retained, except for the manner in which $\boldsymbol{z}_i$ are constituted. Rather than drawing a single K-dimensional vector of probabilities $\boldsymbol{\pi}$ as in (6), we draw a *library* of such vectors:

$$\boldsymbol{\pi}_l^* \sim \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \quad , \quad l = 1, \dots, N_L \tag{10}$$

and each $\boldsymbol{\pi}_l^*$ is associated with a particular segment in the image. One $\boldsymbol{\pi}_i$ is associated with location $\boldsymbol{r}_i$, and drawn

$$\boldsymbol{\pi}_i \sim \sum_{l=1}^{N_L} \beta_l(\boldsymbol{r}_i)\delta_{\boldsymbol{\pi}_l^*} \tag{11}$$

with $\sum_{l=1}^{N_L} \beta_l(\boldsymbol{r}_i) = 1$ for all $\boldsymbol{r}_i$, and $\delta_{\boldsymbol{\pi}_l^*}$ represents a point measure concentrated at $\boldsymbol{\pi}_l^*$. Once $\boldsymbol{\pi}_i$ is associated with a particular $\boldsymbol{x}_i$, the corresponding binary vector $\boldsymbol{z}_i$ is drawn as in the first line of (6). Note that the distinction between DP and PSBP is that in the former the mixture weights $\{\beta_l\}_{l=1,N_L}$ are independent of spatial position $\boldsymbol{r}$, while the latter explicitly utilizes $\boldsymbol{r}$ within $\{\beta_l(\boldsymbol{r})\}_{l=1,N_L}$ (and below we impose that $\beta_l(\boldsymbol{r})$ changes smoothly with $\boldsymbol{r}$).

[1] Proximity can be modeled as in “spatial proximity,” as here developed in detail, or “feature proximity” as in non-local means and related approaches, see [27] and references therein.

The space-dependent weights are constructed as $\beta_l(\boldsymbol{r}) = V_l(\boldsymbol{r})\prod_{h=l}^{l-1}[1 - V_h(\boldsymbol{r})]$ where $0 < V_l(\boldsymbol{r}) < 1$ constitute space-dependent probabilities. We set $V_{N_L} = 1$, and for $l \leq N_L - 1$ the V_l are space-dependent probit functions:

$$V_l(\boldsymbol{r}) = \int_{-\infty}^{g_l(\boldsymbol{r})} dx\mathcal{N}(x|0,1), \;\; g_l(\boldsymbol{r}) = \zeta_{l0} + \sum_{i=1}^{N} \zeta_{li}\mathcal{K}(\boldsymbol{r}, \boldsymbol{r}_i; \psi_l) \tag{12}$$

where $\mathcal{K}(\boldsymbol{r}, \boldsymbol{r}_i; \psi_l)$ is a kernel characterized by parameter ψ_l and $\{\zeta_{li}\}_{i=0,N}$ are a *sparse* set of real numbers. To implement the sparseness on $\{\zeta_{li}\}_{i=0,N}$, within the prior $\zeta_{li} \sim \mathcal{N}(0, \alpha_{li}^{-1})$, and (conjugate) $\alpha_{li} \sim \text{Gamma}(a_0, b_0)$, with (a_0, b_0) set to favor most α_{li} being large (if α_{li} is large, a draw $\mathcal{N}(0, \alpha_{li}^{-1})$ is likely to be near zero, such that most $\{\zeta_{li}\}_{i=0,N}$ are near zero). This sparseness-promoting construction is the same as that employed in the relevance vector machine (RVM) [39]. We here utilize a radial basis function (RBF) kernel $\mathcal{K}(\boldsymbol{r}, \boldsymbol{r}_i; \psi_l) = \exp[-\|\boldsymbol{r}_i - \boldsymbol{r}\|_2/\psi_l]$.

Each $g_l(\boldsymbol{r})$ is encouraged to only be defined by a small set of localized kernel functions, and via the probit link function $\int_{-\infty}^{g_l(\boldsymbol{r})} dx\mathcal{N}(x|0,1)$ the probability $V_l(\boldsymbol{r})$ is characterized by localized segments over which the probability $V_l(\boldsymbol{r})$ is contiguous and smoothly varying. The $V_l(\boldsymbol{r})$ constitute a space-dependent stick-breaking process. Since $V_{N_L} = 1$, $\sum_{l=1}^{N_L} \beta_l(\boldsymbol{r}) = 1$ for all $\boldsymbol{r}$.

The PSBP model is relatively simple to implement within a Gibbs sampler. For example, as indicated above, sparseness on ζ_{li} is imposed as in the RVM, and the probit link function is simply implemented within a Gibbs sampler (which is why it was selected, rather than a logistic link function). Finally, we define a finite set of possible kernel parameters $\{\psi_j\}_{j=1,N_p}$, and a multinomial prior is placed on these parameters, with the multinomial probability vector drawn from a Dirichlet distribution [34] (each of the $g_l(\boldsymbol{r})$ draws a kernel parameter from $\{\psi_j\}_{j=1,N_p}$). The model in (7), with the PSBP construction for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as PSBP-BPFA.

C. Discussion of proposed sparseness-imposing priors

The basic BPFA model is summarized in (7), and three related priors have been developed for the sparse binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$: (*i*) the basic truncated beta-Bernoulli process in (6), (*ii*) a DP-based clustering of the underlying $\{\boldsymbol{\pi}_i\}_{i=1,N}$, and (*iii*) a PSBP clustering of $\{\boldsymbol{\pi}_i\}_{i=1,N}$ that exploits knowledge of the location of the image patches. For (*ii*) and (*iii*), the $\boldsymbol{x}_i$ within a particular cluster have *similar* $\boldsymbol{z}_i$, rather than exactly the same binary vector; we also considered the latter, but this worked less well in practice. As discussed further when presenting results, for denoising and interpolation, all three methods yield comparable performance. However, for CS, (*ii*) and (*iii*) yield marked improvements in image-recovery accuracy relative to (*i*). In anticipation of these results, we provide a further discussion of the three priors on $\{\boldsymbol{z}_i\}_{i=1,N}$ and on the three image-processing problems under consideration.

For the denoising and interpolation problems, we are provided with the data $\{\boldsymbol{x}_i\}_{i=1,N}$, albeit in the presence of noise and potentially with substantial missing pixels. However, for this problem N may be made quite large, since we may consider all possible (overlapping) $B \times B$ patches. A given pixel (apart from near the edges of the image) is present in B^2 different patches. Perhaps because we have such a large quantity of partially overlapping data, for denoising and interpolation we have found that beta-Bernoulli process in (6) is sufficient for inferring the underlying relationships between the different data $\{\boldsymbol{x}_i\}_{i=1,N}$, and processing these data collaboratively. However, the beta-Bernoulli construction does not explicitly segment the image, and therefore an advantage of the PSBP-BPFA construction is that it yields comparable denoising and interpolation performance as (6), while also simultaneously yielding an effective image segmentation.

For the CS problem, we measure $\boldsymbol{y}_i = \boldsymbol{\Sigma}\boldsymbol{x}_i$, and therefore each of the n measurements associated with each image patch ($\boldsymbol{\Sigma} \in \mathbb{R}^{n \times P}$) loses the original pixels in $\boldsymbol{x}_i$ (the projection matrix $\boldsymbol{\Sigma}$ may also change with each patch, denoted $\boldsymbol{\Sigma}_i$). Therefore, for CS one cannot consider all possible shifts of the patches, as the patches are predefined and fixed in the CS measurement (in the denoising and interpolation problems the patches are defined in the subsequent analysis). Therefore, for CS imposition of the clustering behavior via DP or PSBP provides important information, yielding state-of-the-art CS-recovery results.

