Clarify Some Issues on the Sparse Bayesian Learning for Sparse Signal Recovery

Zhilin Zhang and Bhaskar D. Rao
Technical Report
University of California at San Diego
September, 2011

***Abstract*— Sparse Bayesian learning (SBL) is an important family of algorithms for sparse signal recovery and compressed sensing. It has shown superior recovery performance in challenging practical problems, such as highly underdetermined inverse problems, recovering signals with less sparsity, recovering signals based on highly coherent measuring/sensing/dictionary matrices, and recovering signals with rich structure. However, its advantages are smeared in current literature due to some misunderstandings on the parameters of SBL and incorrect parameter settings in algorithm comparison and practical use. This work clarifies some important issues, and serves as a guidance for correctly using SBL.**

***Index Terms*— Sparse Bayesian Learning (SBL), Sparse Signal Recovery, Compressed Sensing, Sparse Representation**

I. Introduction to Sparse Bayesian Learning

The sparse signal recovery problem can be mathematically expressed as

$$\mathbf{Y} = \mathbf{\Phi}\mathbf{X} + \mathbf{V}, \tag{1}$$

where $\mathbf{Y} \triangleq [\mathbf{Y}_{\cdot 1}, \cdots, \mathbf{Y}_{\cdot L}] \in \mathbb{R}^{N \times L}$ is an available measurement matrix (data matrix), $\mathbf{\Phi} \in \mathbb{R}^{N \times M}$ is a known matrix, and $\mathbf{X}$ is an unknown coefficient matrix (in literature it is also called solution matrix or source matrix). $\mathbf{V}$ is an unknown noise matrix. The model is called the single measurement vector (SMV) model when $L = 1$, which is the most common model in compressed sensing. When $L > 1$, it is called the multiple measurement vector (MMV) model [1].

Numerous algorithms have been proposed for the model (1). Among them, sparse Bayesian learning (SBL) [2]–[10] is one important family. In the general SBL framework [7], each coefficient row in $\mathbf{X}$ is assumed to have the parameterized Gaussian distribution

$$p(\mathbf{X}_{i\cdot}; \gamma_i, \mathbf{B}_i) = \mathcal{N}(0, \gamma_i \mathbf{B}_i), \quad i = 1, \cdots, M \tag{2}$$

where γ_i and $\mathbf{B}_i$ are hyperparameters. $\mathbf{B}_i$ is a positive definite and symmetric matrix, functioning as a regularizer [1]. To prevent overfitting, the constraint $\mathbf{B}_i = \mathbf{B} (\forall i)$ is widely used [6], [7]. γ_i is a nonnegative scalar, controlling the row-sparsity of $\mathbf{X}$. When $\gamma_i = 0$, the corresponding i-th row of $\mathbf{X}$ becomes zeros. During the learning procedure, most $\gamma_i (\forall i)$ tend to zero, due to the mechanism of automatic relevance determination [2], [11]. Thus the row-sparsity is encouraged. It also has been mathematically shown that the global solution is the sparsest solution [7]. The sparsity-encouraging mechanism can also be seen after transferring the ordinary SBL algorithms (optimizing in the γ parameter space) to equivalent forms optimizing in the $\mathbf{X}$ parameter space [6], [12]–[14]. Note that in the presence of noise, γ_i will never be zero. Thus a threshold θ will be used to prune out small $\gamma_i (\forall i)$. Generally, $\theta = 10^{-2} \sim 10^{-4}$. Besides, each element of the noise matrix $\mathbf{V}$ is assumed to be Gaussian: $p(\mathbf{V}_{ij}) = \mathcal{N}(0, \lambda)$.

Recently SBL has drawn much attention due to its super ability to handle challenging problems, such as highly underdetermined inverse problems, recovering signals with less sparsity, and recovering signals based on highly coherent measuring/sensing/dictionary matrices. However, due to some misunderstandings on the parameters of SBL, its ability is smeared. In the following we will clarify these issues.

II. Clarify the noise variance term λ

One misunderstanding is the λ, which models the noise variance in the SBL framework (1). In computer simulations and practical problems, people thus set λ to the true noise variance or the estimated noise variance [15], believing that this setting can allow SBL algorithms to reach their full strength. But we have to point out that the optimal value of λ is not the true noise variance.

