Multi-Task Compressive Sensing

Shihao Ji, David Dunson†, and Lawrence Carin
Department of Electrical and Computer Engineering
†Institute of Statistics and Decision Sciences
Duke University, Durham, NC 27708-0291 USA
{shji,lcarin}@ece.duke.edu, dunson@stat.duke.edu

Abstract

Compressive sensing (CS) is a framework whereby one performs N *non-adaptive* measurements to constitute a vector $\boldsymbol{v} \in \mathbb{R}^N$, with $\boldsymbol{v}$ used to recover an approximation $\hat{\boldsymbol{u}} \in \mathbb{R}^M$ to a desired signal $\boldsymbol{u} \in \mathbb{R}^M$, with $N \ll M$; this is performed under the assumption that $\boldsymbol{u}$ is sparse in the basis represented by the matrix $\boldsymbol{\Psi} \in \mathbb{R}^{M \times M}$. It has been demonstrated that with appropriate design of the compressive measurements used to define $\boldsymbol{v}$, the decompressive mapping $\boldsymbol{v} \rightarrow \hat{\boldsymbol{u}}$ may be performed with error $\|\boldsymbol{u}-\hat{\boldsymbol{u}}\|_2^2$ having asymptotic properties analogous to those of the best *adaptive* transform-coding algorithm applied in the basis $\boldsymbol{\Psi}$. The mapping $\boldsymbol{v} \rightarrow \hat{\boldsymbol{u}}$ constitutes an inverse problem, often solved using ℓ_1 regularization or related techniques. In most previous research, if $L > 1$ sets of compressive measurements $\{\boldsymbol{v}_i\}_{i=1,L}$ are performed, each of the associated $\{\hat{\boldsymbol{u}}_i\}_{i=1,L}$ are recovered one at a time, independently. In many applications the L "tasks" defined by the mappings $\boldsymbol{v}_i \rightarrow \hat{\boldsymbol{u}}_i$ are not statistically independent, and it may be possible to improve the performance of the inversion if statistical inter-relationships are exploited. In this paper we address this problem within a multi-task learning setting, wherein the mapping $\boldsymbol{v}_i \rightarrow \hat{\boldsymbol{u}}_i$ for each task corresponds to inferring the parameters (here, wavelet coefficients) associated with the desired signal $\boldsymbol{u}_i$, and a shared prior is placed across all of the L tasks. Under this hierarchical Bayesian modeling, data from all L tasks contribute toward inferring a posterior on the hyperparameters, and once the shared prior is thereby inferred, the data from each of the L individual tasks is then employed to estimate the task-dependent wavelet coefficients. An empirical Bayesian procedure for the estimation of hyperparameters is considered; two fast inference algorithms extending the relevance vector machine (RVM) are developed. Example results on several data sets demonstrate the effectiveness and robustness of the proposed algorithms.

Index Terms

Compressive sensing (CS), multi-task learning, simultaneous sparse approximation, hierarchical Bayesian modeling, relevance vector machine (RVM).

I. INTRODUCTION

The development of wavelets [1], [2] has had a significant impact on several areas of signal processing and compression. An important characteristic of wavelets is the sparse manner in which they represent most natural signals. Specifically, let $\boldsymbol{u} \in \mathbb{R}^M$ represent the original signal, and let the matrix $\boldsymbol{\Psi} \in \mathbb{R}^{M \times M}$ represent a wavelet basis, then with the wavelet decomposition we have $\boldsymbol{u} = \boldsymbol{\Psi}\boldsymbol{\theta}$, where $\boldsymbol{\theta} \in \mathbb{R}^M$ is the wavelet coefficients. If we further let $\hat{\boldsymbol{u}}_m = \boldsymbol{\Psi}\boldsymbol{\theta}_m$ represent an approximation to $\boldsymbol{u}$, where $\boldsymbol{\theta}_m$ is the same as $\boldsymbol{\theta}$ except that the $M - m$ smallest coefficients are set to zero. The compressive properties of wavelets assure that $\|\boldsymbol{u} - \hat{\boldsymbol{u}}_m\|_2^2$ is typically small for $m \ll M$, thereby motivating the use of wavelets in a new generation of compression techniques for images and video [3], [4].

While wavelets have had a profound impact on practical compression schemes, there are issues that warrant further investigation. For example, while most natural signals are highly compressible in a wavelet basis, the specific m wavelet coefficients that have largest amplitude vary strongly from signal to signal. The aforementioned compression techniques must therefore adapt to each new signal under test, this constituting the principal complexity of wavelet-based compression algorithms. Of more practical importance, while the approximated signal $\hat{\boldsymbol{u}}_m$ is highly compressed ($m \ll M$), one first had to measure the M-dimensional signal $\boldsymbol{u}$, and in some sense $M - m$ pieces of data were measured unnecessarily. This latter issue raises the following question: Is it possible to measure the informative part of the signal directly, such that most unnecessary measurements are avoided from the start? This question has recently been answered in the affirmative, with this spawning the new field of compressive sensing (CS) [5], [6].

In the framework of CS, when performing measurements, one does not attempt to directly measure the m dominant wavelet coefficients, as this would require adapting to each new signal. Rather, in a CS measurement one implicitly measures *all* of the wavelet coefficients, with each compressive measurement performed by projecting the signal of interest $\boldsymbol{u}$ on a "random" basis that is constituted with "random" linear combination of the basis functions in $\boldsymbol{\Psi}$ [5], [6].[1] Each random projection corresponds to one CS measurement, and N such measurements constitute the overall CS measurement vector $\boldsymbol{v} \in \mathbb{R}^N$. Written in matrix notation, the CS measurements may be expressed as $\boldsymbol{v} = \boldsymbol{\Phi}\boldsymbol{\Psi}^T\boldsymbol{u} = \boldsymbol{\Phi}\boldsymbol{\theta}$, where $\boldsymbol{\Phi} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_N]^T$ is a $N \times M$ projection matrix, and $\boldsymbol{r}_j \in \mathbb{R}^M$ is the jth aforementioned random weights. The mapping from the CS data $\boldsymbol{v}$ to an approximation of the underlying signal $\boldsymbol{u}$, with the approximation represented as $\hat{\boldsymbol{u}}$, is under-determined, since typically $N \ll M$. However, by exploiting

[1]It is worth emphasizing that the CS framework is not limited to wavelet-based representations, and it is applicable to any signal representation (e.g., Fourier, Gabor, etc.) for which most of the basis-function coefficients are small, implying a sparse representation.

the fact that $\boldsymbol{u}$ is compressible in the basis $\boldsymbol{\Psi}$ (i.e., most of the elements in $\boldsymbol{\theta}$ are equal – or close – to zero), then one may approximate $\boldsymbol{\theta}$ (and therefore $\boldsymbol{u}$) accurately by solving an ℓ_1-regularized formulation [5], [6]:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \{\|\boldsymbol{v} - \boldsymbol{\Phi}\boldsymbol{\theta}\|_2^2 + \lambda\|\boldsymbol{\theta}\|_1\}, \tag{1}$$

where the scalar λ controls the relative importance applied to the Euclidian error and the sparseness term (the first and second expressions, respectively, inside the brackets in (1)). This basic framework has been the starting point for several recent CS inversion algorithms, including linear programming [7] and greedy algorithms [8]–[11] for sparse signal approximation. It has been demonstrated that in the asymptotic limit (large N, and $M > N$) the optimal $\hat{\boldsymbol{u}} = \boldsymbol{\Phi}\hat{\boldsymbol{\theta}}$ estimated from the *non-adaptive* CS measurements $\boldsymbol{v}$ has error $\|\boldsymbol{u} - \hat{\boldsymbol{u}}\|_2^2$ proportional to that of $\|\boldsymbol{u} - \hat{\boldsymbol{u}}_m\|_2^2$ [5], [6].

A. Problem Statement

While there have been numerous techniques developed to constitute the inverse CS mapping $\boldsymbol{v} \rightarrow \hat{\boldsymbol{u}}$, typically these algorithms perform the inversion separately and independently for each compressive measurement $\boldsymbol{v}$. In practice one may perform multiple sets of CS measurements; $L > 1$ sets of measurements are denoted $\{\boldsymbol{v}_i\}_{i=1,L}$. One may anticipate that many of the measurements in $\{\boldsymbol{v}_i\}_{i=1,L}$ are statistically related, particularly when repeated measurements are taken of similar scenes or for the same type of diagnostic task (e.g., repeated MRI images performed in a CS setting). By exploiting the statistical relationships between these L sets of measurements, one may hope to constitute the L mappings $\boldsymbol{v}_i \rightarrow \hat{\boldsymbol{u}}_i$ with fewer total measurements. Specifically, if N_i CS measurements are performed for the ith task to perform the *independent* mapping $\boldsymbol{v}_i \rightarrow \hat{\boldsymbol{u}}_i$ with desired accuracy, then ideally less than $\sum_{i=1}^{L} N_i$ total CS measurements would be required by exploiting statistical inter-relationships between the L sensing "tasks". In this paper, we term this simultaneous inversion of multiple related signals as multi-task CS, and develop the algorithms based on a hierarchical Bayesian model for this problem.

While our motivation to multi-task CS is mainly from a machine-learning perspective, related research has been previously studied in signal processing under the name of "Simultaneous Sparse Approximation" (SSA) [12]–[17] and more recently in compressive sensing with the name "Distributed Compressed Sensing" (DCS) [18]. At the end of the paper a brief review of this related research is provided, with connections to the work presented here.

B. General Framework

The mapping $\boldsymbol{v}_i \rightarrow \hat{\boldsymbol{u}}_i$ may be framed as a sparse linear-regression problem [19]–[22] and solved by a Bayesian algorithm [10]. Specifically, given the projection matrix $\boldsymbol{\Phi}_i \in \mathbb{R}^{N_i \times M}$, the linear relationship between the CS measurements $\boldsymbol{v}_i$ and the underlying signal $\boldsymbol{u}_i$ is known, with this represented in the system of equations: $\boldsymbol{v}_i = \boldsymbol{\Phi}_i \boldsymbol{\Psi}^T \boldsymbol{u}_i = \boldsymbol{\Phi}_i \boldsymbol{\theta}_i$. Furthermore, as it was assumed at the start, $\boldsymbol{\theta}_i \in \mathbb{R}^M$ is sparse in the basis $\boldsymbol{\Psi}$ (i.e., most components of $\boldsymbol{\theta}_i$ vanish, or at least may be set to zero with minimal impact on the reconstruction of $\boldsymbol{u}_i$). Therefore, the CS inversion problem is constituted in terms of solving for $\boldsymbol{\theta}_i$ such that $\boldsymbol{v}_i = \boldsymbol{\Phi}_i \boldsymbol{\theta}_i$, under the constraint that $\boldsymbol{\theta}_i$ is sparse. This may be expressed from a Bayesian standpoint that we have a prior belief that $\boldsymbol{\theta}_i$ should be sparse, data $\boldsymbol{v}_i$ are observed from compressive measurements, and the objective is to provide a posterior belief (density function) for the values of the weights $\boldsymbol{\theta}_i$ [10]. Of particular relevance to multi-task CS, this Bayesian perspective of CS is found to be very useful.

Each of the CS measurements $\{\boldsymbol{v}_i\}_{i=1,L}$ yields a corresponding regression "task" $\boldsymbol{v}_i \rightarrow \hat{\boldsymbol{\theta}}_i$, and performing multiple such learning tasks has been referred to in the machine-learning community as multi-task learning [23], which aims at sharing information effectively among multiple related tasks. Typical approaches to information sharing among tasks include: sharing hidden nodes in neural networks [24], [25], placing a common prior in hierarchical Bayesian models [26]–[28], sharing a common structure on the predictor space [29], and structured regularization in kernel methods [30], among others. Because of the relationship of the CS inversion to linear regression, hierarchical Bayesian models are a natural and convenient framework for multi-task learning of CS.