Before proceeding to the results, we also reconsider the theorem in Section II-B. It proved convenient, such that existing matrix-completion theory could be readily applied, to assume that the latent binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$ clustered. While that analysis demonstrated the promise of recovering missing pixels in natural images, our empirical results suggest that a more-thorough theoretical analysis of this problem is needed. Assume we measure $\boldsymbol{x}_i' = P_{\phi_i}(\boldsymbol{x}_i)$, where ϕ_i denotes the set of pixels observed for $\boldsymbol{x}_i$ (different subsets of pixels are observed for different patches, and by processing all pixels "collaboratively," we may infer the underlying dictionary $\mathbf{D}$ and the sparse $\boldsymbol{w}_i$). The logarithm of the posterior of (all) model parameters $\boldsymbol{\Theta}$, given observed data $\mathcal{D} = \{\boldsymbol{x}_i'\}_{i=1,N}$ and model hyperparameters $\mathcal{H}$, may be expressed as

$$\begin{aligned} -\log\ p(\boldsymbol{\Theta}|\mathcal{D},\mathcal{H}) \ &= \ \frac{\gamma_\epsilon}{2}\sum_{i=1}^{N}\|P_{\phi_i}(\boldsymbol{x}_i - \mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i))\|_2^2 + \frac{P}{2}\sum_{k=1}^{K}\|\boldsymbol{d}_k\|_2^2 + \frac{\gamma_s}{2}\sum_{i=1}^{N}\|\boldsymbol{s}_i\|_2^2 \\ &- \ \log f(\{\boldsymbol{z}_i\}_{i=1}^{N};\mathcal{H}) - \log \mathrm{Gamma}(\gamma_\epsilon|\mathcal{H}) - \log \mathrm{Gamma}(\gamma_s|\mathcal{H}) + Const. \end{aligned} \tag{13}$$

where density function $f(\{\boldsymbol{z}_i\}_{i=1}^{N};\mathcal{H})$ represents the particular prior placed on $\{\boldsymbol{z}_i\}_{i=1,N}$, and we have considered the beta-Bernoulli prior, as well as the DP and PSBP constructions. Note that the Gaussian assumption on $\boldsymbol{\epsilon}_i$ yields a Frobenius norm between the observed image data and that manifested by the model (constituted via $\sum_{i=1}^{N}\|P_{\phi_i}(\boldsymbol{x}_i - \mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i))\|_2^2$). Therefore, this construction is closely linked to the optimization approach advocated for near-low-rank matrix completion [7], which also uses a Frobenius norm. The key distinction, however, is manifested here via $f(\{\boldsymbol{z}_i\}_{i=1}^N; \mathcal{H})$, which moves beyond linear (low-rank) models to nonlinear constructions (the subspace used to represent missing data is patch-dependent).

V. Example Results

A. Reproducible research

The test results and the Matlab code to reproduce them can be downloaded from $http://www.ee.duke.edu/\sim mz1/Results/BPFAImage/$.

B. Parameter settings

For all BPFA, DP-BPFA and PSBP-BPFA computations, the dictionary truncation level was set at $K = 256$ or $K = 512$ based on the size of the image. Not all K dictionary elements are used in the model; the truncated beta-Bernoulli process infers the subset of dictionary elements employed to represent the data $\{\boldsymbol{x}_i\}_{i=1,N}$. The number of DP and PSBP sticks was set at $N_L = 20$. The library of PSBP parameters is defined as in [34]. The hyperparameters within the gamma distributions were set as $c = d = e = f = 10^{-6}$, as is typically done in models of this type [39] (the same settings were used for the gamma prior for the DP precision parameter α). The beta-distribution parameters are set as $a = K$ and $b = 1$ if random initialization is used or $a = K$ and $b = N/8$ if a singular value decomposition (SVD) based initialization is used. None of these parameters have been optimized or tuned. When performing inference, all parameters are initialized randomly (as a draw from the associated prior) or based on the SVD of the image under test. The Gibbs samplers for the BPFA, DP-BPFA and PSBP-BPFA have been found to mix and converge quickly, producing satisfactory results with as few as 20 iterations. The inferred images represent the average from the collection samples. All software was written in non-optimized Matlab. On a Dell Precision T3500 computer with a 2.4 GHz CPU, for $N = 148,836$ patches of size $8 \times 8 \times 3$ with 20% of the RGB pixels observed at random, the BPFA required about 2 minutes per Gibbs iteration (the DP version was comparable), and PSBP-BPFA required about 3 minutes per iteration. For the 106-band hyperspectral imagery, which employed $N = 428,578$ patches of size $4 \times 4 \times 106$ with 2% of the voxels observed uniformly at random, each Gibbs iteration required about 15 minutes.

C. Denoising

The BPFA denoising algorithm is compared with the original KSVD [13], for both grey-scale and color images. Newer denoising algorithms include block matching with 3D filtering (BM3D) [9], the multiscale

KSVD [29], and KSVD with the non-local mean constraints [26]. These algorithms assume the noise variance is known, while the proposed model automatically infers the noise variance from the image under test. Moreover, the BPFA, DP-BPFA and PSBP-BPFA models infer a potentially non-stationary variance, with a broad prior on the variance imposed by the gamma distribution. In the denoising examples we consider the BPFA model in (6); similar results are obtained via the DP-BPFA and PSBP-BPFA models discussed in Section IV.

In Table I we consider images from [13]. The proposed BPFA performs very similarly to KSVD. As one representative example of the model's ability to infer the noise variance, we consider the Lena image from Table I. The mean inferred noise standard deviations are 5.83, 10.59, 15.53, 20.48, 25.44, 50.46 and 100.54 for images contaminated by noise with respective standard deviations of 5, 10, 15, 20, 25, 50 and 100. Each of these noise variances were automatically inferred using exactly the same model, with no changes to the gamma hyperparameters.

In Table II we present similar results, for denoising RGB images; the KSVD comparisons come from [28]. An example denoising result is shown in Figure 1. As another example of the BPFA's ability to infer the underlying noise variance, for the castle image, the mean (automatically) inferred variances are 5.15, 10.18, 15.22 and 25.23 for images with additive noise with true respective standard deviations 5, 10, 15 and 25. The sensitivity of the KSVD algorithm to a mismatch between the assumed and true noise variances is shown in Figure 1 in [42], and the insensitivity of BPFA to changes in the noise variance and to requiring knowledge of the noise variance is deemed an important advantage.

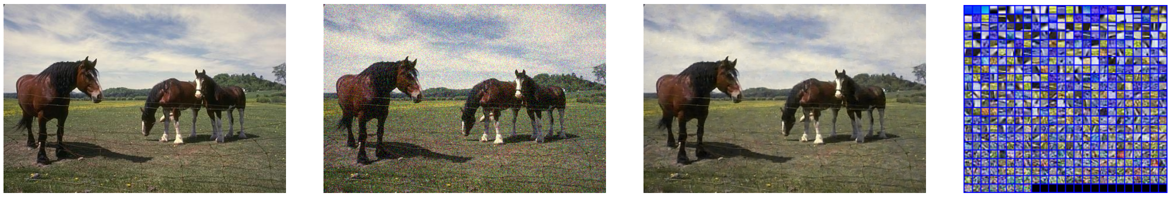


Fig. 1. From left to right: the original horses image, the noisy horses image with the noise standard deviation of 25, the denoised image and the inferred dictionary with its elements ordered in the probability to be used (from top-left). The low-probability dictionary elements are never used to represent $\{\boldsymbol{x}_i\}_{i=1,N}$, and are draws from the prior, showing the ability of the model to learn the number of dictionary elements needed for the data.