Here we carry out a simple experiment to show that the optimal value of λ is largely different to the true noise variance.

The Gaussian dictionary matrix $\mathbf{\Phi}$ was of size 40 by 120. The number of measurement vectors, L, was 3. The true noise variance was 0.01 (SNR was around 10dB). The number of nonzero rows in the coefficient matrix, K, was set to be 4, 12, and 16. For each different K, the experiment was repeated 200 times. In each trial, the M-SBL algorithm [5] was fed with different values of λ. Its performance was measured by two measures: the failure rate, defined in [5], and the mean square error (MSE). The performance curves of M-SBL are shown in Fig.1, from which we have the following observations:

1) The optimal value of λ, which corresponds to the best performance, is not equal to the true noise variance.
2) Different measures correspond to different optimal values of λ. In other words, for given experiment settings, the optimal value of λ in terms of MSE is different to

Z.Zhang and B.D.Rao are with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093-0407, USA. Email:{z4zhang,brao}@ucsd.edu.

[1]In some algorithms $\mathbf{B}_i (\forall i)$ are the identity matrix [5].

the optimal one in terms of the failure rate. We found this observation is more obvious when using T-SBL/T-MSBL [7].

3) Changing any experiment setting (e.g. K, L, N, M), the optimal value of λ may accordingly change.

Next we will see that for different SBL algorithms, under the same experiment settings, the optimal values of λ are different.

III. Clarify the λ learning rules

In the previous section we discussed the issue of setting λ to a fixed value, and we now understand that changing an experiment setting requires to seek the optimal value again. This brings much inconvenience in practical use and computer simulations. Fortunately, many SBL algorithms have their own learning rules for λ. However, we emphasize that many of these learning rules are sensitive to strong noise, and thus the estimate of λ cannot allow the algorithms to achieve the best recovery performance.

To clearly see this, we carry out the following experiment. The experiment is a comparison of 4 SBL algorithms in the single measurement vector (SMV) model (i.e. $L = 1$ in the model (1)) in a noisy environment. The noise variance was 0.01. The 4 SBL algorithms were EM-SBL [4], ExCoV [8], BCS [9], and T-MSBL [7]. Note that although T-MSBL was derived for the multiple measurement vector model, it can also be used in the SMV model. In this case, T-MSBL is similar to EM-SBL. But their key difference is the λ learning rule. The dictionary matrix $\mathbf{\Phi}$ is a Gaussian random matrix of the size 30×80. The number of nonzero elements $K = 5$. The nonzero elements are generated using the Matlab command: $\mathrm{sign}(\mathrm{randn}(K, 1)). * (\mathrm{rand}(K, 1) * 0.5 + 0.5)$.

First, let's see how the λ's value affects their recovery performance. We didn't use their λ learning rules. Instead, we fed them with fixed λ values (ranging from 0.001 to 0.33). Note in this case, EM-SBL and T-MSBL had the same performance curve. The performance curves of these algorithms as functions of λ are plotted in Fig.2 using red, blue, and green solid lines.

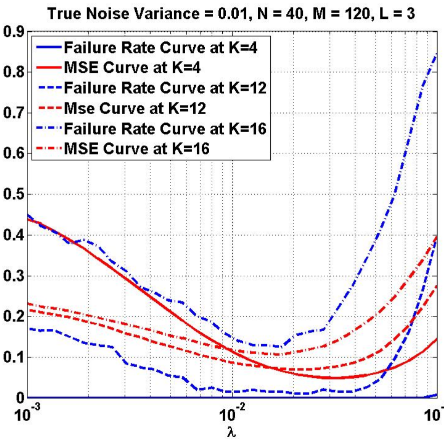

Fig. 1. M-SBL's performance as functions of λ.

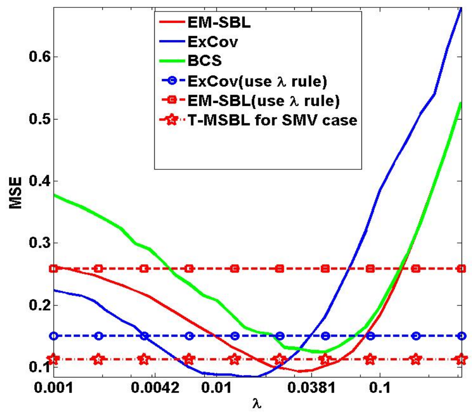

Fig. 2. The effect of fixed λ values and λ learning rules on the performance of T-MSBL, EM-SBL, ExCov and BCS.