Hierarchical Bayesian models are one of the most important approaches for multi-task learning [31]–[35]. Such representations provide the flexibility to model both the individuality of tasks (experiments), and the correlations between tasks. To demonstrate the idea, a graphical model representation of multi-task CS is illustrated in Fig. 1, with detailed parameters to be introduced in the next section. In this formulation usually the bottom layer of the hierarchy is composed of individual models with task-specific parameters. On the layer above, tasks are connected together via a common prior placed on those parameters; on a layer above is a hyper-prior, invoked on parameters of the prior at the level below. This model can achieve efficient information-sharing between tasks for the following reason. Learning of the common prior is a part of the training process, and data from all tasks contribute to learning the common prior, thus making it possible to transfer information between tasks (via sufficient statistics). Given the prior, individual models are learned independently. As a result, the estimation of a regressor (task) is affected

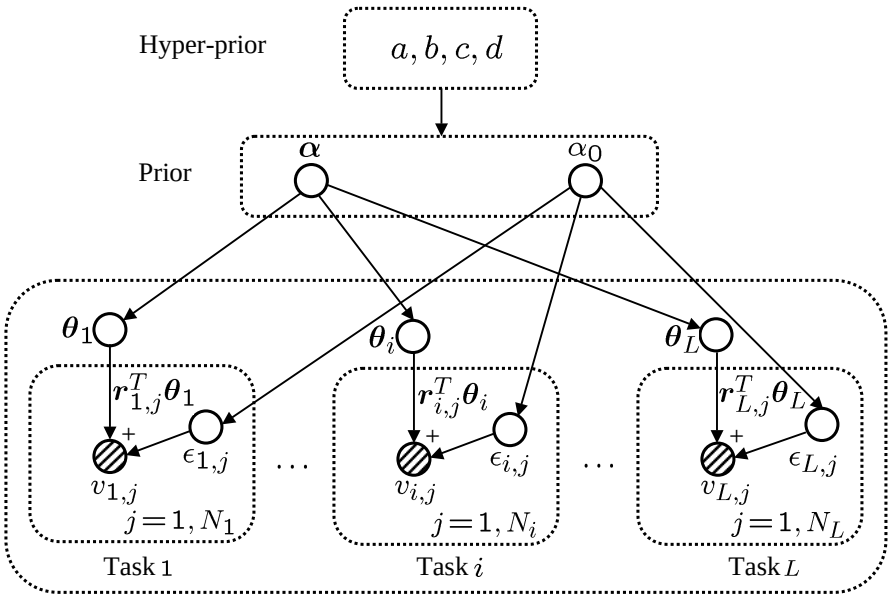


Fig. 1. A hierarchical Bayesian model representation of the mutli-task CS, where $\boldsymbol{\Phi}_i = [\boldsymbol{r}_{i,1}, \boldsymbol{r}_{i,2}, \ldots, \boldsymbol{r}_{i,N_i}]^T$ is the projection matrix of task i, each row of which is a set of random weights drawn i.i.d. from a zero-mean Gaussian distribution. The detailed parameters are to be introduced in Sec. II.

by both its own training data and by data from the other tasks related through the common prior, and the inter-relationships among the tasks are determined automatically through the joint learning.

While the formulation is constituted in a fully Bayesian setting, solving it efficiently (i.e., finding a posterior density function on $\boldsymbol{\alpha}$ and α_0 in Fig. 1) can be challenging (see [36] for an example). Therefore, an empirical Bayesian procedure is employed for the fast point estimate of the hyperparameters $\boldsymbol{\alpha}$ and α_0 in Fig. 1. This yields a computationally efficient multi-task CS inference algorithm that extends previous research in the Bayesian CS analysis [10], wherein the Bayesian inversion $\boldsymbol{v}_i \rightarrow \hat{\boldsymbol{\theta}}_i$ was performed one task at a time (single-task learning, i.e., $L = 1$).

In addition to a hierarchal Bayesian model of multi-task CS and a fast inference algorithm, a modified sparse linear-regression model is developed, of interest both for the single-task and multi-task CS settings. As discussed further below, this extension analytically integrates out the noise-variance term in the regression model, and it yields improved robustness over the previous formulation.

The remainder of the paper is structured as follows. In Sec. II we introduce a hierarchical Bayesian model for multi-task CS that builds naturally upon previous research on Bayesian CS [10]; a fast sequential optimization algorithm based on an empirical Bayesian procedure is developed for inference. In Sec. III we propose a modified sparse linear-regression model by marginalizing the noise variance, and develop

a fast inference algorithm as well. Example results on multiple datasets are presented in Sec. IV. A review of work related to multi-task CS is provided in Sec. V, followed in Sec. VI by conclusions and a discussion of future work.

II. Hierarchical Multi-Task CS Modeling

A. Bayesian Regression Formulation

Assume that L sets of CS measurements are performed, with these multiple sensing tasks statistically inter-related, as defined precisely below. The L sets of measurements are represented as $\{\boldsymbol{v}_i\}_{i=1,L}$, where $\boldsymbol{v}_i = \boldsymbol{\Phi}_i \boldsymbol{\Psi}^T \boldsymbol{u}_i = \boldsymbol{\Phi}_i \boldsymbol{\theta}_i$, and in general each measurement vector $\boldsymbol{v}_i \in \mathbb{R}^{N_i}$ employs a different random projection matrix $\boldsymbol{\Phi}_i \in \mathbb{R}^{N_i \times M}$, for $i = 1, 2, \ldots, L$. This generalizes the formulation considered in [12]–[15], [17], wherein a single $\boldsymbol{\Phi}$ is employed across all the L tasks. In the context of a regression analysis, we assume [10]

$$\boldsymbol{v}_i = \boldsymbol{\Phi}_i \boldsymbol{\theta}_i + \boldsymbol{\epsilon}_i, \tag{2}$$

where $\boldsymbol{\epsilon}_i \in \mathbb{R}^{N_i}$ is a residual error vector, modeled as N_i i.i.d. draws of a zero-mean Gaussian random variable with unknown precision α_0 (variance $1/\alpha_0$). The likelihood function for the parameters $\boldsymbol{\theta}_i$ and α_0, based on the observed data $\boldsymbol{v}_i$, may therefore be expressed as

$$p(\boldsymbol{v}_i | \, \boldsymbol{\theta}_i, \alpha_0) = (2\pi/\alpha_0)^{-N_i/2} \exp\left(-\frac{\alpha_0}{2} \|\boldsymbol{v}_i - \boldsymbol{\Phi}_i \boldsymbol{\theta}_i\|_2^2\right). \tag{3}$$

The parameters $\boldsymbol{\theta}_i$ (here, wavelet coefficients) characteristic of task i are assumed to be drawn from a product of zero-mean Gaussian distributions that are shared by all tasks, and it is in this sense that the L tasks are statistically related. Specifically, letting $\theta_{i,j}$ represent the jth wavelet (or scaling function) coefficient for CS task i, we have

$$p(\boldsymbol{\theta}_i | \, \boldsymbol{\alpha}) = \prod_{j=1}^{M} \mathcal{N}(\theta_{i,j} \mid 0, \alpha_j^{-1}), \tag{4}$$

where $\mathcal{N}(\cdot \mid 0, \alpha_j^{-1})$ is a zero-mean Gaussian density function with precision α_j. It is important to note that the hyperparameters $\boldsymbol{\alpha} = \{\alpha_j\}_{j=1,M}$ are shared among all L tasks, and therefore the data from all CS measurements $\{\boldsymbol{v}_i\}_{i=1,L}$ will contribute to learning the hyperparameters, offering the opportunity to adaptively borrow strength from the different measurements to a degree controlled by $\boldsymbol{\alpha}$.

To promote sparsity over the weights $\boldsymbol{\theta}_i$, Gamma priors are placed on the hyperparameters $\boldsymbol{\alpha}$, and

similarly on the noise precision α_0:

$$p(\alpha_0|\, a, b) = \text{Ga}(\alpha_0|\, a, b) = \frac{b^{\,a}}{\Gamma(a)}\, \alpha_0^{(a-1)} \exp(-b\alpha_0), \tag{5}$$

$$p(\boldsymbol{\alpha}|\, c, d) = \prod_{j=1}^{M} \text{Ga}(\alpha_j|\, c, d). \tag{6}$$

It has been demonstrated [19] that appropriate choice of parameters c and d encourages a sparse representation for the coefficients in the vector $\boldsymbol{\theta}_i$, where here this concept is extended to a multi-task CS setting. Typically, when $c = d = \epsilon$, with $\epsilon > 0$ a small constant, $\text{Ga}(\cdot|\, c, d)$ has a large spike concentrated at zero and a heavy right tail. The spike corresponds to basis functions for which there is essentially no borrowing of information. Such basis functions characterize components that are idiosyncratic to specific signals. At the other extreme, basis functions for which α_j is in the right tail have coefficients that are shrunk strongly to zero for all tasks, favoring sparseness, while borrowing information about which basis functions are not important for any of the signals in the collection. For small ϵ, there will be many such basis functions. As a default choice which avoids subjective choice of c, d and leads to computational simplifications, we set $c = d = 0$. For the Gamma prior on the noise precision α_0, we also let $a = b = 0$ as a default choice. This choice corresponds to a commonly-used improper prior expressing *a priori* ignorance about plausible values for the residual precision.[2] With these parametric definitions, a graphical model representation of multi-task CS is illustrated in Fig. 1.

Given the L sets of CS measurements $\{\boldsymbol{v}_i\}_{i=1,L}$ from the (assumed) statistically related sources, by applying the Bayes' rule, one may in principle infer a posterior density function on the hyperparameters $\boldsymbol{\alpha}$ and the noise precision α_0,

$$p(\boldsymbol{\alpha}, \alpha_0|\, \{\boldsymbol{v}_i\}_{i=1,L}, a, b, c, d) = \frac{p(\alpha_0|\, a, b)p(\boldsymbol{\alpha}|\, c, d) \prod_{i=1}^{L} \int d\boldsymbol{\theta}_i p(\boldsymbol{v}_i|\, \boldsymbol{\theta}_i, \alpha_0) p(\boldsymbol{\theta}_i|\, \boldsymbol{\alpha})}{\int d\boldsymbol{\alpha} \int d\alpha_0 p(\alpha_0|\, a, b)p(\boldsymbol{\alpha}|\, c, d) \prod_{i=1}^{L} \int d\boldsymbol{\theta}_i p(\boldsymbol{v}_i|\, \boldsymbol{\theta}_i, \alpha_0) p(\boldsymbol{\theta}_i|\, \boldsymbol{\alpha})}, \tag{7}$$

where the integral in (7) with respect to $\boldsymbol{\alpha}$ is actually an M-dimensional integral, with each integral linked to one component of $\boldsymbol{\alpha}$; similarly, each integral with respect to $\boldsymbol{\theta}_i$ is an M-dimensional integral, over all wavelet-coefficient weights. To avoid the complexity of evaluating some of these integrals[3], particularly those with respect to $\boldsymbol{\alpha}$ and α_0, we seek a point estimate for the parameters $\boldsymbol{\alpha}$ and α_0, and a maximum

[2]While the sparsity analysis provided here is largely following that of RVM [19], which is intuitive and conceptual, a more recent and rigorous analysis of sparse Bayesian learning and its superior performance on sparse representation can be found at [37], [38]. More relevantly, it is the log-det term of the likelihood (13) that produces sparsity.

[3]A variational Bayesian approach [36] can be applied to compute an approximation to these integrals, with the same computational cost as a MAP solution.

a posteriori (MAP) estimate for $\boldsymbol{\alpha}$ and α_0 is found as

$$\{\boldsymbol{\alpha}^{\text{MAP}}, \alpha_0^{\text{MAP}}\} = \arg\max_{\boldsymbol{\alpha},\alpha_0} \left(\log p(\alpha_0|\, a, b) + \log p(\boldsymbol{\alpha}|\, c, d) + \sum_{i=1}^{L} \log \int d\boldsymbol{\theta}_i \, p(\boldsymbol{v}_i|\, \boldsymbol{\theta}_i, \alpha_0) \, p(\boldsymbol{\theta}_i|\, \boldsymbol{\alpha}) \right), \tag{8}$$

which reduces to the simplified form in the limit as $a, b, c, d \rightarrow 0$ [4]:

$$\{\boldsymbol{\alpha}^{\text{ML}}, \alpha_0^{\text{ML}}\} = \arg\max_{\boldsymbol{\alpha},\alpha_0} \sum_{i=1}^{L} \log \int d\boldsymbol{\theta}_i \, p(\boldsymbol{v}_i|\, \boldsymbol{\theta}_i, \alpha_0) \, p(\boldsymbol{\theta}_i|\, \boldsymbol{\alpha}), \tag{9}$$

which can be interpreted as a MAP estimate under an improper, default prior or as a maximum likelihood (ML) estimate. The strategy that estimates a point estimate for $\boldsymbol{\alpha}$ and α_0 via (9) is related to extensive research in statistics on *empirical Bayesian analysis* [39]. Specifically, in the empirical Bayesian context, a data-dependent (and hence "empirical") prior on the weights $\boldsymbol{\theta}_i$ is invoked, and the hyperparameters of this "empirical" prior are estimated solely from the observed data by integrating out the unknown weights $\boldsymbol{\theta}_i$. This strategy has also been called *evidence maximization* or *type-II maximum likelihood* to describe the optimization process [19], [40].

