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

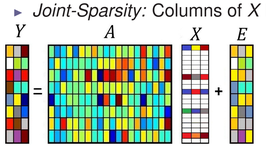
Introduction

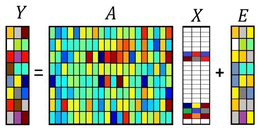
Objective:

- 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

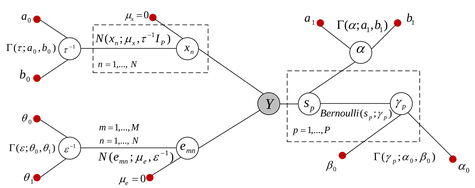




Proposed algorithm:

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

Graphical Representation of the Proposed Bayesian Model



Proposed Bayesian Model and Defining Priors

Basic MMV Model: Solve for X_{sp} in $Y = AX_{sp} + E$, where $Y \in \mathbb{R}^{M\times N}$, $A \in \mathbb{R}^{M\times P}$, $X_{sp} \in \mathbb{R}^{P\times N}$, and ($\mathbf{M} \ll \mathbf{P}$)

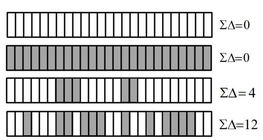
Promoting sparsity: Gaussian-Bernoulli prior by defining $X_{sp} := \mathbf{s} \circ X$, where $\mathbf{s} \in \{0,1\}^{P\times 1}$

- $\mathbf{s}$ accounts for the supports of the solution and "$\circ$" denotes Hadamard product

Promoting clustering pattern: Incorporating total variation like prior on the support learning vector $\mathbf{s}$ using measure of clumpiness $(\Sigma\Delta)_{\mathbf{s}} := \sum_{i=2}^{P} |s_i - s_{i-1}|$

- There exist few transitions for the clustered pattern supports

Examples:



- ***Prior on the solution-value matrix "X":***
  - $X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ are drawn i.i.d. from the normal-gamma distribution
  $$\mathbf{x}_n \sim \mathcal{N}(0, \tau^{-1} I_P),\ \tau \sim \text{Gamma}(a_0, b_0),\ n = 1, \ldots, N.$$
- ***Prior on the support-learning component "s":***
  - Model the elements of $\mathbf{s}$ as Bernoulli random variables
  $$(s_p; \omega_{0,p}, \omega_{1,p}) \sim \text{Bernoulli}\left(\frac{\omega_{1,p}}{\omega_{0,p} + \omega_{1,p}}\right), \forall p = 1, 2, \ldots, P$$
  $$\omega_{k,p} = e^{-\alpha(\Sigma\Delta)_{k,p}} \text{Binomial}(\Sigma_{k,p}, P, \gamma_p),\ \forall k = 0, 1$$
  - The terms $(\Sigma\Delta)_{k,p}$ and $\Sigma_{k,p}$ denote the $(\Sigma\Delta)$ value and the sum over all the elements of $\mathbf{s}$ for the case where $s_p = k$, respectively
  $$\gamma_p \sim \text{Beta}(\alpha_0, \beta_0), p = 1, \ldots, P.$$

Proposed Bayesian Model and Defining Priors (Cntd)

- ***Prior on the noise component "E":***
  $$e_{mn} \sim \mathcal{N}(0, \varepsilon^{-1}),\ m = 1, \ldots, M,\ n = 1, \ldots, N,\ \varepsilon \sim \text{Gamma}(\theta_0, \theta_1)$$
- ***Prior on the clumpiness knob "α":***
  $$\alpha \sim \text{Gamma}(a_1, b_1)$$

Update Rule for "α" (EM-based)

$$Q(\alpha|\Theta^{[t]}) = E_{X,\mathbf{s}|Y,\Theta^{[t]}}\left[\log p(Y, X, \mathbf{s}; \Theta)\right]$$

where $\Theta^{[t]} = \{\alpha^{[t]}, \gamma^{[t]}, \tau^{[t]}, \varepsilon^{[t]}\}$ denotes the current estimates