It is also important to note that the grey-scale KSVD results in Table I were initialized using an over-complete DCT dictionary, while the RGB KSVD results in Table II employed an extensive set of training imagery to learn a dictionary $\mathbf{D}$ that was used to initialize the denoising computations. All BPFA, DP-BPFA and PSBP-BPFA results employ no training data, with the dictionary initialized at random using draws from the prior or with the SVD of the data under test.

TABLE I

GREY-SCALE IMAGE DENOISING PSNR RESULTS, COMPARING KSVD [13] AND BPFA, USING PATCH SIZE 8×8. THE TOP AND BOTTOM PARTS OF EACH CELL ARE RESULTS OF KSVD AND BPFA, RESPECTIVELY.

σ	C.man	House	Peppers	Lena	Barbara	Boats	F.print	Couple	Hill
5	37.87	39.37	37.78	38.60	38.08	37.22	36.65	37.31	37.02
	37.32	39.18	37.24	38.20	37.94	36.43	36.29	36.77	36.24
10	33.73	35.98	34.28	35.47	34.42	33.64	32.39	33.52	33.37
	33.40	36.29	34.31	35.62	34.63	33.70	32.42	33.63	33.31
15	31.42	34.32	32.22	33.70	32.37	31.73	30.06	31.45	31.47
	31.34	34.52	32.46	33.93	32.61	31.97	30.23	31.73	31.64
20	29.91	33.20	30.82	32.38	30.83	30.36	28.47	30.00	30.18
	30.03	33.25	31.10	32.65	31.10	30.70	28.72	30.34	30.47
25	28.85	32.15	29.73	31.32	29.60	29.28	27.26	28.90	29.18
	28.99	32.24	30.00	31.63	29.88	29.70	27.58	29.28	29.57
50	25.73	27.95	26.13	27.79	25.47	25.95	23.24	25.32	26.27
	25.67	28.49	26.46	28.29	26.03	26.50	24.14	25.94	26.81
100	21.69	23.71	21.75	24.46	21.89	22.81	18.30	22.60	23.98
	21.93	24.37	22.73	24.95	22.13	23.32	20.44	23.01	24.22

D. Image interpolation

For the initial interpolation examples, we consider standard RGB images, with 80% of the RGB pixels missing uniformly at random (the data under test are shown in Figure 2). Results are first presented for the Castle and Mushroom images, with comparisons between the BPFA model in (6) and the PSBP-BPFA model discussed in Section IV. The difference between the two is that the former is a "bag-of-patches" model, while the latter accounts for the spatial locations of the patches. Further, the PSBP-BPFA simultaneously performs image recovery and segmentation. The results are shown in Figure 3, presenting the mean reconstructed images and inferred segmentations. Each color in the inferred segmentation represents one PSBP mixture component, and the figure shows the last Gibbs iteration (to avoid issues

TABLE II

RGB image denoising PSNR results comparing KSVD [28] and BPFA, both using a patch size of 7×7. The top and bottom parts of each cell show the results of KSVD and BPFA, respectively.

σ	Castle	Mushroom	Train	Horses	Kangroo
5	40.37	39.93	39.76	40.09	39.00
	40.34	39.73	39.38	39.96	39.00
10	36.24	35.60	34.72	35.43	34.06
	36.28	35.70	34.48	35.48	34.21
15	33.98	33.18	31.70	32.76	31.30
	34.04	33.41	31.63	32.98	31.68
25	31.19	30.26	28.16	29.81	28.39
	31.24	30.62	28.28	30.11	28.86

with label switching between Gibbs iterations). While the BPFA does not directly yield a segmentation, its PSNR results are comparable to those inferred by PSBP-BPFA, as summarized in Table III.

TABLE III

Comparison of interpolation of the Castle and Mushroom images, based upon observing 20% of the pixels, selected uniformly at random. Results are shown using BPFA and PSBP-BPFA, and the analysis is separately performed using $8 \times 8 \times 3$ and $5 \times 5 \times 3$ image patches.

	Castle $8 \times 8 \times 3$	Castle $5 \times 5 \times 3$	Mushroom $8 \times 8 \times 3$	Mushroom $5 \times 5 \times 3$
BPFA	29.32	28.48	31.63	31.17
PSBP-BPFA	29.54	28.46	32.03	31.27

An important additional advantage of Bayesian models like BPFA, DP-BPFA and PSBP-BPFA is that they provide a measure of confidence in the accuracy of the inferred image. In Figure 4 we plot the variance of the inferred error $\{\boldsymbol{\epsilon}_i\}_{i=1,N}$, computed via the Gibbs collection samples.

To provide a more-thorough examination of model performance, in Table IV we present results for several well-studied grey-scale and RGB images, as a function of the fraction of pixels missing. All of these results are based upon BPFA, with DP-BPFA and PSBP-BPFA yielding similar results. Finally, in Table V we perform interpolation and denoising simultaneously, again with no training data and without

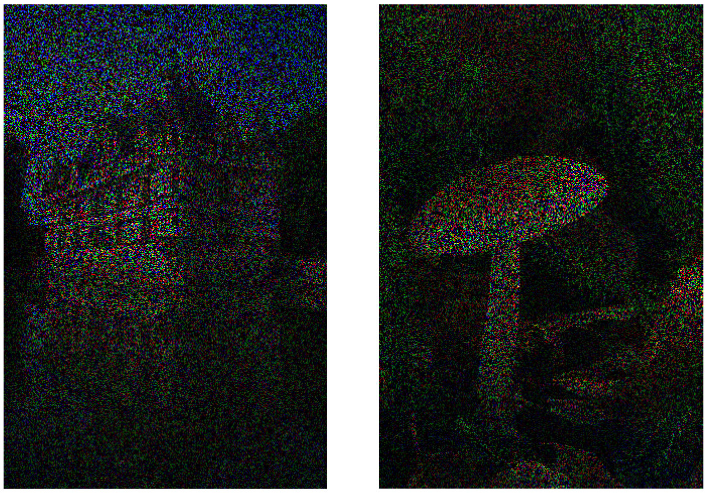


Fig. 2. Images with 80% of the RGB pixels missing at random. Left: castle image, right: mushroom image.