Once the point estimates for $\boldsymbol{\alpha}$ and α_0 have been constituted by the ML approximation (9), the posterior density function for the coefficients $\boldsymbol{\theta}_i$ can be evaluated analytically. In particular, by Bayes' rule, using (3) and (4), we have

$$p(\boldsymbol{\theta}_i|\, \boldsymbol{v}_i, \boldsymbol{\alpha}, \alpha_0) = \frac{p(\boldsymbol{v}_i|\, \boldsymbol{\theta}_i, \alpha_0) \, p(\boldsymbol{\theta}_i|\, \boldsymbol{\alpha})}{\int d\boldsymbol{\theta}_i \, p(\boldsymbol{v}_i|\, \boldsymbol{\theta}_i, \alpha_0) \, p(\boldsymbol{\theta}_i|\, \boldsymbol{\alpha})} = \mathcal{N}(\boldsymbol{\theta}_i|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{10}$$

with mean and covariance given by

$$\boldsymbol{\mu}_i = \alpha_0 \boldsymbol{\Sigma}_i \boldsymbol{\Phi}_i^T \boldsymbol{v}_i, \tag{11}$$

$$\boldsymbol{\Sigma}_i = (\alpha_0 \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_i + \mathbf{A})^{-1}, \tag{12}$$

where $\mathbf{A} = \text{diag}(\alpha_1, \alpha_2, \ldots, \alpha_M)$, each diagonal element of which is from the hyperparameters $\boldsymbol{\alpha}$.

Before proceeding, we note the characteristics of the aforementioned algorithm. Using a ML (empirical Bayesian) procedure, one constitutes point estimates for the hyperparameters $\boldsymbol{\alpha}$ and α_0. Importantly, as implemented in (9), the hyperparameter point estimates are based upon *all* of the observed CS measurements $\{\boldsymbol{v}_i\}_{i=1,M}$, emphasizing the multi-task nature of the analysis. Subsequently, using the point estimate constituted using all of the data, a full posterior estimate is constituted for the basis-

[4]This can be demonstrated by considering a log transformation to random variables $\boldsymbol{\alpha}$ and α_0 that are Gamma distributed, and maximizing the transformed version of (8), as shown in the appendix of [19]. Thus, the limit holds in the logarithmic scale.

function coefficients $\{\boldsymbol{\theta}_i\}_{i=1,M}$, where for this latter calculation $\boldsymbol{\theta}_i$ is only dependent on $\boldsymbol{v}_i$. Thus, to estimate the hyperparameters all of the data are used, while to update an approximation to the wavelet coefficients $\boldsymbol{\theta}_i$ only the associated task-dependent CS measurements are employed (see Fig. 1). This suggests an iterative algorithm that alternates between these global and local solutions, as outlined next.

B. Empirical Bayesian estimate for $\boldsymbol{\alpha}$ and α_0

The empirical Bayesian estimate for $\boldsymbol{\alpha}$ and α_0 via (9) is determined by maximizing the marginal likelihood, or equivalently, its logarithm:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha},\alpha_0) &= \sum_{i=1}^{L}\log p(\boldsymbol{v}_i|\,\boldsymbol{\alpha},\alpha_0)=\sum_{i=1}^{L}\log\int p(\boldsymbol{v}_i|\,\boldsymbol{\theta}_i,\alpha_0)p(\boldsymbol{\theta}_i|\,\boldsymbol{\alpha})d\boldsymbol{\theta}_i \\ &= -\frac{1}{2}\sum_{i=1}^{L}\left[N_i\log 2\pi+\log|\mathbf{C}_i|+\boldsymbol{v}_i^T\mathbf{C}_i^{-1}\boldsymbol{v}_i\right], \qquad (13)\end{aligned}$$

with

$$\mathbf{C}_i=\alpha_0^{-1}\mathbf{I}+\boldsymbol{\Phi}_i\mathbf{A}^{-1}\boldsymbol{\Phi}_i^T. \qquad (14)$$

There are (at least) two approaches that can be applied to maximize $\mathcal{L}(\boldsymbol{\alpha},\alpha_0)$ with respect to $\boldsymbol{\alpha}$ and α_0.

1) Iterative Solution: Differentiating (13) with respect to $\boldsymbol{\alpha}$ and α_0, setting the result to zero and rearranging, following the approach of MacKay [41], yields

$$\alpha_j^{new} = \frac{L-\alpha_j\sum_{i=1}^{L}\Sigma_{i,(j,j)}}{\sum_{i=1}^{L}\mu_{i,j}^2}, \qquad j\in\{1,2,\ldots,M\}, \qquad (15)$$

$$\alpha_0^{new} = \frac{\sum_{i=1}^{L}(N_i-M+\sum_{j=1}^{M}\alpha_j\Sigma_{i,(j,j)})}{\sum_{i=1}^{L}\|\boldsymbol{v}_i-\boldsymbol{\Phi}_i\boldsymbol{\mu}_i\|_2^2}, \qquad (16)$$

where $\mu_{i,j}$ is the jth component of $\boldsymbol{\mu}_i$, and $\Sigma_{i,(j,j)}$ is the jth diagonal element of $\boldsymbol{\Sigma_i}$. Note that $\boldsymbol{\alpha}^{new}$ and α_0^{new} are a function of $\{\boldsymbol{\mu}_i\}_{i=1,L}$ and $\{\boldsymbol{\Sigma}_i\}_{i=1,L}$, while $\{\boldsymbol{\mu}_i\}_{i=1,L}$ and $\{\boldsymbol{\Sigma}_i\}_{i=1,L}$ are a function of $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}_0$. This suggests an iterative algorithm, which iterates between (11)-(12) and (15)-(16), until a convergence criterion has been satisfied. In this process, it is observed that many of the α_j tend to infinity (or numerically indistinguishable from infinity given the machine precision) for those $\{\theta_{i,j}\}_{i=1}^{L}$ that have insignificant amplitudes for representation of $\{\boldsymbol{v}_i\}_{i=1}^{L}$; only a relatively small set of $\{\theta_{i,j}\}_{i=1}^{L}$, for which the corresponding α_j remains relatively small, contribute for the representation, and the level of *joint* sparseness is determined automatically.

While the iterative algorithm described above has been demonstrated to yield a highly accurate sparse linear-regression representation (e.g., see [19] for $L=1$), the following practical limitation is observed

when applied to large-scale problems. When evaluating (12) one must invert matrices of size $M \times M$, an $\mathcal{O}(M^3)$ operation[5], thereby making this approach relatively slow for data $\{\boldsymbol{v}_i\}_{i=1,L}$ of large dimension N_i (at least for the first few iterations). This motivates development of the following fast algorithm.

2) Fast Algorithm: Similar to [43], considering the dependence of $\mathcal{L}(\boldsymbol{\alpha}, \alpha_0)$ on a single hyperparameter α_j, $j \in \{1, 2, \ldots, M\}$, we can decompose $\mathbf{C}_i$ in (14) as

$$\begin{aligned} \mathbf{C}_i &= \alpha_0^{-1}\mathbf{I} + \sum_{k \neq j} \alpha_k^{-1} \boldsymbol{\Phi}_{i,k} \boldsymbol{\Phi}_{i,k}^T + \alpha_j^{-1} \boldsymbol{\Phi}_{i,j} \boldsymbol{\Phi}_{i,j}^T \\ &= \mathbf{C}_{i,-j} + \alpha_j^{-1} \boldsymbol{\Phi}_{i,j} \boldsymbol{\Phi}_{i,j}^T, \end{aligned} \tag{17}$$

where $\boldsymbol{\Phi}_i = [\boldsymbol{\Phi}_{i,1}, \boldsymbol{\Phi}_{i,2}, \ldots, \boldsymbol{\Phi}_{i,M}]$, and $\mathbf{C}_{i,-j}$ is $\mathbf{C}_i$ with the contribution of basis function $\boldsymbol{\Phi}_{i,j}$ removed. Applying the matrix determinant and inversion lemmas, we can write the terms of interest in $\mathcal{L}(\boldsymbol{\alpha}, \alpha_0)$ as

$$|\mathbf{C}_i| = |\mathbf{C}_{i,-j}| \, |1 + \alpha_j^{-1} \boldsymbol{\Phi}_{i,j}^T \mathbf{C}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j}|, \tag{18}$$

$$\mathbf{C}_i^{-1} = \mathbf{C}_{i,-j}^{-1} - \frac{\mathbf{C}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j} \boldsymbol{\Phi}_{i,j}^T \mathbf{C}_{i,-j}^{-1}}{\alpha_j + \boldsymbol{\Phi}_{i,j}^T \mathbf{C}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j}}. \tag{19}$$

From this, we can write

$$\begin{aligned} \mathcal{L}(\boldsymbol{\alpha}, \alpha_0) &= -\frac{1}{2} \sum_{i=1}^{L} \left[N_i \log 2\pi + \log |\mathbf{C}_{i,-j}| + \boldsymbol{v}_i^T \mathbf{C}_{i,-j}^{-1} \boldsymbol{v}_i - \log \left(\frac{\alpha_j}{\alpha_j + s_{i,j}} \right) - \frac{q_{i,j}^2}{\alpha_j + s_{i,j}} \right] \\ &= \mathcal{L}(\boldsymbol{\alpha}_{-j}, \alpha_0) + \frac{1}{2} \sum_{i=1}^{L} \left[\log \left(\frac{\alpha_j}{\alpha_j + s_{i,j}} \right) + \frac{q_{i,j}^2}{\alpha_j + s_{i,j}} \right] \\ &= \mathcal{L}(\boldsymbol{\alpha}_{-j}, \alpha_0) + \ell(\alpha_j, \alpha_0), \end{aligned} \tag{20}$$

where $\boldsymbol{\alpha}_{-j}$ is the same as $\boldsymbol{\alpha}$ except the jth component is removed, and we have defined

$$s_{i,j} \triangleq \boldsymbol{\Phi}_{i,j}^T \mathbf{C}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j}, \quad \text{and} \quad q_{i,j} \triangleq \boldsymbol{\Phi}_{i,j}^T \mathbf{C}_{i,-j}^{-1} \boldsymbol{v}_i. \tag{21}$$

Differentiating $\ell(\alpha_j, \alpha_0)$ with respect to α_j and setting the result to zero, followed by algebra, yields

$$\frac{\partial \mathcal{L}(\boldsymbol{\alpha}, \alpha_0)}{\partial \alpha_j} = \sum_{i=1}^{L} \frac{s_{i,j}^2 / \alpha_j + s_{i,j} - q_{i,j}^2}{2(\alpha_j + s_{i,j})^2} = 0. \tag{22}$$

[5]A simple modification to (12) is available from [42] by exploiting the matrix inversion lemma, which leads to an $\mathcal{O}(N_i^3)$ operation per iteration. Nonetheless, the iterative implementation still does not scale well.

Except for the trivial solution $\alpha_j = \infty$, the other solutions of (22) are infeasible to be expressed analytically as this requires finding the zeros of a polynomial of degree $2L-1$. To avoid the complexity of the zero-finding of polynomials, we thus assume that $\alpha_j \ll s_{i,j}$ [6] and the denominator of (22) is now relatively invariant with respect to α_j. Therefore, we may approximate another solution as

$$\alpha_j \approx \frac{L}{\sum_{i=1}^{L}(q_{i,j}^2 - s_{i,j})/s_{i,j}^2}, \qquad \text{if } \sum_{i=1}^{L}(q_{i,j}^2 - s_{i,j})/s_{i,j}^2 > 0, \tag{23}$$

$$\alpha_j = \infty, \qquad \text{otherwise.} \tag{24}$$

The analysis in Appendix A shows that the finite approximate solution (23) is at the vicinity of a stationary point of $\ell(\alpha_j, \alpha_0)$, where $\ell(\alpha_j, \alpha_0)$ has a maximum (which may be not unique). Due to this approximation, we no longer perform an exact maximum likelihood estimation of α_j, but only (monotonically) increase $\mathcal{L}(\boldsymbol{\alpha}, \alpha_0)$ at each iteration, and thus more iterations are incurred upon convergence. But we find this approximation is extremely effective as it allows much faster computation of α_j than exactly solving (22). In addition, (23) reduces to the exact formula of α_j when $L = 1$, which corresponds to single-task learning as considered in [43].

The remaining formulas are similar to those considered for the fast algorithm of the RVM, and therefore one may refer to [43] for more details. We here only summarize some of its key properties. Compared with the iterative algorithm presented above, the fast algorithm operates in a constructive manner, i.e., sequentially adds (or deletes) candidate basis function to the model until all m "relevant" basis functions[7] (for which the associated weights are nonzero) have been included. Thus, the complexity of the algorithm is more related to m than M. Further, by using the matrix inversion lemma, the inverse operation in (12) has been implemented by iterative update formulae with reduced complexity (see the appendix of [43]). Detailed empirical analysis of this fast algorithm shows that it has complexity $\mathcal{O}(LMm^2)$, which is more efficient than the iterative solution, especially when the underlying signals are truly jointly sparse ($m \ll M$).

[6] This has generally been found to be valid numerically, e.g., typically $s_{i,j} > 20\alpha_j$. The consequence of this approximation is a fast solving of (22), but with slightly more iterations upon convergence. See the text that follows for explanations.

[7] Here, the set of m "relevant" basis functions is a union of all the "relevant" basis functions selected by the algorithm for all the L tasks. One basis function will be included if it is relevant to *at least* one task. Because of the (assumed) statistical inter-relationship among the tasks, most of the "relevant" basis functions selected for each task are expected to be overlapped, and thus ideally $m \ll M$.