As we can see, if we could obtain the optimal λ for each algorithm, then EM-SBL(T-MSBL for SMV) and ExCov had the similar performance, while BCS had the poorer performance. However, if we chose a wrong λ, say $\lambda = 0.0381$ (this value was calculated according to the suggestion in [16]), then we might conclude that the EM-SBL had the best performance while ExCov had the worst performance. But if we chose $\lambda = 0.0042$ (this value was calculated according to the suggestion in [**?**]), then we might conclude that ExCov had the best performance while BCS had the worst performance. So, unthoughtful choices of λ may lead to wrong conclusions. Again, we've seen the noise variance (0.01) was not the optimal lambda values of all the SBL algorithms.

Next, let's see how λ learning rules affect the recovery performance. The dashed lines in Fig.2 show the performance of EM-SBL, ExCov and T-MSBL when they used their λ learning rules (The code of BCS does not provide the λ learning rule, so we did not compare it here). Clearly, we can see all the λ learning rules could not allow the associated SBL algorithms to achieve their full strength. We also see the λ learning rule of EM-SBL resulted in very poor performance, so poor that even setting λ to a random guessed value might lead to better performance. In contrast, the λ learning rule of T-MSBL was the most effective one, which led to near-optimal performance.

Base on the above observations, we believe the work of deriving more effective λ learning rules has the same value as the work of deriving new SBL algorithms. Take the ExCov and the EM-SBL for example. In the above experiment, when both algorithms chose their optimal λ values, ExCov had slightly better performance than EM-SBL. On the other hand, if both algorithms used their λ learning rules, ExCov had much better performance than EM-SBL. But if EM-SBL used the λ learning rule of T-MSBL, EM-SBL could have better performance than ExCov (see the performance curve denoted by '*T-MSBL for SMV case*').

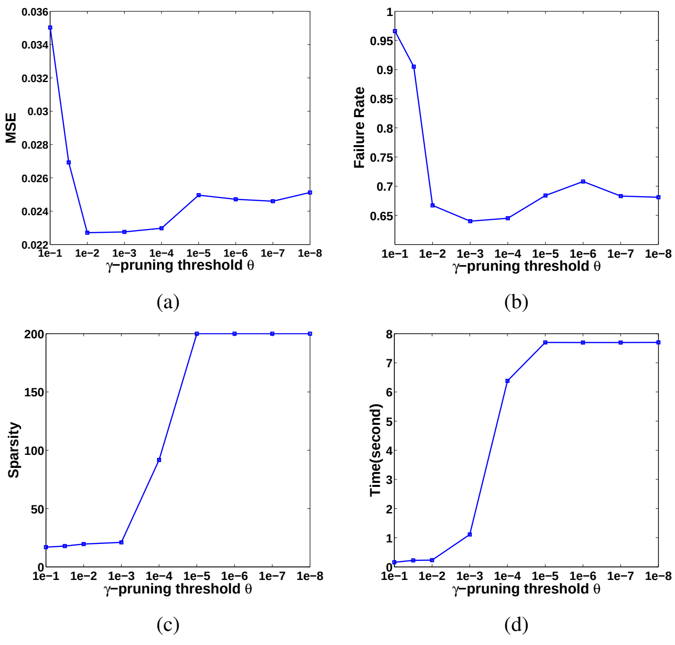

Fig. 3. Effects of the γ-pruning threshold θ.