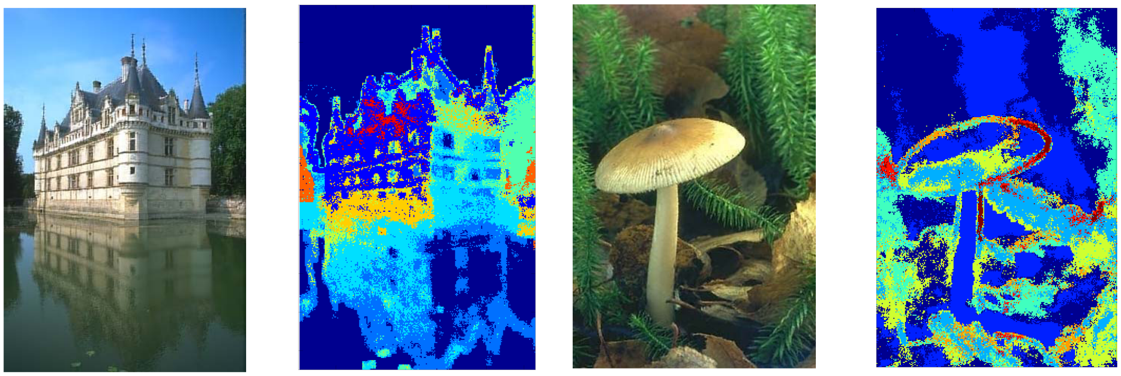


Fig. 3. PSBP-BPFA analysis with 80% of the RGB pixels missing uniformly at random (see Figure 2). The analysis is based on $8 \times 8 \times 3$ image patches, considering all possible (overlapping) parches. For a given pixel, the results are the average based upon all patches in which it is contained. For each example, recovered image based on an average of Gibbs collection samples (left), and each color representing one of the PSBP mixture components (right).

prior knowledge of the noise level. An example result is shown in Figure 5. To our knowledge, this is the first time such a result has been presented.

For all of the examples considered above, for both grey-scale and RGB images, we also attempted a direct application of matrix completion based on the incomplete matrix $\mathbf{X} \in \mathbb{R}^{P \times N}$, with columns defined by the image patches. We specifically considered the algorithm in [19], using software from Prof. Candès' website. For most of the examples considered above, even after very careful tuning of the

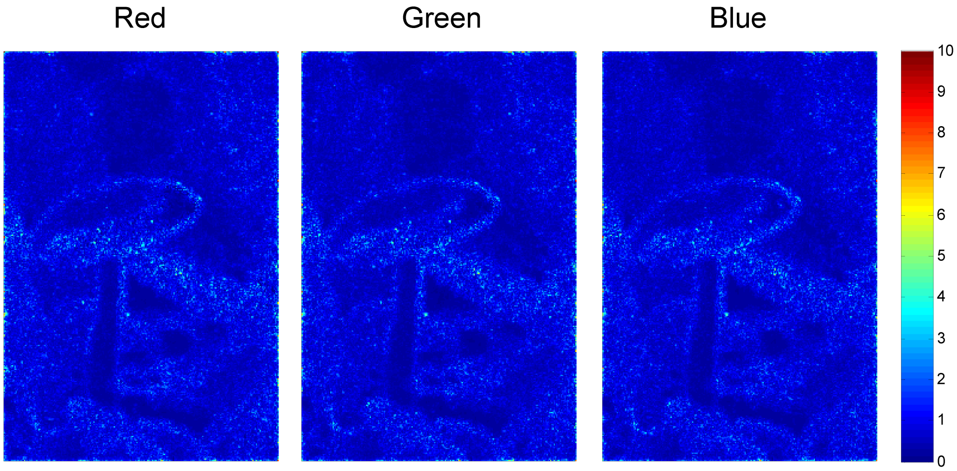


Fig. 4. Expected variance of each pixel for the (Mushroom) data considered in Figure 3.

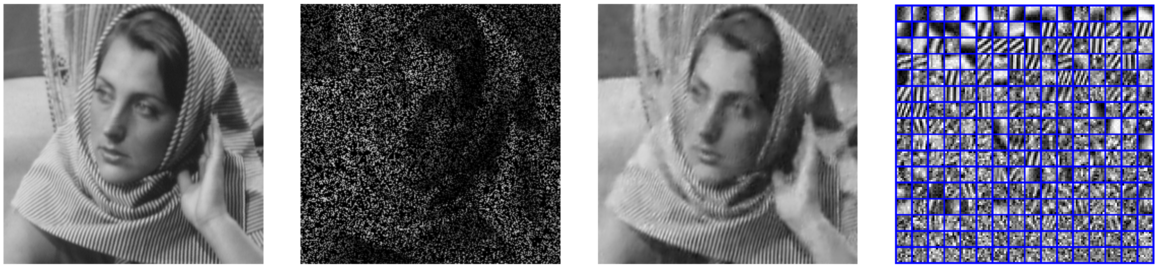


Fig. 5. From left to right: the original barbara256 image, the noisy and incomplete barbara256 image with the noise standard deviation of 15 and 70% of its pixels missing at random, the restored image and the inferred dictionary with its elements ordered in the probability to be used (from top-left).

parameters, the algorithm diverged, suggesting that the low-rank assumptions were violated. For examples for which the algorithm did work, the PSNR values were typically 4 to 5 dB worse than those reported here for our model.

E. Interpolation of hyperspectral imagery

The basic BPFA, DP-BPFA and PSBP-BPFA technology may also be applied to hyperspectral imagery, and it is here where these methods may have significant practical utility. Specifically, the amount of data that need be measured and read off a hyperspectral camera is often enormous. By selecting a small fraction of voxels for measurement and read-out, selected uniformly at random, the quantity of data that need be handled is reduced substantially. Further, one may simply modify existing hyperspectral cameras. We consider hyperspectral data with 106 spectral bands, measured by the US National Geospatial Agency

TABLE IV

Top: BPFA gray-scale image interpolation PSNR results, using patch size 8×8. Bottom: BPFA RGB image interpolation PSNR results, using patch size 7×7.

data ratio	C.man	House	Peppers	Lena	Barbara	Boats	F.print	Man	Couple	Hill
20%	24.11	30.12	25.92	31.00	24.80	27.81	26.03	28.24	27.72	29.33
30%	25.71	33.14	28.19	33.31	27.52	30.00	09.01	30.06	30.00	31.21
50%	28.90	38.02	32.58	36.94	33.17	33.78	33.53	33.29	35.56	34.23
80%	34.70	43.03	37.73	41.27	40.76	39.50	40.17	39.11	38.71	38.75

data ratio	Castle	Mushroom	Train	Horses	Kangroo
20%	29.12	31.56	24.59	29.99	29.59
30%	32.02	34.63	27.00	32.52	32.21
50%	36.45	38.88	32.00	37.27	37.34
80%	41.51	42.56	40.73	41.97	42.74

TABLE V

Simultaneous image denoising and interpolation PSNR results for BPFA, considering the Barbara256 image and using patch size 8×8.

σ	10%	20%	30%	50%	100%
0	23.47	26.87	29.83	35.60	42.94
5	23.34	26.73	29.27	33.61	37.70
10	23.16	26.07	28.17	31.17	34.31
15	22.66	25.17	26.82	29.31	32.14
20	22.17	24.27	25.62	27.90	30.55
25	21.68	23.49	24.72	26.79	29.30

(NGA). Because of the significant statistical correlation across the multiple spectral bands, the fraction of data that need be read is further reduced, relative to grey-scale or RGB imagery. In this example we considered 2% of the voxels, selected uniformly at random, and used image patches of size $4 \times 4 \times 106$. Other than the increased data dimensionality, nothing in the model was changed.

In Figure 6 we show example (mean) inferred images, at two (arbitrarily selected) spectral bands, as computed via BPFA. All 106 spectral bands are analyzed simultaneously. The average PSNR for the data cube (size $845 \times 512 \times 106$) is 30.96 dB. While the PSNR value is of interest, for data of this type the more important question concerns the ability to classify different materials based upon the hyperspectral data. In a separate forthcoming paper we consider classification based on the full datacube, and based upon the BPFA-inferred datacube using 2% of the voxels, with encouraging results reported. We also tried the low-rank matrix completion algorithm from [19] for the hyperspectral data, and even after extensive parameter tuning, the algorithm diverged for all hyperspectral data considered.