III. Integrating Out Regression Noise Variance

To apply the fast algorithm discussed above, an initial guess of α_0 is required, and this value is then fixed thereafter to allow the iterative update formulae [43]. The α_0 is a nuisance parameter, which in effect may have an identifiability issue [17], and an inappropriate value of it may contaminate algorithm performance significantly. In this section we introduce a modified sparse-regression model for multi-task CS inversion. The algorithm integrates α_0 out, rather than seeking a point estimate of α_0, and the computation is solely concentrated on recovering the hyperparameters $\boldsymbol{\alpha}$. This allows a fast sequential optimization method, which is similar to the fast algorithm in Sec. II-B2, but without the constraint of having a fixed point estimate for α_0. As to be demonstrated in the next section, the modified fast algorithm has improved robustness to the parameter setting over the original RVM and the fast algorithm in Sec. II-B2.

A. Modified Sparse Linear-Regression Model

Similar to the original RVM formulation [19] and the formulation in Sec. II, we define a zero-mean Gaussian prior for each component of $\boldsymbol{\theta}_i$, and define a Gamma prior on the noise precision α_0:

$$p(\boldsymbol{\theta}_i|\,\boldsymbol{\alpha},\alpha_0) = \prod_{j=1}^{M}\mathcal{N}(\theta_{i,j}\mid 0,\alpha_0^{-1}\alpha_j^{-1}), \tag{25}$$

$$p(\alpha_0|\,a,b) = \mathrm{Ga}(\alpha_0|\,a,b). \tag{26}$$

Note that the only difference between the formulation specified above and that in the original RVM is α_0 is included in the prior of $\boldsymbol{\theta}_i$ [44]. Mathematically, this modification allows the integration involved in the sequel to be performed analytically. As we can see, given $\boldsymbol{\alpha}$ and CS measurements $\boldsymbol{v}_i$, the likelihood function of $\boldsymbol{\theta}_i$ may be expressed as

$$\begin{aligned} p(\boldsymbol{\theta}_i|\,\boldsymbol{v}_i,\boldsymbol{\alpha}) &= \int p(\boldsymbol{\theta}_i|\,\boldsymbol{v}_i,\boldsymbol{\alpha},\alpha_0)p(\alpha_0|\,a,b)d\alpha_0 \\ &= \frac{\Gamma(a+M/2)\left[1+\frac{1}{2b}(\boldsymbol{\theta}_i-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{\theta}_i-\boldsymbol{\mu}_i)\right]^{-(a+M/2)}}{\Gamma(a)(2\pi b)^{M/2}|\boldsymbol{\Sigma}_i|^{1/2}}, \end{aligned} \tag{27}$$

where

$$\boldsymbol{\mu}_i = \boldsymbol{\Sigma}_i\boldsymbol{\Phi}_i^T\boldsymbol{v}_i, \tag{28}$$

$$\boldsymbol{\Sigma}_i = (\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_i+\mathbf{A})^{-1}, \tag{29}$$

with $\mathbf{A} = \text{diag}(\alpha_1, \alpha_2, \ldots, \alpha_M)$. By integrating α_0 out, we notice that the likelihood function has been changed from a multivariate Gaussian distribution (10) to a multivariate Student-t distribution (27). Therefore, theoretically, this modified formulation has the advantage of inducing a heavy-tailed distribution on the basis coefficients, as is apparent in (27), and allows for more robust shrinkage and borrowing of information, as some tasks can be outliers [45], [46].

As shown in expression (2), the additive noise is modeled as i.i.d. draws from a zero-mean Gaussian random variable with precision α_0. In placing a Gamma prior on the residual precision α_0, and then marginalized out the precision α_0, we are not changing the additive noise structure of expression (2). Instead, we are inducing a heavier-tailed Student-t distribution on the residual noise, which is more robust in allowing outlying measurements. Hence, instead of introducing an assumption, we are relaxing the assumption of Gaussian noise to obtain a more robust approach. The approach of placing a hyperprior on parameters to induce heavier-tailed distributions for greater robustness is common in Bayesian statistics. The approach of also incorporating the residual precision, α_0, in the prior for the basis coefficients (as in expression (25)) is a common specification [44]. Because we still have a distinct α_j for every element of the coefficient vector, this is not restrictive but is strictly incorporated for tractability and to induce a heavier-tailed prior for the basis coefficients.

B. Empirical Bayesian Estimate for $\boldsymbol{\alpha}$

Similar to the algorithms in Sec. II, an empirical Bayesian approach can be applied to estimate hyperparameters $\boldsymbol{\alpha}$, i.e., seeking $\boldsymbol{\alpha}$ to maximize the marginal likelihood, or equivalently, its logarithm:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}) &= \sum_{i=1}^{L} \log p(\boldsymbol{v}_i|\, \boldsymbol{\alpha}) = \sum_{i=1}^{L} \log \int p(\boldsymbol{v}_i|\, \boldsymbol{\theta}_i, \alpha_0) p(\boldsymbol{\theta}_i|\, \boldsymbol{\alpha}, \alpha_0) p(\alpha_0|\, a, b) d\boldsymbol{\theta}_i \, d\alpha_0 \\ &= -\frac{1}{2} \sum_{i=1}^{L} \left[(N_i + 2a) \log \left(\boldsymbol{v}_i^T \mathbf{B}_i^{-1} \boldsymbol{v}_i + 2b \right) + \log |\mathbf{B}_i| \right] + \text{const}, \end{aligned} \tag{30}$$

with

$$\mathbf{B}_i = \mathbf{I} + \boldsymbol{\Phi}_i \mathbf{A}^{-1} \boldsymbol{\Phi}_i^T. \tag{31}$$

1) Iterative Solution: Both direct differentiation and the EM algorithm can be applied to maximize (30) for a point estimate of $\boldsymbol{\alpha}$, yielding

$$\alpha_j = \frac{L}{\sum_{i=1}^{L} \mu_{i,j}^2 (N_i + 2a) / (\boldsymbol{v}_i^T \mathbf{B}_i^{-1} \boldsymbol{v}_i + 2b) + \Sigma_{i,(j,j)}}, \qquad j \in \{1, 2, \ldots, M\}. \tag{32}$$

This suggests an iterative algorithm that iterates between (32) and (28)–(29) until convergence is achieved. Since the computation of (29) involves the matrix inversion of size $M \times M$, an $\mathcal{O}(N_i^3)$ operation, this undermines the applications of this approach for data $\{\boldsymbol{v}_i\}_{i=1,L}$ of large dimension N_i.

2) Fast Algorithm: A fast algorithm can be derived in a manner parallel to that of the fast RVM algorithm [43]. Considering the dependence of $\mathcal{L}(\boldsymbol{\alpha})$ on a single hyperparameter α_j, $j \in \{1, 2, \ldots, M\}$, we may decompose $\mathbf{B}_i$ in (31) as

$$\begin{aligned}\mathbf{B}_i &= \mathbf{I} + \boldsymbol{\Phi}_i \mathbf{A}_i^{-1} \boldsymbol{\Phi}_i^T = \mathbf{I} + \sum_{k \neq j} \alpha_k^{-1} \boldsymbol{\Phi}_{i,k} \boldsymbol{\Phi}_{i,k}^T + \alpha_j^{-1} \boldsymbol{\Phi}_{i,j} \boldsymbol{\Phi}_{i,j}^T \\ &= \mathbf{B}_{i,-j} + \alpha_j^{-1} \boldsymbol{\Phi}_{i,j} \boldsymbol{\Phi}_{i,j}^T, \end{aligned} \tag{33}$$

where $\mathbf{B}_{i,-j}$ is $\mathbf{B}_i$ with the contribution of basis function $\boldsymbol{\Phi}_{i,j}$ removed. The matrix determinant and inversion lemmas may be used to express

$$|\mathbf{B}_i| = |\mathbf{B}_{i,-j}| \, |1 + \alpha_j^{-1} \boldsymbol{\Phi}_{i,j}^T \mathbf{B}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j}|, \tag{34}$$

$$\mathbf{B}_i^{-1} = \mathbf{B}_{i,-j}^{-1} - \frac{\mathbf{B}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j} \boldsymbol{\Phi}_{i,j}^T \mathbf{B}_{i,-j}^{-1}}{\alpha_j + \boldsymbol{\Phi}_{i,j}^T \mathbf{B}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j}}. \tag{35}$$

From this, we may write

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}) &= -\frac{1}{2} \sum_{i=1}^{L} \left[(N_i + 2a) \log \left(\frac{1}{2} \boldsymbol{v}_i^T \mathbf{B}_{i,-j}^{-1} \boldsymbol{v}_i + b \right) + \log |\mathbf{B}_{i,-j}| \right] + \text{const} \\ &\quad -\frac{1}{2} \sum_{i=1}^{L} \left[\log(1 + \alpha_j^{-1} s_{i,j}) + (N_i + 2a) \log \left(1 - \frac{q_{i,j}^2 / g_{i,j}}{\alpha_j + s_{i,j}} \right) \right] \\ &= \mathcal{L}(\boldsymbol{\alpha}_{-j}) + \ell(\alpha_j), \end{aligned} \tag{36}$$

where $\boldsymbol{\alpha}_{-j}$ is the same as $\boldsymbol{\alpha}$ except the jth component is removed, and we have defined

$$s_{i,j} \triangleq \boldsymbol{\Phi}_{i,j}^T \mathbf{B}_{i,-j}^{-1} \boldsymbol{\Phi}_{i,j}, \quad q_{i,j} \triangleq \boldsymbol{\Phi}_{i,j}^T \mathbf{B}_{i,-j}^{-1} \boldsymbol{v}_i, \quad \text{and} \quad g_{i,j} \triangleq \boldsymbol{v}_i^T \mathbf{B}_{i,-j}^{-1} \boldsymbol{v}_i + 2b. \tag{37}$$

Differentiating $\mathcal{L}(\boldsymbol{\alpha})$ with respect to α_j and setting the result to zero, followed by algebra, yields:

$$\frac{\partial \mathcal{L}(\boldsymbol{\alpha})}{\partial \alpha_j} = \sum_{i=1}^{L} \frac{s_{i,j}(s_{i,j} - q_{i,j}^2/g_{i,j})/\alpha_j - (N_i + 2a) q_{i,j}^2/g_{i,j} + s_{i,j}}{2(\alpha_j + s_{i,j})(\alpha_j + s_{i,j} - q_{i,j}^2/g_{i,j})} = 0. \tag{38}$$

Similar to the fast algorithm in Sec. II-B2, except for the trivial solution $\alpha_j = \infty$, the other solutions of (38) are infeasible to be expressed analytically. We thus assume that $\alpha_j \ll s_{i,j}$ (again, this generally holds numerically, e.g., typically $s_{i,j} > 20\alpha_j$) and the denominator of (38) is now relative invariant with

respect to α_j. Therefore, we may approximate another solution as

$$\alpha_j \approx \frac{L}{\sum_{i=1}^{L} \frac{(N_i+2a)q_{i,j}^2/g_{i,j}-s_{i,j}}{s_{i,j}(s_{i,j}-q_{i,j}^2/g_{i,j})}}, \qquad \text{if } \sum_{i=1}^{L} \frac{(N_i+2a)q_{i,j}^2/g_{i,j}-s_{i,j}}{s_{i,j}(s_{i,j}-q_{i,j}^2/g_{i,j})} > 0, \tag{39}$$

$$\alpha_j = \infty, \qquad \text{otherwise.} \tag{40}$$

Again, the analysis in Appendix A shows that the finite approximate solution (39) is at the vicinity of a stationary point of $\ell(\alpha_j)$, where $\ell(\alpha_j)$ has a maximum (which may be not unique). Due to this approximation, we no longer perform an exact maximum likelihood estimate of α_j, but only (monotonically) increase $\mathcal{L}(\boldsymbol{\alpha})$ at each iteration. We find this approximation is extremely effective as it allows much faster computation of α_j than exactly solving (38). In addition, (39) reduces to the exact formula of α_j when $L = 1$, which corresponds to single-task CS.

Recall that setting $\alpha_j = \infty$ is equivalent to setting $\theta_{i,j} = 0$, and hence removing $\boldsymbol{\Phi}_{i,j}$ from the representation; hence, (39)–(40) controls the addition and deletion of particular $\boldsymbol{\Phi}_{i,j}$ from the signal representation. If we perform these operations sequentially for varying j, we realize an efficient learning algorithm.