- Maximization step of *EM*:
  $$\alpha^{[t+1]} = \arg\max_{\alpha}\ E_{X,\mathbf{s}|Y,\Theta^{[t]}}\left[Q(\alpha|\Theta^{[t]})\right].$$
- Update rule for "α" is obtained via solving:
  $$\sum_{p=1}^{P} \frac{(\bar{\Sigma\Delta})_p^{[t]}}{1 + \frac{1}{c_p^{[t]}} e^{\alpha^{[t+1]}(\bar{\Sigma\Delta})_p^{[t]}}} + \sum_{p=1}^{P} (1 - s_p^{[t]})(\bar{\Sigma\Delta})_p^{[t]} + \frac{a_1 - 1}{\alpha^{[t+1]}} - b_1 = 0, \qquad (1)$$

where $(\bar{\Sigma\Delta})_p := (\Sigma\Delta)_{0,p} - (\Sigma\Delta)_{1,p}$

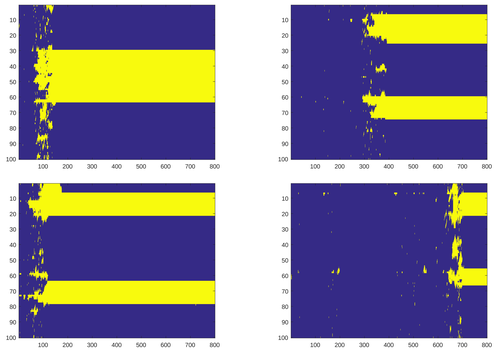
C-SBL Algorithm

C-SBL Algorithm:

$\{\Theta^{(i)}\}_{i=1 \text{ to } N_{\text{collect}}} =$ **C-SBL**$(Y, A, \Theta_0, N_{\text{burn-in}}, N_{\text{collect}})$
For Iter = 1 to $N_{\text{burn-in}} + N_{\text{collect}}$
% Support-learning vector component
For p = 1 to P
$y_{mn}^{p} = y_{mn} - \sum_{l \neq p}^{P} a_{ml} s_l x_{ln},\quad \forall m = 1 \text{ to } M, \forall n = 1 \text{ to } N$
$c_p = \frac{1-\gamma_p}{\gamma_p} \frac{\Sigma_{1,p}}{P - 1 - \Sigma_{1,p}}$
$k_p = e^{\frac{1}{2}\left(\left(\|\mathbf{a}_p\|_2^2 \sum_{n=1}^{N} x_{pn}^2\right) - 2\mathbf{a}_p^T \left(\sum_{n=1}^{N} x_{pn} \mathbf{y}_n^p\right)\right)}$
$(s_p|-) \sim \text{Bernoulli}\left(\frac{1}{1 + c_p k_p e^{-\alpha\left((\Sigma\Delta)_{0,p} - (\Sigma\Delta)_{1,p}\right)}}\right)$
% Solution-value matrix component
For l = 1 to P
$\Sigma_x = \left(\tau + \varepsilon s_l^2 \|\mathbf{a}_l\|_2^2\right)^{-1}$
$\bar{\mu} = \varepsilon s_l \Sigma_x \mathbf{a}_l$
$\mathbf{y}_n^{l} = \mathbf{y}_n - A(\mathbf{s} \circ \mathbf{x}_n) + s_l x_{l,n} \mathbf{a}_l,\ \forall n = 1, \ldots, N$
$(x_{l,n}|-) \sim \mathcal{N}(\bar{\mu}^T \mathbf{y}_n^{l}, \Sigma_x),\ \forall n = 1, \ldots, N$
End For $\{l\}$
$(\gamma_p|-) \sim \text{Beta}\left(\alpha_0 + 1 + 2\sum_{k \neq p}^{P} s_k,\ \beta_0 - 1 + 2(P - \sum_{k \neq p}^{P} s_k)\right)$
End For $\{p\}$
$(\tau|-) \sim \text{Gamma}(a_0 + \frac{NP}{2}, b_0 + \frac{1}{2}\|X\|_F^2)$
$(\varepsilon|-) \sim \text{Gamma}(\theta_0 + \frac{MP}{2}, \theta_1 + \frac{1}{2}\|Y - A(\mathbf{s} \circ X)\|_F^2)$
α : obtained from solving (1) for $\alpha^{[t+1]}$
$\Theta^{(\text{Iter} - N_{\text{burn-in}})} \leftarrow \Theta,\quad \forall \text{Iter} > N_{\text{burn-in}}$
End For {Iter}