In Table VI we summarize algorithm performance on another hyperspectral data set, composed of 210 spectral bands. We show the PSNR values as a function of percentage of observed data, and as a function of the size of the image patch. Note that the 1×1 patches only exploit spectral information, while the other patch sizes exploit both spatial and spectral information.

TABLE VI

BPFA HYPERSPECTRAL IMAGE INTERPOLATION PSNR RESULTS. FOR THIS EXAMPLE THE TEST IMAGE IS A 150×150 URBAN IMAGE WITH 210 SPECTRAL BANDS. RESULTS ARE SHOWN AS A FUNCTION OF THE PERCENTAGE OF OBSERVED VOXELS, FOR DIFFERENT SIZED PATCHES ($e.g.$, THE 4×4 CASE CORRESPONDS TO $4 \times 4 \times 210$ "PATCHES").

Observed data (%)	1×1	2×2	3×3	4×4
2	15.34	21.09	22.72	23.46
5	17.98	23.58	25.30	25.88
10	20.41	25.27	26.36	26.68
20	22.22	26.50	27.02	27.16

F. Compressive sensing

We consider a CS example in which the image is divided into 8×8 patches, with these constituting the underlying data $\{\boldsymbol{x}_i\}_{i=1,N}$ to be inferred. For each of the N blocks, a vector of CS measurements $\boldsymbol{y}_i = \boldsymbol{\Sigma}\boldsymbol{x}_i$ is measured, where the number of projections per patch is n, and the total number of CS projections is $n \cdot N$. In our first examples the elements of $\boldsymbol{\Sigma}$ are constructed randomly, as draws from $\mathcal{N}(0,1)$; many other random projection classes may be considered [2] (and below we also consider optimized projections $\boldsymbol{\Sigma}$, matched to the dictionary $\mathbf{D}$). Each $\boldsymbol{x}_i$ is assumed represented in terms of a dictionary $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i$, and three constructions for $\mathbf{D}$ were considered: (i) a DCT expansion; (ii) learning of $\mathbf{D}$ using BPFA, using training images; (iii) using the BPFA to perform *joint* CS inversion and learning of $\mathbf{D}$. For (ii), the training data consisted of 4000 8×8 patches chosen at random from 100 images selected from the Microsoft database ($http : //research.microsoft.com/en - us/projects/objectclassrecognition$). The dictionary was set to $K = 256$, and the offline beta process inferred a dictionary of size $M = 237$.

Representative CS reconstruction results are shown in Figure 7 (left) based upon a DCT dictionary, for a grey-scale version of the "castle" image. The results in Figure 7 (right) are based on a learned dictionary; except for the "online BP" results (where $\mathbf{D}$ and $\{\boldsymbol{w}_i\}_{i=1,N}$ are learned jointly), all of these results employ the same dictionary $\mathbf{D}$ learned off-line as mentioned above, and the algorithms are distinguished by different ways of estimating $\{\boldsymbol{w}_i\}_{i=1,N}$. A range of CS-inversion algorithms are considered from the literature, and several BPFA-based constructions are considered as well for CS inversion. The online BPFA results (with no training data) are quite competitive with those based on a dictionary learned off-line.

Note that results based on a learned dictionary are markedly better than those based on the DCT; similar results were achieved when the DCT was replaced by a wavelet representation. For the DCT-based results, note that the DP-BPFA and PSBP-BPFA CS inversion results are significantly better than those of all other CS inversion algorithms. The results reported here are consistent with tests we performed using over 100 images from the aforementioned Microsoft database, not reported here in detail for brevity.

In all previous results the projection matrix $\mathbf{\Sigma}$ was constituted randomly. We now consider a simple means of matching $\mathbf{\Sigma}$ to a $\mathbf{D}$ learned offline, based upon representative training images. Assume a learned $\mathbf{D} \in \mathbb{R}^{P \times K}$, with $K > P$, which may be represented via SVD as $\mathbf{D} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$; $\mathbf{U} \in \mathbb{R}^{P \times P}$ and $\mathbf{V} \in \mathbb{R}^{K \times P}$ are each composed of orthonormal columns, and $\mathbf{\Lambda}$ is a $P \times P$ diagonal matrix. The columns of $\mathbf{U}$ span the linear subspace of $\mathbb{R}^P$ in which the columns of $\mathbf{D}$ reside. Further, since the columns of $\mathbf{D}$ are generally *not* orthonormal, each column of $\mathbf{D}$ is "spread out" when expanded in the columns of $\mathbf{U}$. Therefore, one expects that $\mathbf{U}$ and $\mathbf{D}$ are incoherent. Hence, a simple means of matching CS projections to the data is to define the rows of $\mathbf{\Sigma}$ in terms of randomly selected columns of $\mathbf{U}$. This was done in Figure 8 for the grey-scale "castle" image, using the same learned dictionary as considered in Figure 7. It is observed that this procedure yields a marked improvement in CS recovery accuracy, for all CS inversion algorithms considered.

Concerning computational costs, all CS inversions were run efficiently on PCs, with the specifics computational times dictated by the detailed Matlab implementation and the machine run on. A rough ranking of the computational speeds, from fastest to slowest, is as follows: StOMP-CFAR, Fast BCS, OMP, BPFA, LARS/Lasso, Online BPFA, DP-BPFA, PSBP-BPFA, VB BCS, Basis Pursuit; in this list, algorithms BPFA through Basis Pursuits have approximately the same computational costs.

VI. CONCLUSIONS

The truncated beta-Bernoulli process has been employed to learn dictionaries matched to image patches $\{\boldsymbol{x}_i\}_{i=1,N}$. The basic nonparametric Bayesian model is termed a beta process factor analysis (BPFA) framework, and extensions have also been considered. Specifically, the Dirichlet process (DP) has been employed to cluster the $\{\boldsymbol{x}_i\}_{i=1,N}$, encouraging similar dictionary-element usage within respective clusters. Further, the probit stick-breaking process (PSBP) has been used to impose that proximate patches are more likely to be clustered similarly (imposing that they are more probable to employ similar dictionary elements). All inference has been performed by a Gibbs sampler, with analytic update equations. The PBFA, DP-BPFA and PSBP-BPFA have been applied to three problems in image processing: (i) denoising, (ii) image interpolation based upon a subset of pixels selected uniformly at random, and (iii) learning dictionaries for compressive sensing and also compressive sensing inversion. We have also considered jointly performing (i) and (ii). Important advantages of the proposed methods are: (i) a full posterior on model parameters are inferred, and therefore "error bars" may be placed on the inverted images; (ii) the noise variance need not be known, and is inferred within the analysis and may be nonstationary; (iii) while training data may be used to initialize the dictionary learning, this is not needed, and the BPFA results are highly competitive even based upon random initializations. In the context of compressive sensing, the DP-BPFA and PSBP-BPFA results are state of the art, significantly better than existing published methods. Finally, based upon the learned dictionary, a simple method has been constituted for optimizing the CS projections.