In practice, it is relatively straightforward to compute $s_{i,j}$ and $q_{i,j}$ for all the basis vector $\boldsymbol{\Phi}_{i,j}$, including those not currently utilized in the model (i.e., for which $\alpha_j = \infty$). These quantities can be computed by maintaining and updating values of

$$S_{i,j} = \boldsymbol{\Phi}_{i,j}^T \mathbf{B}_i^{-1} \boldsymbol{\Phi}_{i,j}, \quad Q_{i,j} = \boldsymbol{\Phi}_{i,j}^T \mathbf{B}_i^{-1} \boldsymbol{v}_i, \quad \text{and} \quad G_i = \boldsymbol{v}_i^T \mathbf{B}_i^{-1} \boldsymbol{v}_i + 2b, \tag{41}$$

and from these it follows simply:

$$s_{i,j} = \frac{\alpha_j S_{i,j}}{\alpha_j - S_{i,j}}, \quad q_{i,j} = \frac{\alpha_j Q_{i,j}}{\alpha_j - S_{i,j}}, \quad \text{and} \quad g_{i,j} = G_i + \frac{Q_{i,j}^2}{\alpha_j - S_{i,j}}. \tag{42}$$

Further, it is convenient to utilize the matrix inversion lemma to obtain the quantities of interest:

$$S_{i,j} = \boldsymbol{\Phi}_{i,j}^T \boldsymbol{\Phi}_{i,j} - \boldsymbol{\Phi}_{i,j}^T \boldsymbol{\Phi}_i \boldsymbol{\Sigma}_i \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_{i,j}, \tag{43}$$

$$Q_{i,j} = \boldsymbol{\Phi}_{i,j}^T \boldsymbol{v}_i - \boldsymbol{\Phi}_{i,j}^T \boldsymbol{\Phi}_i \boldsymbol{\Sigma}_i \boldsymbol{\Phi}_i^T \boldsymbol{v}_i, \tag{44}$$

$$G_i = \boldsymbol{v}_i^T \boldsymbol{v}_i - \boldsymbol{v}_i^T \boldsymbol{\Phi}_i \boldsymbol{\Sigma}_i \boldsymbol{\Phi}_i^T \boldsymbol{v}_i + 2b. \tag{45}$$

Here quantities $\boldsymbol{\Phi}_i$ and $\boldsymbol{\Sigma}_i$ contain only those basis vectors that are currently included in the model, and the computation thus scales as the cube of that measure, which is typically only a very small fraction of M. Furthermore, these quantities can also be calculated via the update formulae, as shown in the Appendix

B, with reduced computation. Similar update formulae are applied to the original fast RVM algorithm [43] when α_0 is fixed. However, our modified fast algorithm is applicable without this constraint.

IV. Example Results

We denote the fast algorithm in Sec. II-B2 as BCS, and the fast algorithm in Sec. III-B2 as BCS*, and test the performance of BCS and BCS* on both single-task (ST) and multi-task (MT) CS inverse problems. To be concise, in the example CS-reconstruction figures that follow we only present the BCS* results, and the full quantitative performance comparison between BCS and BCS* is summarized in tables. For a fair comparison between BCS and BCS*, we initialize $\alpha_0 = 10^2/\mathrm{std}(\boldsymbol{v})^2$ for BCS and fix this value thereafter (for the fast algorithm); with regard to BCS*, we set $a = 10^2/\mathrm{std}(\boldsymbol{v})^2$ and $b = 1$ such that the mean of the Gamma prior $p(\alpha_0|\, a, b)$[8] is aligned with the fixed value of α_0 in BCS. As a comparison, we also provide the performances of Orthogonal Matching Pursuit (OMP) [8] for ST learning and Simultaneous Orthogonal Matching Pursuit (S-OMP) [47] for MT learning. In the experiments we evaluate the reconstruction error as $\|\boldsymbol{u} - \hat{\boldsymbol{u}}_{\mathrm{method}}\|_2/\|\boldsymbol{u}\|_2$. All the computations presented here were performed on a 3.4GHz Pentium machine. The Matlab code is available online at `http://www.ece.duke.edu/~shji/BCS.html`.

A. 1D Signals

In the first example we consider $L = 2$ signals of length $M = 512$, each containing 20 spikes created by choosing 20 locations at random and then putting ± 1 at these points (Figs. 2(a-b)). The two original signals are created such that they have 75% spikes at the same positions, but all have random ± 1 amplitudes. The projection matrix $\boldsymbol{\Phi}_i$ is constructed by first creating a $N_i \times M$ matrix with i.i.d. draws of a Gaussian distribution $\mathcal{N}(0, 1)$, and then the rows of $\boldsymbol{\Phi}_i$ are normalized to unit norm. Zero-mean Gaussian noise with standard deviation $\sigma_0 = 0.005$ is added to each of the N_i measurements that define the data $\boldsymbol{v}_i$. In the experiment $N_1 = 90$, $N_2 = 70$ and the reconstructions are implemented by ST-CS and MT-CS, respectively.

Figures 2(c-d) demonstrate the reconstruction results with BCS* for single-task inference. Because of insufficient number of measurements (N_i is smaller than a minimum quantity required for faithful reconstruction [5], [6]), the reconstructed signals are highly noisy. However, since two original signals are not statistically independent, multi-task CS is able to take advantage of the inter-relationships and

[8]The variance of the Gamma prior is a/b^2.

yields almost perfect reconstructions (Figs. 2(e-f)). The results of BCS are very similar to BCS*, and therefore are omitted here.

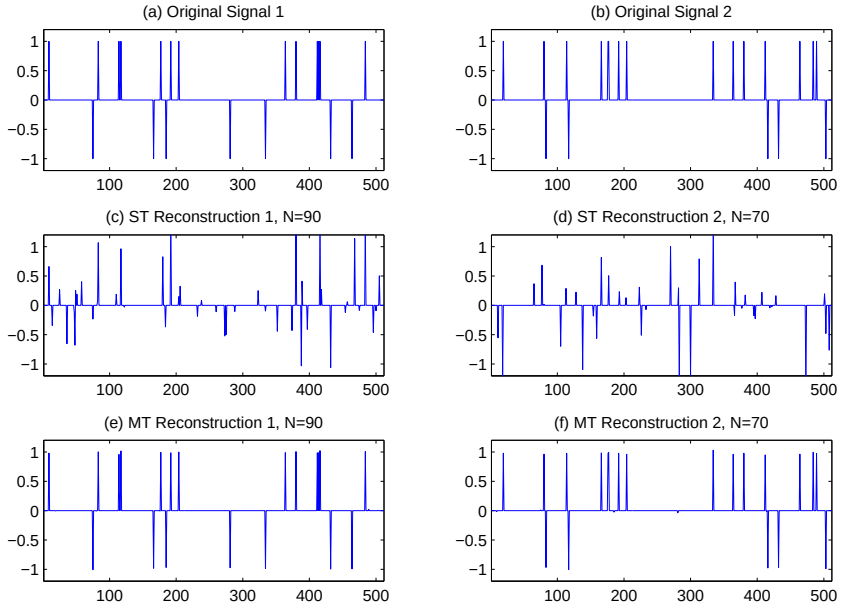


Fig. 2. Reconstruction of the *Spikes* of length $M = 512$. The two original signals have 75% spikes at the same positions, but all have random ± 1 amplitudes. (a-b) Original signals; (c-d) reconstructed signals by ST-BCS*; (e-f) reconstructed signals by MT-BCS*.

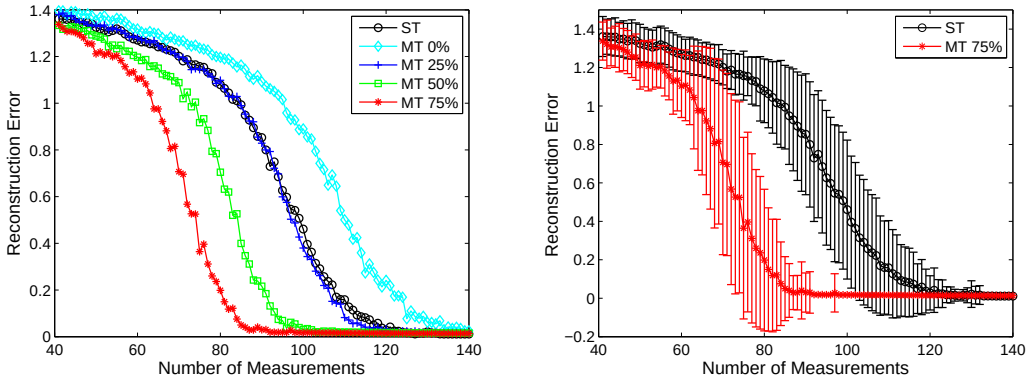


Fig. 3. Reconstruction errors of ST-BCS* and MT-BCS* as a function of increasing N. The two original signals have 75%, 50%, 25%, 0% spikes at the same positions, with random ± 1 amplitudes. The results are averaged over 100 runs. (a) The average reconstruction errors for 75%, 50%, 25% and 0% similarity; (b) the variance of reconstruction errors for 75% similarity.

To study how the similarity between the original signals affects the reconstruction performance of MT-CS, in the second experiment we use the same dataset as in Fig. 2 and study the performances of BCS* for different similarity levels, e.g., 75%, 50%, 25% or 0% spikes are at the same locations. For

each similarity level, starting at the 41st measurement, after each random measurement is conducted, the associated reconstruction errors are computed for ST-BCS* and MT-BCS*, until a total 140 measurements are conducted for each of the two original signals. Because of the randomness in the experiment (e.g., the random CS measurements, the random locations and ± 1 amplitudes of the spikes, and the random additive noise), we execute the experiment 100 times with the average performance reported in Fig. 3.

It is demonstrated in Fig. 3 that the reconstruction error of the MT-BCS* is much smaller than that of the ST-BCS*, when the similarities are at 75% and 50%. However, when the similarity is 25%, the improvements are minor or none; when the similarity is 0% (i.e., two signals are totally independent), the MT performances are even worse than ST. This is consistent with our intuition that for multi-task CS to be superior, the original signals should have at least some level of similarity, otherwise transferring information among totally independent tasks will deteriorate rather than help to improve the performance. Therefore, the multi-task CS framework developed here is particularly relevant for problems in which the images under test have a relatively high degree of similarity, e.g., when performing CS inversion of multiple medical images of the same body part, with the multiple CS measurements taken from the same or different individuals; or for video data, where consecutive images are expected to have a high degree of statistical similarity. The example results of this kind are provided next.

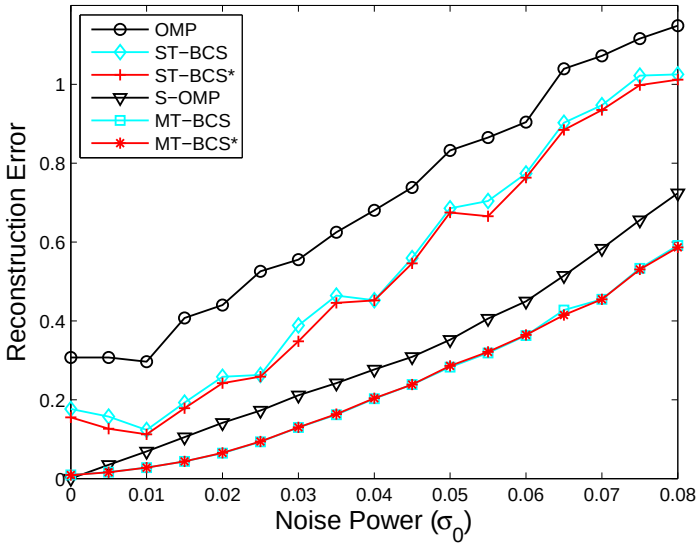


Fig. 4. Reconstruction errors of OMP, BCS, and BCS* for single-task (ST) learning and multi-task (MT) learning as a function of increasing noise power σ_0. The two original signals have 75% spikes at the same positions, with random ± 1 amplitudes; we made $N = 110$ random measurements for each original signal. The results are averaged over 100 runs.

To study how the additive noise affects the reconstruction performance of MT-CS, in the third ex-

periment we use the same dataset as in Fig. 2 and study the performance of BCS* at an increasing level of noise power σ_0. As a comparison, we also provide the performances of OMP and BCS for ST-CS and MT-CS. In the experiment, the two original signals have 75% spikes at the same positions, with random ± 1 amplitudes; we made $N = 110$ random measurements for each original signal; the algorithms then exploited these two sets of measurements independently or jointly for reconstruction. Again, due to the randomness in the experiment, we execute the experiment 100 times with the average performance reported in Fig. 4.

It is demonstrated in Fig. 4 that as the noise power σ_0 increases, the reconstruction errors generally increase for all the algorithms considered, while typically the MT-CS algorithms outperform the ST-CS algorithms. Regarding the performance comparison between BCS, BCS* and OMP, the superiority of BCS and BCS* is demonstrated by their lower reconstruction errors than that of OMP, consistently over a range of noise powers. Comparing the reconstruction errors of BCS and BCS*, the benefit of BCS* over BCS is more pronounced in single-task learning than in multi-task learning. This is likely because in multi-task learning one utilizes more data, and therefore the differences manifested by an improved algorithm are less apparent. The results indicate that, in general, the marginalization of α_0, as in BCS*, may be a preferred approach rather than estimating α_0 as in BCS, particularly when the available data are not abundant.

B. 2D Images

In the following set of experiments, the performance of MT-CS is compared to ST-CS on three example problems that involve 2D images. All the projection matrices $\mathbf{\Phi}$ considered here are drawn from a uniform spherical distribution [48].