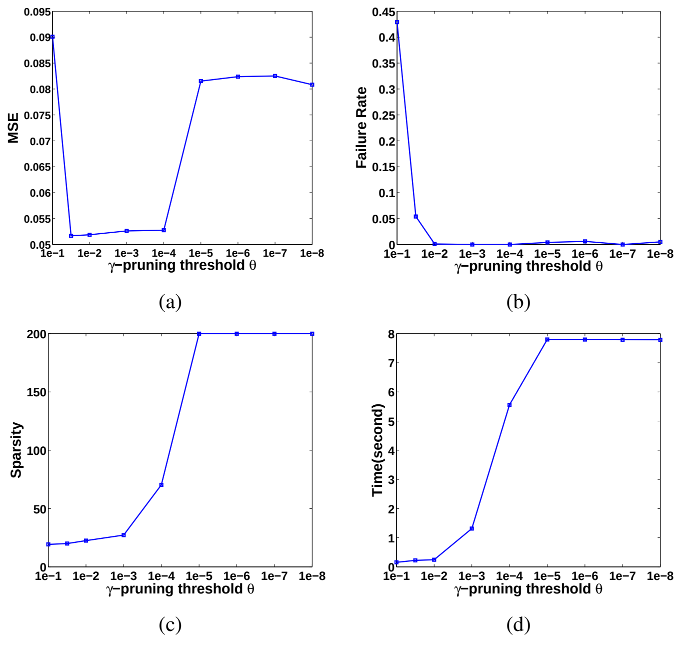

Fig. 4. Effects of the γ-pruning threshold θ when all the nonzero rows of $\mathbf{X}$ were normalized.

IV. Effect of the Threshold to Prune Out γ_i

Many SBL algorithms have the pruning mechanism, namely, during the learning procedure, once a γ_i for some i is smaller than a user-defined threshold, θ, the γ_i is pruned out and the associated i-th coefficient (in the SMV model) or the i-th coefficient row (in the MMV model) is set to zero and never used again. It has been shown [3], [4], [17] that in noiseless cases the final values of γ_j $(\forall j)$ after convergence are either zeros or some positive values [2]. Therefore, in noiseless experiments the threshold θ can be arbitrarily small, e.g. $\theta = 10^{-5}$ or $\theta = 10^{-10}$. However, in noisy cases, one needs to be very careful to choose the value of θ. Unfortunately, this issue didn't draw enough attention and resulted in wrong conclusions in some published works.

We point out that setting the value of θ in a suitable range significantly affects the performance measured by various performance indexes.

To see this, we carry out the following experiment, in which we consider an MMV model with temporally correlated source vectors. T-MSBL [7] was used here (Using other SBL algorithms lead to the same conclusions.). The matrix $\mathbf{\Phi}$ was of the size 100×200. The coefficient matrix $\mathbf{X}$ was of the size 200×3. Its each nonzero row was generated from a Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma} \triangleq \mathrm{Toeplitz}([1, \beta, \beta^2])$ with $\beta = 0.8$. The number of nonzero rows was 20. Noise variance was 0.01. We measured the algorithm's performance in terms of failure rates, MSE, speed, and sparsity (the number of nonzero rows). We considered various cases, in which the threshold θ chose the values: $10^{-1}, 5 \times 10^{-2}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}$.

Results are shown in Fig.3 and Fig.4. In Fig.4 each nonzero row of $\mathbf{X}$ was normalized to have unit Euclidean norm. We have the following observations and comments.

Remark 1: Larger θ values result in faster speed (Fig.3(d) and Fig.4 (d)). Obviously, the faster speed is due to the fact that a larger number of $\gamma_i (\forall i)$ are pruned out by a larger θ and thus the algorithm effectively operates on a smaller sized parameter space. Note that the running time can differ in several orders-of-magnitude when using different θ values.

Remark 2: Larger θ values result in fewer nonzero rows in the estimated $\widehat{\mathbf{X}}$ (Fig.3(c) and Fig.4(c)). In other words, when θ is very small, there are many γ_i not pruned out, which correspond to zero rows in $\mathbf{X}$. As a result, there are lots of nonzero 'residual rows' in $\widehat{\mathbf{X}}$, which should be zero rows. But note that these residual rows have very tiny elements. To see this, we plot a result of a trial in Fig. 5, in which the ℓ_2 norm of each row of the estimated $\widehat{\mathbf{X}}$ is calculated and plotted. The top subfigure corresponds to the true $\mathbf{X}$, the middle subfigure corresponds to the case when $\theta = 10^{-2}$, and the bottom subfigure corresponds to $\theta = 10^{-5}$. When $\theta = 10^{-2}$, the number of nonzero rows in $\widehat{\mathbf{X}}$ was 20, while when $\theta = 10^{-5}$, the number of nonzero rows in $\widehat{\mathbf{X}}$ was 182. However, by comparing the middle and the bottom subfigures, we cannot find any discernable difference. This is because the ℓ_2 norms of the 'residual rows' (or equivalently, their associated γ_i) are smaller than the ℓ_2 norms of true nonzero rows (or equivalently, the associated γ_i) by several orders-in-magnitude (see Fig.6).