The interpolation problem is related to CS, in that we exploit the fact that $\{\boldsymbol{x}_i\}_{i=1,N}$ reside on a low-dimensional nonlinear subspace of $\mathbb{R}^P$, such that the total number of measurements is small relative to $N \cdot P$ (recall $\boldsymbol{x}_i \in \mathbb{R}^P$). However, in CS one employs projection measurements $\boldsymbol{\Sigma}\boldsymbol{x}_i$, where $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times P}$, ideally with $n \ll P$. The interpolation problem corresponds to the special case in which the rows of $\boldsymbol{\Sigma}$ are randomly selected rows of the $P \times P$ identity matrix. This problem is closely related to the problem of matrix completion [6], [22], [35], where the incomplete matrix $\mathbf{X} \in \mathbb{R}^{P \times N}$ has columns defined by $\{\boldsymbol{x}_i\}_{i=1,N}$. However, the $\{\boldsymbol{x}_i\}_{i=1,N}$ reside in a *nonlinear* subspace of $\mathbb{R}^P$, while low-rank-based methods assume that the data reside in a low-dimensional *linear* subspace. We showed that if the $\{\boldsymbol{x}_i\}_{i=1,N}$ may be clustered, manifesting a union-of-subspace model, then matrix completion theory may be employed relatively simply to place bounds on anticipated accuracy of recovered missing data.

While the PSBP-BPFA successfully segmented the image while recovering missing data, we found that the PSNR performance of direct BPFA analysis performed very close to that of PSBP-BPFA. This suggests that the properties of the beta-Bernoulli process (recall the Indian buffet process discussion in Section III-A) naturally manifests an effective clustering of these data. An important area of future research is to extend the matrix-completion theory to the case for which the columns of $\mathbf{X}$ come from a general non-linear subspace of $\mathbb{R}^P$, such as that impose via the beta-Bernoulli prior.

APPENDIX: GIBBS SAMPLING INFERENCE

The Gibbs sampling update equations are given below; we provide the update equations for the BPFA, and the DP and PSBP versions are relatively simple extensions. Below, $\mathbf{\Sigma}_i$ represents the projection matrix on the data, for image patch $\boldsymbol{x}_i$. For the CS problem, $\mathbf{\Sigma}_i$ is typically fully populated, while for the interpolation problem each row of $\mathbf{\Sigma}_i$ is all zeros except for a single one, corresponding to the specific pixel that is measured. The update equations are the conditional probability of each parameter, conditioned on all other parameters in the model.

Sample $\boldsymbol{d}_k$

$$p(\boldsymbol{d}_k|-) \propto \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i; \mathbf{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i), \gamma_\epsilon^{-1} \mathbf{I}_{\|\mathbf{\Sigma}_i\|_0}) \mathcal{N}(\boldsymbol{d}_k; 0, P^{-1}\mathbf{I}_P)$$

It can be shown that $\boldsymbol{d}_k$ can be drawn from a normal distribution

$$p(\boldsymbol{d}_k|-) \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{d}_k}, \mathbf{\Sigma}_{\boldsymbol{d}_k})$$

with the covariance $\mathbf{\Sigma}_{\boldsymbol{d}_k}$ and mean $\boldsymbol{\mu}_{\boldsymbol{d}_k}$ expressed as

$$\mathbf{\Sigma}_{\boldsymbol{d}_k} = \left(P\mathbf{I} + \gamma_\epsilon \sum_{i=1}^{N} z_{ik}^2 s_{ik}^2 \mathbf{\Sigma}_i^T \mathbf{\Sigma}_i \right)^{-1}$$

$$\boldsymbol{\mu}_{\boldsymbol{d}_k} = \gamma_\epsilon \mathbf{\Sigma}_{\boldsymbol{d}_k} \sum_{i=1}^{N} z_{ik} s_{ik} \widetilde{\boldsymbol{x}}_i^{-k}$$

where

$$\widetilde{\boldsymbol{x}}_i^{-k} = \mathbf{\Sigma}_i^T \boldsymbol{y}_i - \mathbf{\Sigma}_i^T \mathbf{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i) + \mathbf{\Sigma}_i^T \mathbf{\Sigma}_i \boldsymbol{d}_k (s_{ik} \odot z_{ik}).$$

Sample $\boldsymbol{z}_{k:} = [z_{1k}, z_{2k}, \cdots, z_{Nk}]$

$$p(z_{ik}|-) \propto \mathcal{N}(\boldsymbol{y}_i; \mathbf{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i), \gamma_\epsilon^{-1} \mathbf{I}_{\|\mathbf{\Sigma}_i\|_0}) \text{Bernoulli}(z_{ik}; \pi_k)$$

The posterior probability that $z_{ik} = 1$ is proportional to

$$p_1 = \pi_k \exp\left[-\frac{\gamma_\epsilon}{2} (s_{ik}^2 \boldsymbol{d}_k^T \mathbf{\Sigma}_i^T \mathbf{\Sigma}_i \boldsymbol{d}_k - 2 s_{ik} \boldsymbol{d}_k^T \widetilde{\boldsymbol{x}}_i^{-k}) \right]$$

and the posterior probability that $z_{ik} = 0$ is proportional to

$$p_0 = 1 - \pi_k$$

so z_{ik} can be drawn from a Bernoulli distribution as

$$z_{ik} \sim \text{Bernoulli}(\frac{p_1}{p_0 + p_1}). \tag{14}$$

Sample $\boldsymbol{s}_{k:} = [s_{1k}, s_{2k}, \cdots, s_{Nk}]$

$$p(s_{ik}|-) \propto \mathcal{N}(\boldsymbol{y}_i; \boldsymbol{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i), \gamma_\epsilon^{-1} \mathbf{I}_{\|\boldsymbol{\Sigma}_i\|_0}) \mathcal{N}(\mathbf{s}_i; 0, \gamma_s^{-1} \mathbf{I}_K)$$

It can be shown that s_{ik} can be drawn from a normal distribution

$$p(s_{ik}|-) \sim \mathcal{N}(\mu_{s_{ik}}, \Sigma_{s_{ik}}) \tag{15}$$

with the variance $\Sigma_{s_{ik}}$ and mean $\mu_{s_{ik}}$ expressed as

$$\Sigma_{s_{ik}} = \left(\gamma_s + \gamma_\epsilon z_{ik}^2 \boldsymbol{d}_k^T \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \boldsymbol{d}_k\right)^{-1}$$

$$\mu_{s_{ik}} = \gamma_\epsilon \Sigma_{s_{ik}} z_{ik} \boldsymbol{d}_k^T \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \widetilde{\mathbf{x}}_i^{-k}.$$

Note z_{ik} is equal to either 1 or 0, $\Sigma_{s_{ik}}$ and $\mu_{s_{ik}}$ can be further expressed as

$$\Sigma_{s_{ik}} = \begin{cases} \left(\gamma_s + \gamma_\epsilon \boldsymbol{d}_k^T \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \boldsymbol{d}_k\right)^{-1} & \text{if } z_{ik} = 1 \\ \gamma_s^{-1} & \text{if } z_{ik} = 0 \end{cases}$$

$$\mu_{s_{ik}} = \begin{cases} \gamma_\epsilon \Sigma_{s_{ik}} \boldsymbol{d}_k^T \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \widetilde{\mathbf{x}}_i^{-k} & \text{if } z_{ik} = 1 \\ 0 & \text{if } z_{ik} = 0 \end{cases} .$$

Sample π_k

$$p(\pi_k|-) \propto \text{Beta}(\pi_k; a, b) \prod_{i=1}^{N} \text{Bernoulli}(z_{ik}; \pi_k)$$