1) Random-Bars: Figure 5 shows the reconstruction results for *Random-Bars*, where Fig. 5(a) is from [48] and the other two images (b-c) are modified from (a) to represent similar tasks for simultaneous CS inversion, e.g., the intensities of all the rectangles in (b-c) are randomly permuted from (a), and the positions of all the rectangles are shifted by distances randomly sampled from a uniform distribution in $[-10, 10]$. All three original images have the size 1024×1024. We used the Haar wavelet expansion, which is well suited to images of this type, with a coarsest scale $j_0 = 3$, and a finest scale $j_1 = 6$. Figures 5(a-c) shows the result of linear reconstruction (i.e., the inverse wavelet transform) with $N = 4096$ samples, which represents the best performance that could be achieved by all the CS implementations considered here. Figures 5(d-f) have results of ST-BCS* by using the hybrid CS scheme (i.e., the CS measurements are made only on the fine-scale coefficients; no compression on the coarsest-scale coefficients) [48] with

$N = 670$ compressed samples for each task, whereas Figs. 5(g-i) have the results of MT-BCS*. The performance comparison between BCS and BCS* is summarized in Table I.

It is demonstrated that MT-CS yields a better reconstruction performance than that of ST-CS, both for BCS and BCS*; comparing the reconstruction errors of BCS and BCS* for the case of single task learning, ST-BCS* is markedly better than ST-BCS, whereas for multi-task learning MT-BCS* is only slightly better than MT-BCS. This is consistent to the observations in Fig. 4, where the benefit of BCS* over BCS is more apparent in ST-CS than in MT-CS.

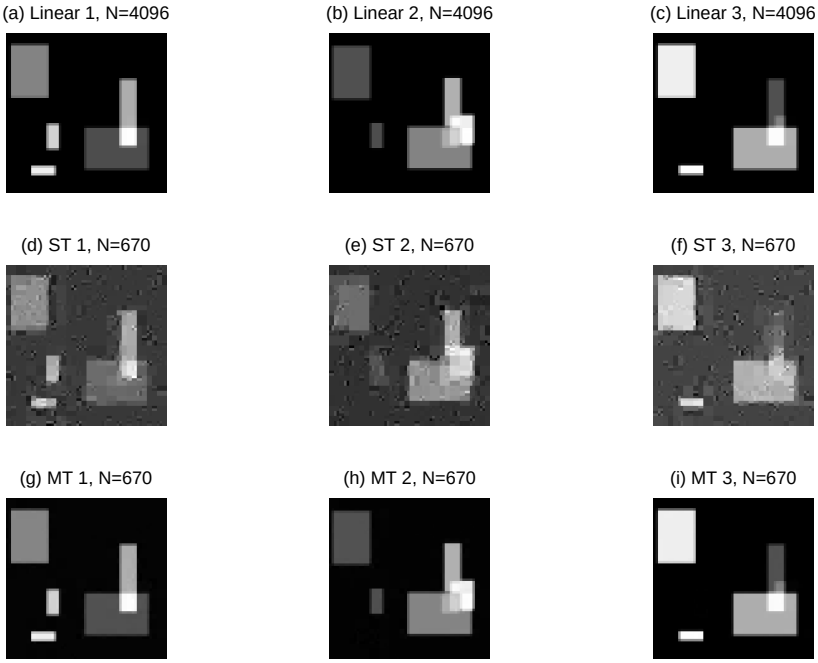


Fig. 5. Reconstruction of *Random-Bars* with hybrid CS. (a-c) Linear reconstructions of three original images. Example (a) is from [48], and (b-c) are the modified images from (a) by us to represent similar tasks for simultaneous CS inversion. The intensities of all the rectangles in (b-c) are randomly permuted from (a), and the positions of all the rectangles are shifted by distances randomly sampled from a uniform distribution in $[-10, 10]$. (d-f) reconstructed images by ST-BCS*; (g-i) reconstructed images by MT-BCS*.

2) MRI Images: Figure 6 shows the reconstruction results for *MRI Images*, which includes five image slices of a human head. All five original images have the size 128×128. We used a hybrid CS scheme [48] for image reconstruction, with a coarsest scale $j_0 = 3$, and a finest scale $j_1 = 6$ on the "Daubechies 8" wavelet. Figure 6(a-e) show the results of linear reconstruction with $N = 4096$ samples, which represents the best performance that could be achieved by all the CS implementations considered here. Figures 6(f-j) have results for the ST-BCS* with $N = 1636$ compressed samples for each task, whereas Figs. 6(k-o) have the results for MT-BCS*. The full performance comparison between BCS and BCS*

TABLE I
RECONSTRUCTION PERFORMANCES OF LINEAR, ST-CS AND MT-CS ON *Random-Bars*.

	Recon. Error			Run Time (secs)		
	Task 1	Task 2	Task 3	Task 1	Task 2	Task 3
ST-BCS	0.4439	0.3619	0.3674	63.91	40.72	24.53
MT-BCS	0.2319	0.2281	0.1977	67.76 per task		
ST-BCS*	0.3678	0.3502	0.3038	57.91	39.99	33.33
MT-BCS*	0.2277	0.2181	0.1936	39.74 per task		
Linear	0.2271	0.2178	0.1936	0		

are summarized in Table II. The relative performance of ST-BCS to MT-BCS, and between BCS and BCS*, are consistent with the above *Random-Bars* results.

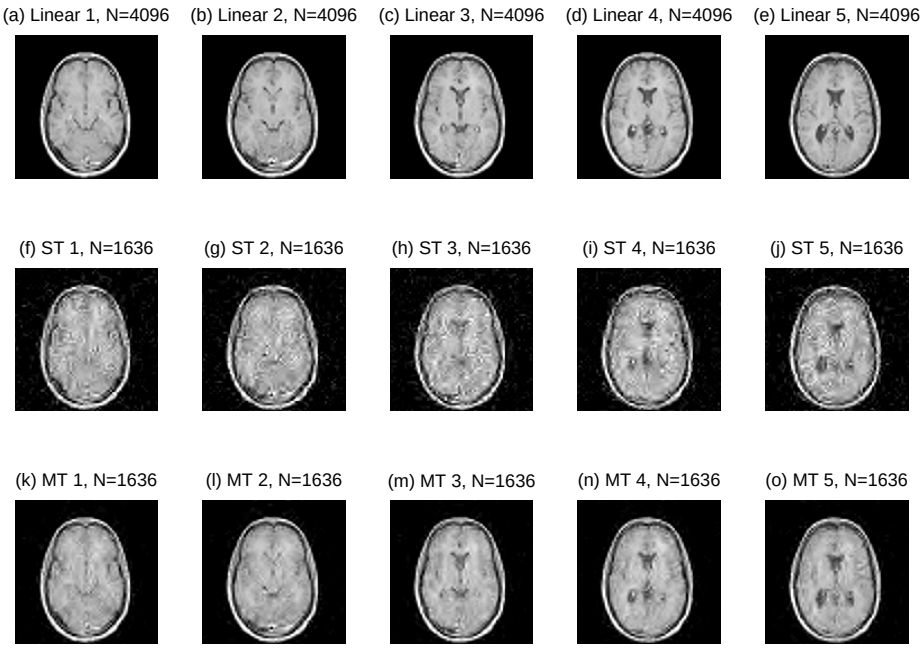


Fig. 6. Reconstruction of MRI images with hybrid CS. (a-e) Linear reconstructions of five original MRI images that are image slices of a human head; (f-j) reconstructed images by ST-BCS*; (k-o) reconstructed images by MT-BCS*.

3) Still Images from Video Sequence: Figure 7 shows the reconstruction results for *Duke Video Images*, which are five snapshots from a web-camera. All five original images have the size 240×256. We used a hybrid CS scheme [48] for image reconstruction, with a coarsest scale $j_0 = 3$, and a finest scale $j_1 = 6$ on the "Daubechies 8" wavelet. Figure 6(a-e) show the results of linear reconstruction with $N = 4096$ samples, which represents the best performance that could be achieved by all the CS implementations considered here. Figures 6(f-j) have results for the ST-BCS* with $N = 1717$ compressed samples for

TABLE II
RECONSTRUCTION PERFORMANCES OF LINEAR, ST-CS AND MT-CS ON MRI IMAGES.

	Recon. Error					Run Time (secs)				
	Task 1	Task 2	Task 3	Task 4	Task 5	Task 1	Task 2	Task 3	Task 4	Task 5
ST-BCS	0.2838	0.2859	0.2808	0.2943	0.2890	263.41	145.35	150.88	256.01	78.26
MT-BCS	0.2019	0.2019	0.2081	0.2079	0.2191	405.77 per task				
ST-BCS*	0.2515	0.2531	0.2658	0.2645	0.2744	262.70	387.95	821.45	158.80	498.38
MT-BCS*	0.1937	0.1937	0.1998	0.1999	0.2099	332.63 per task				
Linear	0.1690	0.1692	0.1777	0.1777	0.1851	0				

each task, whereas Figs. 6(k-o) have the results for MT-BCS*. The full performance comparison between BCS and BCS* is summarized in Table III. Again, the conclusions on the relative performance of the different algorithms are consistent with those from the examples above.

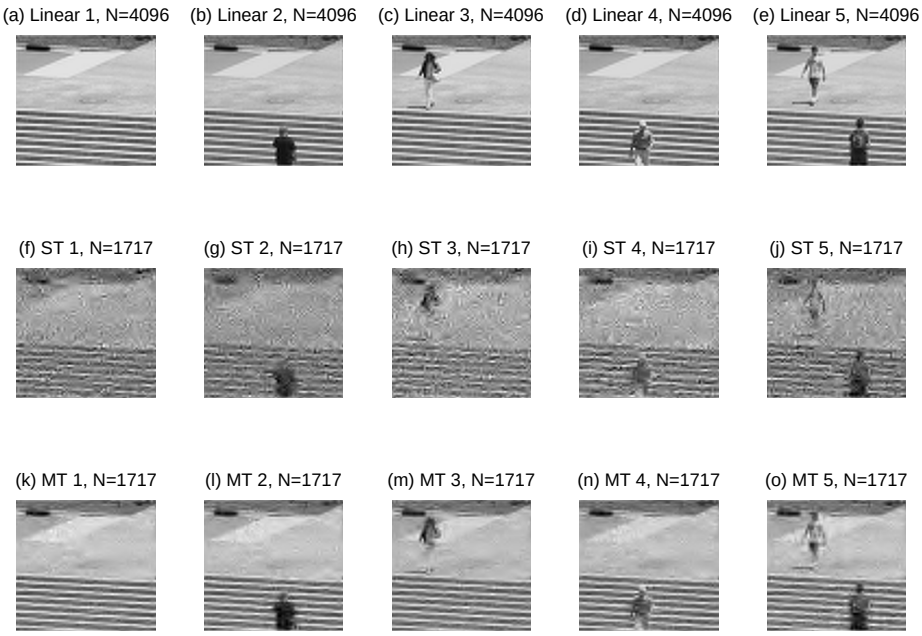


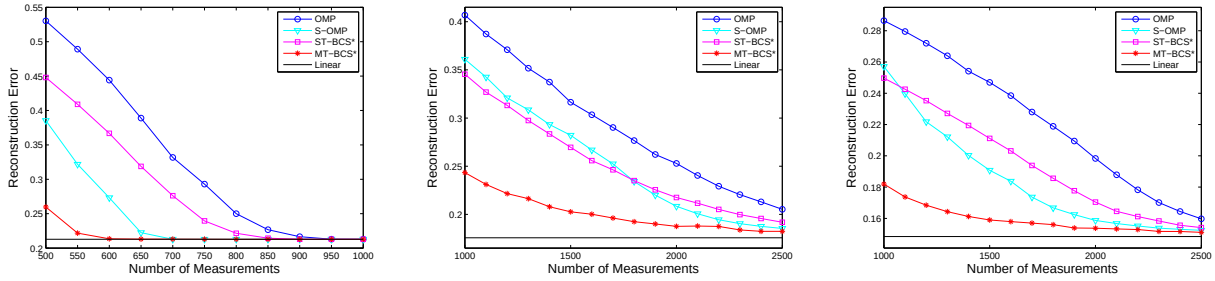
Fig. 7. Reconstruction of video images with hybrid CS. (a-e) Linear reconstructions of five image snapshots from a web-camera; (f-j) reconstructed images by ST-BCS*; (k-o) reconstructed images by MT-BCS*.

In the last set of experiments, we compare the reconstruction errors of OMP and BCS* for single-task (ST) learning and multi-task (MT) learning as a function of number of measurements N on the above three 2D examples, similar to that performed in Fig. 3. Five methods are considered, including OMP [8], S-OMP [47], ST-BCS*, MT-BCS* and linear reconstruction (as a baseline performance). The results are

TABLE III
RECONSTRUCTION PERFORMANCES OF LINEAR, ST-CS AND MT-CS ON VIDEO IMAGES.

