**New Bayesian Compressive Sensing Algorithm for Sparse Signal Recovery**

Mohammad Shekaramiz, Todd K. Moon, and Jacob H. Gunther

Information Dynamics Laboratory, ECE Dept., Utah State University

**Introduction**

- **Objective:** Recovering sparse signal $X$ from a small set of linear noisy measurements using either single- or multiple-measurement vectors (SMV or MMVs)

- **Assumptions:**
  - Sparse Clustered Pattern: Non-zero elements of the underlying signal may appear in clusters with an unknown structure on each column of $X$
  - Joint-Sparsity: Columns of $X$ have the same non-zero locations

- **Proposed algorithm:**
  - C-SBL: Sparse Bayesian learning model for sparse signals with unknown clustered pattern

**Proposed Bayesian Model and Defining Priors (Cndt)**

- **Prior on the noise component $\varepsilon$:**
  $\varepsilon_{mn} \sim \mathcal{N}(0, \sigma^2)$, $m = 1, \ldots, M$, $n = 1, \ldots, N$

- **Prior on the clumpiness knob $\gamma$:**
  $\gamma \sim \text{Gamma}(a, b)$

**Update Rule for $\gamma$ (EM-based)**

$$Q(\gamma | \mathbf{y}) = E_{X|\mathbf{y}} \left( \log p(Y, X | \mathbf{y}) \right)$$

where $Q(\gamma | \mathbf{y}) = (\gamma_1, \ldots, \gamma_n)$ denotes the current estimates

- **Maximization step of EM:**
  $\gamma_m^{(n+1)} = \arg \max \ E_{\mathbf{X}|\gamma^{(n)}} [Q(\gamma | \mathbf{y})]$

- **Update rule for $\gamma$:** obtained via solving:

$$p \sum_{j=1}^{L} E_{X|\mathbf{y}} \left( (\Sigma X_j)^2 \right) - \sum_{j=1}^{L} E_{X|\mathbf{y}} \left( (\Sigma X_j)^2 \right) - 2 p \sum_{j=1}^{L} E_{X|\mathbf{y}} \left( (\Sigma X_j) \right) b_n - 0$$

where $(\Sigma X_j)_b = (\Sigma X_j)_a - (\Sigma X_j)_b$

**C-SBL Algorithm**

**Proposed Bayesian Model and Defining Priors**

- **Basic MMV Model:** Solve for $X_m$ in $Y = AX_m + \varepsilon$, where $Y \in \mathbb{R}^{N \times P}$, $A \in \mathbb{R}^{M \times N}$, and $(M < N)$

- **Promoting sparsity:** Gaussian-Bernoulli prior by defining $X_m \sim s \times X$, where $s \in \{0, 1\}$

- **s accounts for the supports of the solution and $\times$ denotes Hadamard product**

- **Promoting clustering pattern:** Incorporating total variation like prior on the support learning vector $s$ using measure of clumpiness $(\Sigma X)_a - (\Sigma X)_b$

- **There exist few transitions for the clustered pattern supports**

- **Examples:**

  - $\begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$
  - $\begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix}$
  - $\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}$
  - $\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$

- **Prior on the solution-value matrix $X$:**

  - $X_{\mathcal{X}_m} \sim \mathcal{N}(0, \sigma^2 \mathbf{1})$, $X \sim \text{Gamma}(a, b)$, $n = 1, \ldots, N$

- **Prior on the support-learning component $s$:**

  - Model the elements of $s$ as Bernoulli random variables
    $s_{k, l} \sim \text{Bernoulli} \left( \frac{\sigma^2 (\gamma - M)}{\sigma^2 (\gamma - M) + \sigma^2} \right)$, $\gamma \in [0, 1]$

- **Examples of support learning (MCMC-Gibbs Sampler)**

**Simulation Results on Synthetic Data**

- **Aspects of performance of C-SBL for the SMV problem**

- **Aspects of performance of C-SBL for MMV (with $N = 2$)**

- **Comparisons of various algorithms for the SMV case**

- **Comparisons of various algorithms for the MMV ($N = 2$) case**

**Performance Comparison on Hand-Written Digits**

- **Original images (borrowed from MNIST) are scaled up to be of size 100 x 100 pixels**

- **The pixel values were normalized to be within [0, 1], then they were subtracted from 1, and those with value of less than 0.3 were deemed to zero**

- **For SMV we solve each column of $X$ (for each digit) one at the time**

- **The number of measurements for each column of $X$ is set to 55**

- **The measurements for each column of digit are computed by $y_{mn} = X_{mn} + \varepsilon_{mn}$, where $x_{mn} \in \mathbb{R}^{100}$ for $1 \ldots 100$ with SNR = 25 dB**

- **We set the estimated pixels values lower value than 0.3 to 0**

- **The true $(\Sigma X)$ for digits 0, 1, 4, 5, 6, 7, 8, 9 are 208, 206, 160, 316, and 220, respectively**

- **C-SBL: sensing the measurement performance in terms of $\text{SNR}$**

- **MFOCUSS: choosing the measurement performance in terms of $\text{SNR}$**

- **MTCS, C-SBL, TMSBL, MSBL, MFOCUS, and MFOCUSS algorithms, respectively**

- **SMV case: Columns from left to right show the true supports, support recovery via C-SBL, TMSBL, MSBL, MFOCUS, and MFOCUSS algorithms, respectively**

- **MNIST case: Columns from left to right show support reconstruction via C-SBL, TMSBL, MSBL, MFOCUS, and MFOCUSS algorithms, respectively**

- **SMV case: Looking for the measurement performance in terms of $\text{SNR}$**

- **MNIST case: Columns from left to right show support reconstruction via C-SBL, TMSBL, MSBL, MFOCUS, and MFOCUSS algorithms, respectively**