It can be shown that π_k can be drawn from a Beta distribution as

$$p(\pi_k|-) \sim \text{Beta}(\frac{a}{K} + \sum_{i=1}^{N} z_{ik}, \frac{b_0(K-1)}{K} + N - \sum_{i=1}^{N} z_{ik})$$

Sample γ_s

$$p(\gamma_s|-) \propto \Gamma(\gamma_s; c_0, d_0) \prod_{i=1}^{N} \mathcal{N}(\mathbf{s}_i; 0, \gamma_s^{-1} \mathbf{I}_K)$$

It can be shown that γ_s can be drawn from a Gamma distribution as

$$p(\gamma_s|-) \sim \Gamma\left(c_0 + \frac{1}{2}KN, d_0 + \frac{1}{2}\sum_{i=1}^{N} \mathbf{s}_i^T \mathbf{s}_i\right)$$

Sample γ_ϵ

$$p(\gamma_\epsilon|-) \propto \Gamma(\gamma_\epsilon; e_0, f_0) \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i; \boldsymbol{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i), \gamma_\epsilon^{-1} \mathbf{I}_{\|\boldsymbol{\Sigma}_i\|_0}) \tag{16}$$

It can be shown that γ_ϵ can be drawn from a Gamma distribution as

$$p(\gamma_\epsilon|-) \sim \Gamma\left(e_0 + \frac{1}{2}\sum_{i=1}^{N} \|\boldsymbol{\Sigma}_i\|_0, f_0 + \frac{1}{2}\sum_{i=1}^{N} \|\boldsymbol{\Sigma}_i^T \boldsymbol{y}_i - \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i)\|_{\ell_2}\right). \tag{17}$$

Note that $\boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i$ is a sparse identity matrix, $\boldsymbol{\Sigma}_{\boldsymbol{d}_k}$ is a diagonal matrix, and $\mathbf{Z}$ is a sparse matrix, it is easy to find that only basic arithmetical operations are needed and many unnecessary calculations can be avoided, leading to fast computation and low memory requirement.

REFERENCES

[1] M. Aharon, M. Elad, and A. M. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54:4311–4322, 2006.

[2] R.G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24:118–124, 2007.

[3] A.M. Bruckstein, D.L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51:34–81, 2007.

[4] A. Buades, B. Coll, J.-M. Morel, and C. Sbert. Self-similarity driven color demosaicking. *IEEE Trans. Image Processing*, 18(6):1192–1202, 2009.

[5] E. Candès and T. Tao. Near-optimal signal recovery from random projections: universal encoding strategies? *IEEE Trans. Information Theory*, 52:5406–5425, 2006.

[6] E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans. Inform. Theory*, 2010.

[7] E.J. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 2010.

[8] E.J. Candès and B. Recht. Exact matrix completion via convex optimization. *Found. of Comput. Math.*, pages 717–772, 2008.

[9] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3d transform-domain collaborative filtering. *IEEE Trans. Image Processing*, 16:2007, 2007.

[10] D.L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306, 2006.

[11] M.F. Duarte, M.A. Davenport, D. Takhar, J.N. Laska, T. Sun, K.F. Kelly, and R.G. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 2008.

[12] J.M. Duarte-Carvajalino and G. Sapiro. Learning to sense sparse signals: simultaneous sensing matrix and sparsifying dictionary optimization. *IEEE Transactions on Image Processing*, pages 1395–1408, 2009.

[13] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Processing*, 15:3736–3745, 2006.

[14] M. Elad and I. Yavneh. A weighted average of sparse representations is better than the sparsest one alone. *Preprint*, 2010.

[15] Y.C. Eldar and M. Mishali. Robust recovery of signals from a structured union of subspaces. *IEEE Trans. Inform. Theory*, 2009.

[16] T. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230, 1973.

[17] S. Gleichman and Y.C. Eldar. Blind compressed sensing. *Preprint (on Arxiv.org).*

[18] T.L. Griffiths and Z. Ghahramani. Infinite latent feature models and the Indian buffet process. In *Proc. Advances in Neural Information Processing Systems*, pages 475–482, 2005.

[19] E.J. Candès J.-F. Cai and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, pages 1956–1982, 2008.

[20] S. Ji, Y. Xue, and L. Carin. Bayesian compressive sensing. *IEEE Trans. Signal Processing*, 56:2346–2356, 2008.

[21] D. Knowles and Z. Ghahramani. Infinite sparse factor analysis and infinite independent components analysis. In *Proc. International Conference on Independent Component Analysis and Signal Separation*, 2007.

[22] N.D. Lawrence and R. Urtasun. Non-linear matrix factorization with gaussian processes. In *Proc. International Conference on Machine Learning*, pages 601–608, 2009.

[23] Y.M. Lu and M.N. Do. A theory for sampling signals from a union of subspaces. *IEEE transactions on signal processing*, 2008.

[24] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proc. International Conference on Machine Learning*, 2009.

[25] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In *Proc. Neural Information Processing Systems*, 2008.

[26] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proc. International Conference on Computer Vision*, 2009.

[27] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *International Conference on Computer Vision*, 2009.

[28] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Trans. Image Processing*, 17:53–69, 2008.

[29] J. Mairal, G. Sapiro, , and M. Elad. Learning multiscale sparse representations for image and video restoration. *SIAM Multiscale Modeling and Simulation*, 7:214 – 241, 2008.

[30] J. Paisley and L. Carin. Nonparametric factor analysis with beta process priors. In *Proc. International Conference on Machine Learning*, 2009.

[31] R. Raina, A. Battle, H. Lee, B. Packer, and A.Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proc. International Conference on Machine Learning*, 2007.

[32] I. Ramirez and G. Sapiro. Universal sparse modeling. *arXiv:1003.2941*, 2010.

[33] M. Ranzato, C. Poultney, S. Chopra, and Y. Lecun. Efficient learning of sparse representations with an energy-based model. In *Proc. Neural Information Processing Systems*, 2006.

[34] L. Ren, L. Du, D. Dunson, and L. Carin. The logistic stick breaking process. *J. Machine Learning Research*, preprint.

[35] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proc. International Conference on Machine Learning*, pages 880–887, 2008.

[36] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[37] M. Shankar, N.P. Pitsianis, and D.J. Brady. Compressive video sensors using multichannel imagers. *Appl. Opt.*, 49, 2010.

[38] R. Thibaux and M.I. Jordan. Hierarchical beta processes and the indian buffet process. In *Proc. International Conference on Artificial Intelligence and Statistics*, 2007.

[39] M. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, June 2001.

[40] J. Wright, A.Y. Yang, A. Ganesh, S.S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Analysis Machine Intelligence*, 31:210–227, 2009.

[41] J. Yang, J. Wright, T. Huang, and Y. Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 2009.

[42] M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin. Non-parametric bayesian dictionary learning for sparse image representations. In *Proc. Neural Information Processing Systems*, 2009.