	Recon. Error					Run Time (secs)				
	Task 1	Task 2	Task 3	Task 4	Task 5	Task 1	Task 2	Task 3	Task 4	Task 5
ST-BCS	0.2217	0.2090	0.2317	0.2059	0.2221	380.69	776.94	351.91	948.72	319.54
MT-BCS	0.1595	0.1531	0.1605	0.1487	0.1643	430.30 per task				
ST-BCS*	0.2029	0.1993	0.2146	0.1867	0.2080	353.11	328.68	306.40	227.92	529.63
MT-BCS*	0.1591	0.1524	0.1601	0.1485	0.1630	240.63 per task				
Linear	0.1539	0.1449	0.1518	0.1407	0.1508	0				

reported in Fig. 8 for *Random-Bars*, *MRI images* and *Duke video images*, respectively. The improvements of MT-CS over ST-CS are significant on all the three examples considered, and BCS* outperforms OMP both in ST-CS and MT-CS.



(a) (b) (c)

Fig. 8. Reconstruction errors of OMP and BCS* for ST learning and MT learning as a function of increasing N, on (a) Random-Bars, (b) MRI images, and (c) Duke video images. The results are averaged over 10 runs.

V. RELATED WORK

While our motivation to multi-task CS is mainly from a machine-learning perspective, related research has been previously studied in signal processing under the name of "Simultaneous Sparse Approximation" (SSA) [12]–[16] and more recently in compressive sensing with the name "Distributed Compressed Sensing" (DCS) [18]. Most of these previous work extend the existing algorithms, such as Basis Pursuits (BP) [7] or Orthogonal Matching Pursuit (OMP) [8], with a variety of strategies for jointly recovering the nonzero weights. Typical approaches include the S-OMP algorithm in [12], [14], [18], the M-OMP algorithm in [13], the convex relaxation algorithm in [15], [18], and the M-BP algorithm in [13], among others. The exhaustive descriptions of these algorithms are beyond the scope of this paper. However, from

a Bayesian perspective, all these approaches have a similar sharing mechanism that is directed toward the wavelet coefficients, while our Bayesian approach has a sharing mechanism that is directed toward the prior on the wavelet coefficients, i.e., one level higher than the previous methods.

Of more particular relevance, the work of Divorra *et al.* [16] is also related to the multi-task CS problem considered in this paper. Instead of dealing with the multi-task CS problem directly, Divorra *et al.* studied the effect of using *a prior* knowledge for (single) sparse signal approximation, assuming that a reliable *a priori* knowledge about a signal is available. Similar to this paper, their weighted-BPDN and weighted-MP algorithms are motivated from a Bayesian perspective. However, once their modified objective functions are therefore formed, the objective functions are again solved in the way similar to the matching pursuit or convex programming. In comparison, our method is formulated and solved fully in a Bayesian framework, and we learn knowledge about a signal ensemble automatically and transfer information among tasks.

As this paper is under review, we also noticed a related approach [17] has just been published. Similar to our approach, Wipf and Rao [17] also considered an empirical Bayesian strategy for SSA. However, their approach is an extension of a relatively slower version of the RVM [19] (i.e., the iterative algorithm in Sec. II-B1), while our algorithm is a fast sequential optimization approach. In addition, we provide a modified sparse linear-regression model, which marginalizes the noise variance, with improved robustness. In [17], the authors also provided an extensive performance comparison of their Bayesian approach against the other approaches (e.g., M-BP and M-OMP, etc.), and the Bayesian approach demonstrated a superior performance. Although their studies are based on the iterative algorithm of the RVM, these results indeed shed light on the fast implementation considered in this paper, since both implementations are based on the same cost function (13) or similarly (30). Nonetheless, more rigorous experimental comparison among various methods deserves further inquiry. We provided the Matlab code developed in this paper online at `http://www.ece.duke.edu/~shji/BCS.html`, with the hope to make the further comparison convenient.

VI. Conclusions

This paper has analyzed the problem of simultaneous inversion of multiple related signals to enhance the CS reconstructions. Similar problem has been previously studied under the name of "Simultaneous Sparse Approximation" [12]–[16] or "Distributed Compressed Sensing" [18], while here the application of multi-task learning to compressive sensing has been examined. Specifically, a hierarchical Bayesian framework has been introduced to this problem.

Within this framework, two fast inference algorithms extending the relevance vector machine (RVM) have been developed. In particular, a method has been introduced whereby the noise variance in the regression analysis is integrated out analytically. In previous sparse regression analyses of the type considered here a point estimate for the noise variance has been performed, in a ML/MAP sense, along with a ML/MAP estimate of the hyperparameters of the sparseness-promoting prior. By integrating out the noise variance analytically, the associated uncertainty in this parameter is retained throughout, and the resulting algorithm is more robust with respect to the parameter settings than before. This modified fast algorithm has been compared with the original fast RVM algorithm [10], [43], and it has been demonstrated to improve performance, both in single-task and multi-task CS. In fact, the advantages of this new algorithm were shown to be more pronounced in single-task CS (where CS inversion of each image is performed independently), with this attributed to the fact that less data are utilized in such cases; when appropriate, the sharing of data inherent to multi-task learning, between the different tasks, reduces the amount of data required for any one task, and is likely to improve the accuracy of an ML/MAP estimate. In addition, a performance comparison between S-OMP [47] and MT-BCS has been presented, demonstrating superior performance of the Bayesian algorithms.

A significant limitation of the multi-task CS analysis as considered here is the sensitivity of the wavelet coefficients to shifts in the image. This limitation is manifested because the sharing mechanism, as implemented, is directed toward the prior on the wavelet coefficients. Consider two images, with the same basic object (e.g., picture of person) in both images, but the object is significantly shifted in one image with respect to the other. While a human viewing these two images would be able to share information by looking at both, the multi-task CS algorithm presented here would not share information, once the object shift between the two images is sufficient. This suggests that, to generalize the multi-task CS, the sharing mechanism should not be directly on the wavelet coefficients, but rather imposed at a higher level. This is an area of open research, but one may conjecture about possible future directions. For example, rather than placing the shared prior on the wavelet coefficients, one may share a prior on the *statistics* of quadtrees [2]. The sharing in this case is imposed not at the wavelet-coefficient level, but at the quadtree level. Considering the previous example again, the same object shifted within an image may have similar local quadtree statistics, although the location of the similar quadtrees are shifted within the image, commensurate with the associated object shift in the original image. Statistical models such as the hidden Markov tree [49] may be used to model the statistics of the quadtrees, and the multi-task sharing mechanisms may be implemented using more-sophisticated multi-task learning tools than those investigated here. For example, the Dirichlet process [50] has proven to be a very effective tool for

multi-task learning; this type of model is also within the hierarchical Bayesian family, but with far more sophistication and generality than that considered here. Future research may be considered to extend these techniques to multi-task CS with emphasis on computational efficiency and sparse solutions.

APPENDIX A
PROPERTIES OF MARGINAL LIKELIHOOD FUNCTIONS

Some properties of the marginal likelihood functions in (20) and (36) with respect to their approximate solutions are analyzed in this section. A rigorous analysis for $L = 1$ is provided in [51]. However, when $L > 1$, the analysis becomes very complicated due to the combinatory structure of these objective functions. Therefore, in the following, our analysis is based on the approximation that $\alpha_j \ll s_{i,j}$, which has generally been found to be valid numerically, and typically $s_{i,j} > 20\alpha_j$. Although the analysis and the algorithms are developed under this approximation, in practice we did not encounter any problem due to this approximation, except that the algorithms more likely converge to a suboptimal solution.[9]

For the marginal likelihood function in (20), we compute the first derivative of $\mathcal{L}(\boldsymbol{\alpha}, \alpha_0)$ with respect to α_j, which, under the condition of $\alpha_j \ll s_{i,j}$, can be expressed approximately as

$$\frac{\partial \mathcal{L}(\boldsymbol{\alpha}, \alpha_0)}{\partial \alpha_j} = \frac{\partial \ell(\alpha_j)}{\partial \alpha_j} \approx \frac{L}{2\alpha_j} - \sum_{i=1}^{L} \frac{q_{i,j}^2 - s_{i,j}}{2s_{i,j}^2}, \tag{46}$$

and similarly, for the second derivative of $\mathcal{L}(\boldsymbol{\alpha}, \alpha_0)$ with respect to α_j, we have

$$\frac{\partial^2 \mathcal{L}(\boldsymbol{\alpha}, \alpha_0)}{\partial \alpha_j^2} \approx -\frac{L}{2\alpha_j^2}, \tag{47}$$

which is always non-positive. For the case of the finite α_j given by (23), (47) is negative, and therefore $\ell(\alpha_j)$ must have a *unique* maximum at this approximate solution. However, this is achieved under the approximation that $\alpha_j \ll s_{i,j}$. In reality, without this approximation, the exact solving of (22) may yield two or more positive solutions of α_j, one of which is observed relatively close to the approximate solution given by (23). Therefore, (23) must be at the vicinity of a stationary point of $\ell(\alpha_j, \alpha_0)$, which may only correspond to a local maximum of $\ell(\alpha_j, \alpha_0)$.

Similarly, for the marginal likelihood function in (36), under the condition of $\alpha_j \ll s_{i,j}$, the first and

[9]Another reason for the suboptimal solution is due to the greedy property of the algorithm, including the case when $L = 1$.

second derivatives of $\mathcal{L}(\boldsymbol{\alpha})$ with respect to α_j, can be expressed, respectively, as

$$\frac{\partial \mathcal{L}(\boldsymbol{\alpha})}{\partial \alpha_j} \approx \frac{L}{2\alpha_j} - \sum_{i=1}^{L} \frac{(N_i + 2a)q_{i,j}^2/g_{i,j} - s_{i,j}}{2s_{i,j}(s_{i,j} - q_{i,j}^2/g_{i,j})}, \tag{48}$$

$$\frac{\partial^2 \mathcal{L}(\boldsymbol{\alpha})}{\partial \alpha_j^2} \approx -\frac{L}{2\alpha_j^2}. \tag{49}$$

Following the same analysis as above, the approximate solution given by (39) must be at the vicinity of a stationary point of $\ell(\alpha_j)$, which may correspond to a local maximum of $\ell(\alpha_j)$.

APPENDIX B
EFFICIENT CALCULATIONS FOR SEQUENTIAL OPTIMIZATION

In the implementation of the fast algorithm in Sec. III-B2, it is necessary to recompute $\boldsymbol{\Sigma}_i$, $\boldsymbol{\mu}_i$, and all quantities $s_{i,j}$, $q_{i,j}$ and $g_{i,j}$. For the sequential nature of the fast algorithm, these quantities can be calculated iteratively. In addition, we must calculate the increase or decrease of the marginal likelihood $\mathcal{L}(\boldsymbol{\alpha}) = \sum_{i=1}^{L} \mathcal{L}_i(\boldsymbol{\alpha})$ according to which basis functions are added, deleted or re-estimated. Efficient calculations of these quantities are given below.

A. Notation

The fast algorithm operates in a constructive manner, i.e., at each step t it may add a basis to the model, or delete a basis from the model, or re-estimate the parameters of the model. Therefore, $\boldsymbol{\Phi}_i$ as used below need only comprise columns of included basis functions. Denote the number of the basis functions in $\boldsymbol{\Phi}_i$ at step t as M_t, so $\boldsymbol{\Phi}_i$ is of size $N_i \times M_t$. Similarly, $\boldsymbol{\Sigma}_i$ and $\boldsymbol{\mu}_i$ are computed only for the "current" basis and therefore are of order M_t (all other entries in the "full" version of $\boldsymbol{\Sigma}_i$ and $\boldsymbol{\mu}_i$ would be zero). The integer $j \in \{1, 2, \ldots, M\}$ is used to index the single basis function for which α_j is to be updated, and the integer $k \in \{1, 2, \ldots, M_t\}$ to denote the index within the current basis that corresponds to j. The index $l \in \{1, 2, \ldots, M\}$ ranges over all basis functions, including those not currently utilized in the model. For convenience, define $K_i = N_i + 2a$. Updated quantities are denoted by a tilde (e.g., $\widetilde{\alpha}_i$).