These residual rows generally have negligible negative effect on the recovery performance in some applications such as source localization, DOA estimation and power spectrum estimation. In these fields, people generally are only interested in the nonzero coefficient rows with large ℓ_2 norms (and thus the failure rate is preferred to MSE).

However, in other applications such as compressed sensing of videos and MRI image series, people more tend to use

[2]The original literature [3], [17] showed $1/\gamma_i$ is either infinity or positive.

MSE as performance measure. In these applications the MSE of algorithms generally ranges from -15dB to -30dB. In this case, the residual rows can have significant effect on the MSE. Figure 7 shows a comparison result using a large-scale dataset. In the experiment, the matrix $\mathbf{\Phi}$ was of the size $N \times M$ with M fixed to 5000, where N varied such that M/N ranged from 5 to 25. The number of nonzero rows in $\mathbf{X}$ was $N/3$. SNR was 25dB. The number of measurement vectors was $L = 4$. The figure shows that the MSE differed largely when θ chose 10^{-2} and 10^{-3}, respectively.

Fortunately, since the ℓ_2 norms of the residual rows are smaller than the ℓ_2 norms of the true nonzero rows by several orders-in-magnitude, we can easily remove most of these residual rows. For example, we can sort the $\gamma_i(\forall i)$ and plot the γ spectrum, as in Fig.6. Then look for the corner point in the γ spectrum. All the nonzero rows whose associated γ_i are smaller than the corner point are set to zeros.

Remark 3: The optimal value of θ associated with the best recovery performance (in terms of both MSE and failure rates) is around the noise variance (0.01); fortunately, values in a relatively wide range ($\theta = 10^{-2} \sim 10^{-4}$) also result in near-optimal performance (Fig.3(a)-(b) and Fig.4(a)-(b)). So, in practice one just needs to roughly know the range of noise variance to choose a good value for the threshold. Generally, $\theta = 10^{-3}$ is a good value for most practical applications when noise is presented.

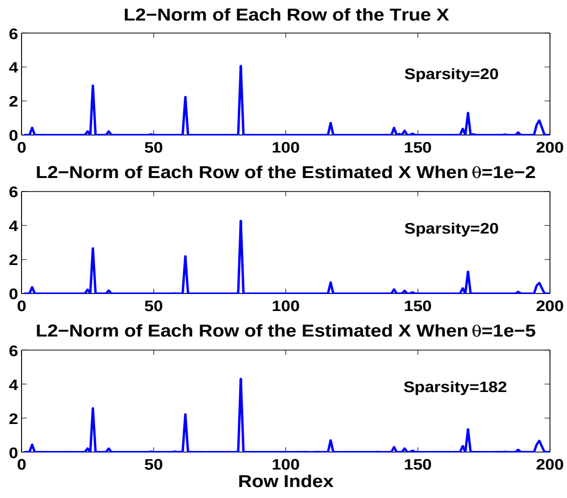

Fig. 5. Compare the 2-norms of rows of $\mathbf{X}$ (top) to those of $\widehat{\mathbf{X}}$ when $\theta = 10^{-2}$ (middle) and $\theta = 10^{-5}$ (bottom).

V. Conclusion

Sparse Bayesian learning is a group of algorithms with super ability to handle challenging sparse signal recovery problems and compressed sensing problems. However, due to some misunderstandings on its parameters and inaccurate settings of these parameters, its advantages are greatly discounted. In this work we discussed three issues of SBL algorithms, i.e. the λ value, the λ learning rule, and the γ-pruning threshold. Some understandings on these parameters are clarified.

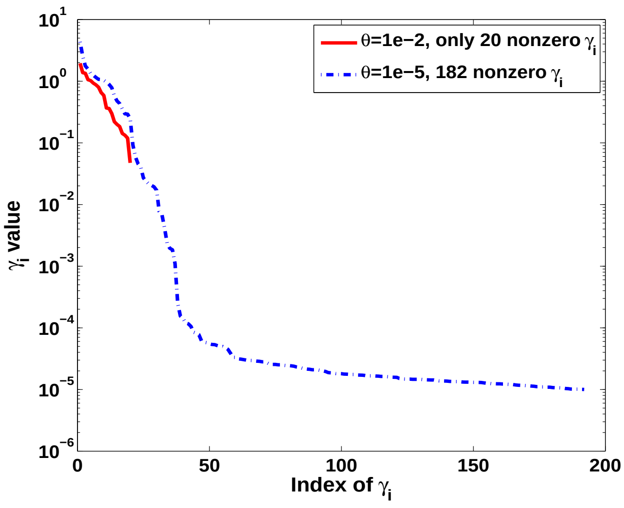

Fig. 6. Comparison of the sorted γ_i values when $\theta = 10^{-2}$ and $\theta = 10^{-5}$.