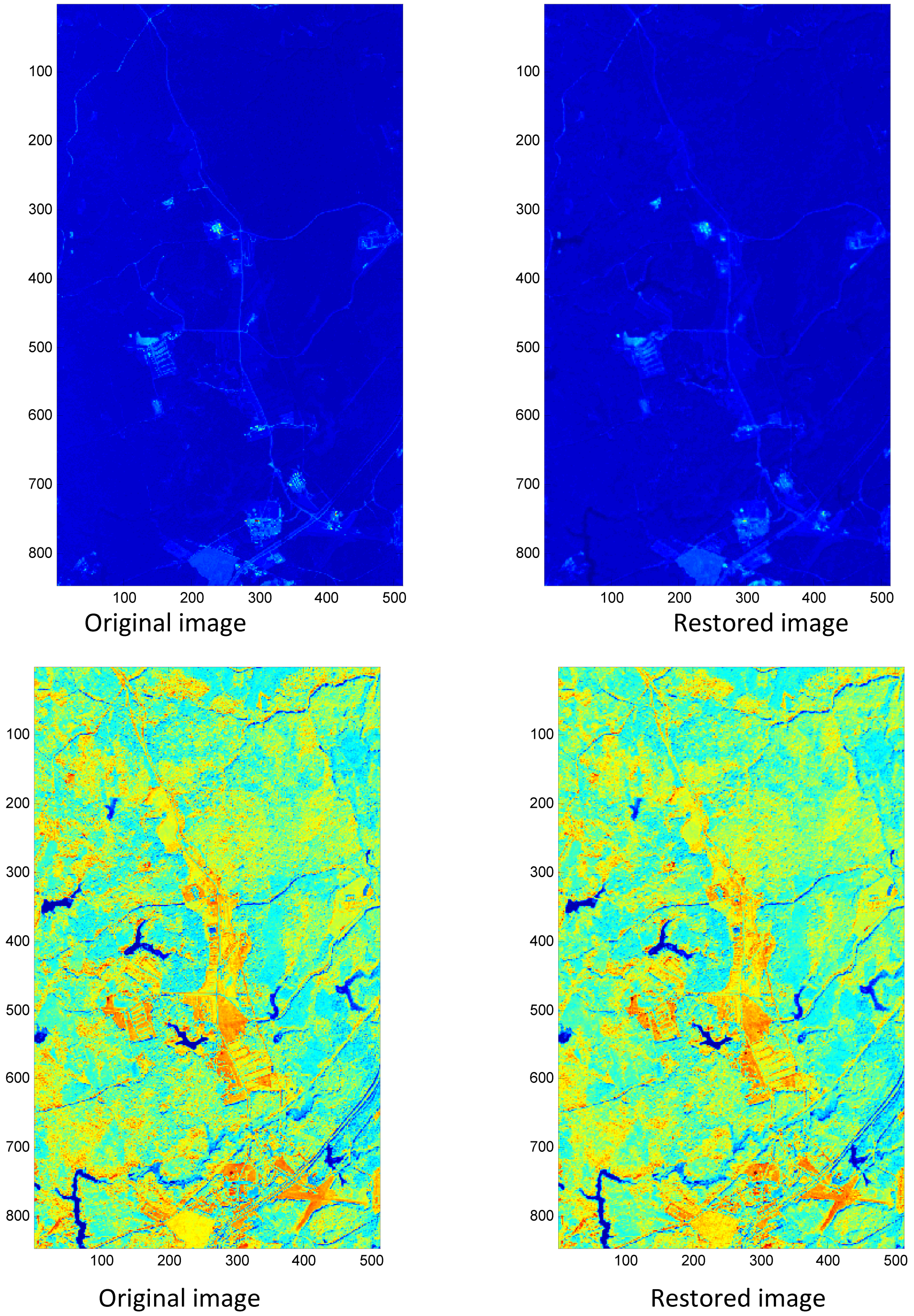


Fig. 6. Comparison of recovered band (average from Gibbs collection iterations) for hyperspectral imagery with 106 spectral bands. The interpolation is performed using 2% of the hyperspectral datacube, selected uniformly at random. The analysis employs $4 \times 4 \times 106$ patches. All spectral bands are analyzed at once, and here the data (recovered and original) are shown (arbitrarily) for bands 1 (top) and 50 (bottom). Results are computed using the BPFA model.

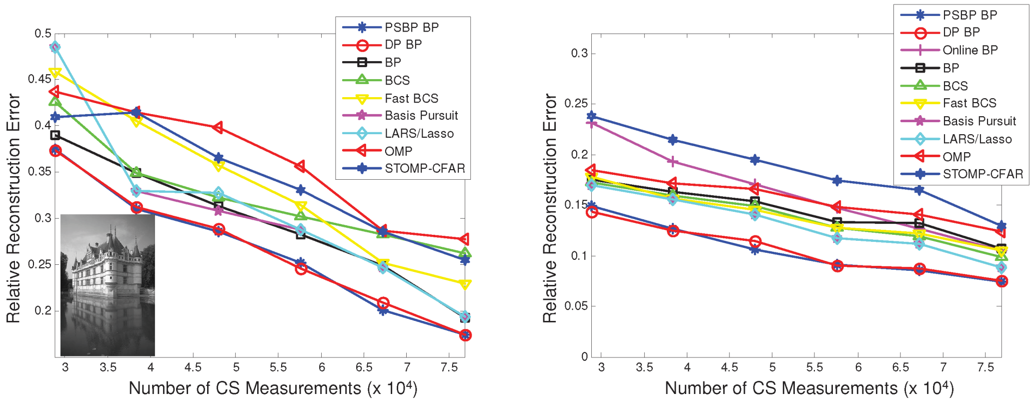


Fig. 7. Left: Compressive sensing (CS) results on grey-scale Castle image, based on a DCT dictionary $\mathbf{D}$. The CS projection matrix $\mathbf{\Sigma}$ is constituted randomly, with elements drawn iid from $\mathcal{N}(0, 1)$. Results are shown using the DP-BPFA and PSBP-BPFA models in Section IV. Comparisons are also made with several CS inversion algorithms from the literature. Right: Same as on the left but based on a learned dictionary $\mathbf{D}$ instead of DCT. The online BP results employ BPFA to learn $\mathbf{D}$ and do CS inversion jointly. All other results are based upon a learned $\mathbf{D}$ with learning performed offline using distinct training images.

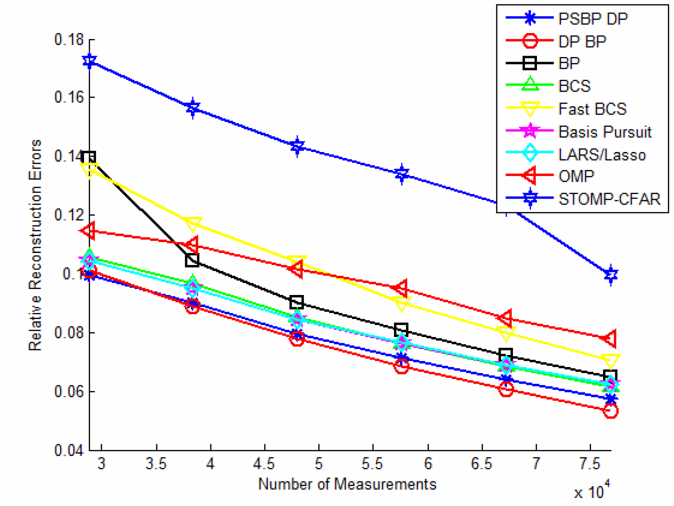


Fig. 8. Compressive sensing (CS) results on grey-scale Castle image, based on a learned dictionary $\mathbf{D}$ (learning performed offline, using distinct training data). The projection matrix $\mathbf{\Sigma}$ is matched to $\mathbf{D}$, based upon an SVD of $\mathbf{D}$.