B. Adding a new basis function

$$2\Delta\mathcal{L}_i = \log\frac{\alpha_j}{\alpha_j+s_{i,j}} - K_i\log\left(1-\frac{q_{i,j}^2/g_{i,j}}{\alpha_j+s_{i,j}}\right), \tag{50}$$

$$\widetilde{\boldsymbol{\Sigma}}_i = \begin{bmatrix} \boldsymbol{\Sigma}_i+\Sigma_{i,(jj)}\boldsymbol{\Sigma}_i\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_{i,j}\boldsymbol{\Phi}_{i,j}^T\boldsymbol{\Phi}_i\boldsymbol{\Sigma}_i & -\Sigma_{i,(jj)}\boldsymbol{\Sigma}_i\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_{i,j} \\ -\Sigma_{i,(jj)}(\boldsymbol{\Sigma}_i\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_{i,j})^T & \Sigma_{i,(jj)} \end{bmatrix}, \tag{51}$$

$$\widetilde{\boldsymbol{\mu}}_i = \begin{bmatrix} \boldsymbol{\mu}_i-\mu_{i,j}\boldsymbol{\Sigma}_i\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_{i,j} \\ \mu_{i,j} \end{bmatrix}, \tag{52}$$

$$\widetilde{S}_{i,l} = S_{i,l}-\Sigma_{i,(jj)}(\boldsymbol{\Phi}_{i,l}^T\boldsymbol{e}_{i,j})^2, \tag{53}$$

$$\widetilde{Q}_{i,l} = Q_{i,l}-\mu_{i,j}(\boldsymbol{\Phi}_{i,l}^T\boldsymbol{e}_{i,j}), \tag{54}$$

$$\widetilde{G}_i = G_i-\Sigma_{i,(jj)}(\boldsymbol{v}_i^T\boldsymbol{e}_{i,j})^2. \tag{55}$$

where $\Sigma_{i,(jj)} = (\alpha_{i,j}+S_{i,j})^{-1}$ is the jth diagonal element of $\boldsymbol{\Sigma}_i$, $\mu_{i,j} = \Sigma_{i,(jj)}Q_{i,j}$ and we define $\boldsymbol{e}_{i,j} \triangleq \boldsymbol{\Phi}_{i,j}-\boldsymbol{\Phi}_i\boldsymbol{\Sigma}_i\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_{i,j}$.

C. Re-estimating a basis function

$$2\Delta\mathcal{L}_i = (K_i-1)\log(1+S_{i,j}(\widetilde{\alpha}_j^{-1}-\alpha_j^{-1}))+K_i\log\frac{[(\alpha_j+s_{i,j})g_{i,j}-q_{i,j}^2]\widetilde{\alpha}_j}{[(\widetilde{\alpha}_j+s_{i,j})g_{i,j}-q_{i,j}^2]\alpha_j}, \tag{56}$$

$$\widetilde{\boldsymbol{\Sigma}}_i = \boldsymbol{\Sigma}_i-\gamma_{i,k}\boldsymbol{\Sigma}_{i,k}\boldsymbol{\Sigma}_{i,k}^T, \tag{57}$$

$$\widetilde{\boldsymbol{\mu}}_i = \boldsymbol{\mu}_i-\gamma_{i,k}\mu_{i,k}\boldsymbol{\Sigma}_{i,k}, \tag{58}$$

$$\widetilde{S}_{i,l} = S_{i,l}+\gamma_{i,k}(\boldsymbol{\Sigma}_{i,k}^T\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_{i,l})^2, \tag{59}$$

$$\widetilde{Q}_{i,l} = Q_{i,l}+\gamma_{i,k}\boldsymbol{\mu}_{i,k}(\boldsymbol{\Sigma}_{i,k}^T\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_{i,l}), \tag{60}$$

$$\widetilde{G}_i = G_i+\gamma_{i,k}(\boldsymbol{\Sigma}_{i,k}^T\boldsymbol{\Phi}_i^T\boldsymbol{v}_i)^2. \tag{61}$$

where $\boldsymbol{\Sigma}_{i,k}$ as the kth column of $\boldsymbol{\Sigma}_i$, and we define $\gamma_{i,k} \triangleq (\Sigma_{i,(kk)}+(\widetilde{\alpha}_j-\alpha_j)^{-1})^{-1}$.

D. Deleting a basis function

$$2\Delta\mathcal{L}_i = -K_i \log\left(1 + \frac{Q_{i,j}^2/G_i}{\alpha_j - S_{i,j}}\right) - \log\left(1 - \frac{S_{i,j}}{\alpha_j}\right), \tag{62}$$

$$\widetilde{\boldsymbol{\Sigma}}_i = \boldsymbol{\Sigma}_i - \frac{1}{\Sigma_{i,(kk)}} \boldsymbol{\Sigma}_{i,k} \boldsymbol{\Sigma}_{i,k}^T, \tag{63}$$

$$\widetilde{\boldsymbol{\mu}}_i = \boldsymbol{\mu}_i - \frac{\mu_{i,k}}{\Sigma_{i,(kk)}} \boldsymbol{\Sigma}_{i,k}, \tag{64}$$

$$\widetilde{S}_{i,l} = S_{i,l} + \frac{1}{\Sigma_{i,(kk)}} (\boldsymbol{\Sigma}_{i,k}^T \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_{i,l})^2, \tag{65}$$

$$\widetilde{Q}_{i,l} = Q_{i,l} + \frac{\mu_{i,k}}{\Sigma_{i,(kk)}} (\boldsymbol{\Sigma}_{i,k}^T \boldsymbol{\Phi}_i^T \boldsymbol{\Phi}_{i,l}), \tag{66}$$

$$\widetilde{G}_i = G_i + \frac{1}{\Sigma_{i,(kk)}} (\boldsymbol{\Sigma}_{i,k}^T \boldsymbol{\Phi}_i^T \boldsymbol{v}_i)^2. \tag{67}$$

Following updates (63) and (64), the appropriate row and/or column k is removed from $\widetilde{\boldsymbol{\Sigma}}_i$ and $\widetilde{\boldsymbol{\mu}}_i$.

ACKNOWLEDGEMENT

The authors wish to thank the anonymous reviewers for their constructive suggestions and for pointing out the related research in the signal processing literature. The authors also thank I. Pruteanu for providing the video images in the experiments. This work was supported by the Office of Naval Research and the Defense Advanced Research Project Agency (DARPA) under the Mathematical Time Reversal program.

REFERENCES

[1] I. Daubechies, *Ten lectures on wavelets*. SIAM, 1992.

[2] S. Mallat, *A wavelet tour of signal processing*, 2nd ed. Academic Press, 1998.

[3] A. Said and W. A. Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Systems for Video Technology*, vol. 6, pp. 243–250, 1996.

[4] W. A. Pearlman, A. Islam, N. Nagaraj, and A. Said, "Efficient, low-complexity image coding with a set-partitioning embedded block coder," *IEEE Trans. Circuits Systems Video Technology*, vol. 14, pp. 1219–1235, Nov. 2004.

[5] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Information Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[6] D. L. Donoho, "Compressed sensing," *IEEE Trans. on Information Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[7] S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.

[8] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Trans. on Information Theory*, 2007, Preprint.

[9] D. L. Donoho, Y. Tsaig, I. Drori, and J.-C. Starck, "Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit," Mar. 2006, Preprint.

[10] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Trans. on Signal Processing*, 2007, to appear.

[11] M. Figueiredo, R. D. Nowak, and S. J. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," 2007, Preprint.

[12] D. Leviatan and V. N. Temlyakov, "Simultaneous approximation by greedy algorithms," Univ. South Carolina, Dept. Math., Columbia, SC, Tech. Rep., 2003.

[13] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Trans. on Signal Processing*, vol. 53, no. 7, pp. 2477–2488, July 2005.

[14] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, "Algorithms for simultaneous sparse approximation. Part I: Greedy pursuit," *Signal Processing*, vol. 86, pp. 572–588, Apr. 2006.

[15] J. A. Tropp, "Algorithms for simultaneous sparse approximation. Part II: Convex relaxation," *Signal Processing*, vol. 86, pp. 589–602, Apr. 2006.

[16] Òscar Divorra Escoda, L. Granai, and P. Vandergheynst, "On the use of a priori information for sparse signal approximations," *IEEE Transactions on Signal Processing*, vol. 54, no. 9, pp. 3468–3482, Sept. 2006.

[17] D. P. Wipf and B. D. Rao, "An empirical Bayesian strategy for solving the simultaneous sparse approximation problem," *IEEE Trans. on Signal Processing*, vol. 55, no. 7, pp. 3704–3716, July 2007.

[18] D. Baron, M. B. Wakin, M. F. Duarte, S. Sarvotham, and R. G. Baraniuk, "Distributed compressed sensing," Nov. 2005, Preprint.

[19] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.

[20] M. Figueiredo, "Adaptive sparseness using Jeffreys prior," in *Advances in Neural Information Processing Systems (NIPS 14)*, 2002.

[21] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal. Statist. Soc B.*, vol. 58, no. 1, pp. 267–288, 1996.

[22] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[23] R. Caruana, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[24] J. Baxter, "Learning internal representations," in *COLT: Proceedings of the Workshop on Computational Learning Theory*, 1995.

[25] ——, "A model of inductive bias learning," *Journal of Artificial Intelligence Research*, 2000.

[26] N. D. Lawrence and J. C. Platt, "Learning to learn with the informative vector machine," in *Proc. of the 21st International Conference on Machine Learning (ICML 21)*, 2004.

[27] K. Yu, V. Tresp, and A. Schwaighofer, "Learning Gaussian processes from multiple tasks," in *Proc. of the 22nd International Conference on Machine Learning (ICML 22)*, 2005.

[28] J. Zhang, Z. Ghahramani, and Y. Yang, "Learning multiple related tasks using latent independent component analysis," in *Advances in Neural Information Processing Systems 18*, 2005.

[29] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *Journal of Machine Learning Research*, vol. 6, pp. 1817–1853, 2005.

[30] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *Journal of Machine Learning Research*, vol. 6, pp. 615–637, 2005.

[31] D. Burr and H. Doss., "A Bayesian semiparametric model for random-effects meta-analysis," *Journal of the American Statistical Association*, vol. 100, no. 469, pp. 242–251, Mar. 2005.

[32] F. Dominici, G. Parmigiani, R. Wolpert, and K. Reckhow, "Combining information from related regressions," *Journal of Agricultural, Biological, and Environmental Statistics*, vol. 2, no. 3, pp. 294–312, 1997.

[33] P. D. Hoff, "Nonparametric modeling of hierarchically exchangeable data," University of Washington Statistics Department, Tech. Rep. 421, 2003.

[34] P. Müller, F. Quintana, and G. Rosner, "A method for combining inference across related nonparametric Bayesian models," *Journal of the Royal Statistical Society Series B*, vol. 66, no. 3, pp. 735–749, 2004.

[35] B. K. Mallick and S. G. Walker, "Combining information from several experiments with nonparametric priors," *Biometrika*, vol. 84, no. 3, pp. 697–706, 1997.

[36] C. M. Bishop and M. E. Tipping, "Variational relevance vector machines," in *Proc. of the 16th Conference on Uncertainty in Artificial Intelligence (UAI 16)*, 2000, pp. 46–53.

[37] D. P. Wipf and B. D. Rao, "Comparing the effects of different weight distributions on finding sparse representations," in *Advances in Neural Information Processing Systems 18*, 2006.

[38] D. P. Wipf and S. Nagarajan, "A new view of automatic relevance determination," in *Advances in Neural Information Processing Systems 20*, 2008.

[39] B. P. Carlin and T. A. Louis, *Bayes and Empirical Bayes Methods for Data Analysis*. Chapman & Hall-CRC, 2000.

[40] D. J. C. MacKay, "Bayesian interpolation," *Neural Computation*, vol. 4, no. 3, pp. 415–447, 1992.

[41] D. MacKay, "Bayesian interpolation," *Neural Computation*, vol. 4, no. 3, pp. 415–447, 1992.

[42] D. Wipf and B. Rao, "Sparse Bayesian learning for basis selection," *IEEE Trans. Signal Processing*, vol. 52, no. 8, pp. 2153–2164, Aug. 2004.

[43] M. E. Tipping and A. C. Faul, "Fast marginal likelihood maximisation for sparse Bayesian models," in *Proc. of the 9th International Workshop on AIStats*, C. M. Bishop and B. J. Frey, Eds., 2003.

[44] D. B. Dunson, "Empirical Bayes density regression," *Statistica Sinica*, vol. 17, pp. 481–504, 2007.

[45] M. Svensén and C. M. Bishop, "Robust Bayesian mixture modelling," *Neurocomputing*, vol. 64, pp. 235–252, 2004.

[46] S. Yu, V. Tresp, and K. Yu, "Robust multi-task learning with t-processes," in *Proc. of the 24th International Conference on Machine Learning (ICML 24)*, 2007.

[47] J. A. Tropp, A. C. Gilbert, and M. J. Strauss, "Simultaneous sparse approximation via greedy pursuit," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 5, Mar. 2005, pp. 721–724.

[48] Y. Tsaig and D. L. Donoho, "Extensions of compressed sensing," *Signal Processing*, vol. 86, no. 3, pp. 549–571, Mar. 2006.

[49] M. Crouse, R. Nowak, and R. Baraniuk, "Wavelet-based statistical signal processing using hidden markov models," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 886–902, April 1998.

[50] T. S. Ferguson, "A Bayesian analysis of some nonparametric problems," *Annals of Statistics*, vol. 1, no. 2, pp. 209–230, 1973.

[51] A. C. Faul and M. E. Tipping, "Analysis of sparse Bayesian learning," in *Advances in Neural Information Processing Systems (NIPS 14)*, 2002.