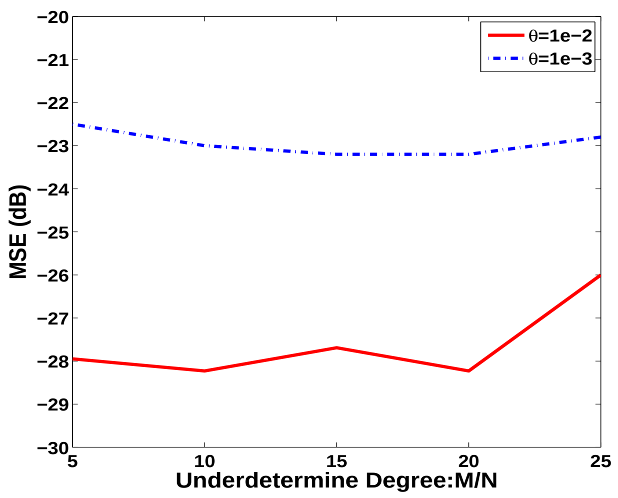

Fig. 7. Effect of θ when recovery quality is high.

References

[1] 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, 2005.
[2] M. E. Tipping, “Sparse Bayesian learning and the relevance vector machine,” *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.
[3] A. C. Faul and M. E. Tipping, “Analysis of sparse bayesian learning,” in *Advances in Neural Information Processing Systems 14*, 2002, pp. 383–389.
[4] D. P. Wipf and B. D. Rao, “Sparse Bayesian learning for basis selection,” *IEEE Trans. on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, 2004.
[5] ——, “An empirical Bayesian strategy for solving the simultaneous sparse approximation problem,” *IEEE Trans. on Signal Processing*, vol. 55, no. 7, pp. 3704–3716, 2007.
[6] Z. Zhang and B. D. Rao, “Iterative reweighted algorithms for sparse signal recovery with temporally correlated source vectors,” in *Proc. of the 36th International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2011)*, Prague, the Czech Republic, 2011.
[7] ——, “Sparse signal recovery with temporally correlated source vectors using sparse bayesian learning,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 5, pp. 912–926, 2011.
[8] K. Qiu and A. Dogandzic, “Variance-component based sparse signal

reconstruction and model selection," *IEEE Trans. on Signal Processing*, vol. 58, no. 6, pp. 2935–2952, 2010.

[9] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Trans. on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, 2008.

[10] S. D. Babacan, R. Molina, and A. K. Katsaggelos, "Bayesian compressive sensing using laplace priors," *IEEE Trans. Image Processing*, vol. 19, no. 1, pp. 53–64, 2010.

[11] R. M. Neal, *Bayesian learning for neural networks*. Springer, 1996.

[12] Z. Zhang and B. D. Rao, "Exploiting correlation in sparse signal recovery problems: Multiple measurement vectors, block sparsity, and time-varying sparsity," in *ICML 2011 Workshop on Structured Sparsity: Learning and Inference*, 2011.

[13] D. Wipf and S. Nagarajan, "Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 317–329, 2010.

[14] D. Wipf, B. D. Rao, and S. Nagarajan, "Latent variable Bayesian models for promoting sparsity," *accepted by IEEE Trans. on Information Theory*, 2010.

[15] D. Wipf, J. P. Owen, H. T. Attias, and et al, "Robust Bayesian estimation of the location, orientation, and time course of multiple correlated neural sources using meg," *NeuroImage*, vol. 49, pp. 641–655, 2010.

[16] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Sci. Comput.*, vol. 20, no. 1, pp. 33–61, 1998.

[17] M. E. Tipping and A. C. Faul, "Fast marginal likelihood maximisation for sparse bayesian models," in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.