MCMC inference for implementation of C-SBL using Gibbs sampling

Examples of support learning (MCMC-Gibbs Sampler)

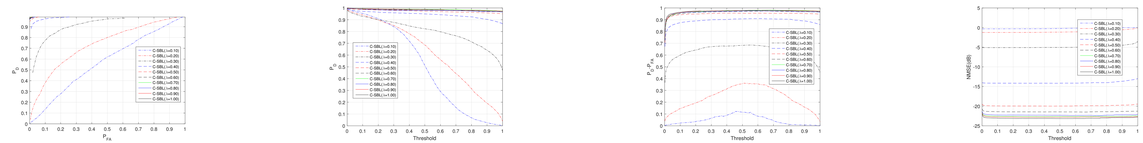


Simulation Results on *Synthetic Data*

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

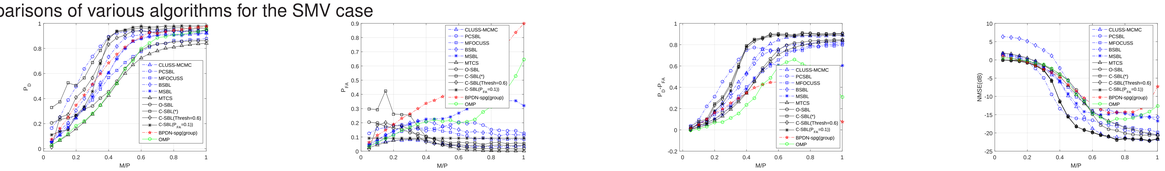
(a) Empirical ROC. (b) Detection rate P_D. (c) Support recovery $P_D - P_{FA}$. (d) NMSE (dB).

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



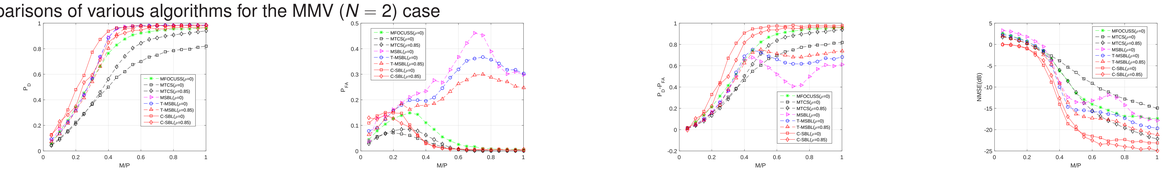
(a) Empirical ROC. (b) Detection rate P_D. (c) Support recovery $P_D - P_{FA}$. (d) NMSE (dB).

- Comparisons of various algorithms for the SMV case



(a) Detection rate P_D (b) False alarm P_{FA} (c) Support recovery $P_D - P_{FA}$ (d) NMSE (dB)

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



(a) Detection rate P_D. (b) False alarm rate P_{FA}. (c) Support recovery $P_D - P_{FA}$. (d) NMSE (dB).

Performance Comparison on Hand-Written Digits

- Original images (borrowed from MNIST) are scaled up to be of size 100×100 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 each digit are computed by $\mathbf{y}_n = A\mathbf{x}_n + \mathbf{e}_n$, where $\mathbf{x}_n \in \mathbb{R}^{100}, \forall n = 1, \ldots, 100$ with SNR=25 dB
- We set the estimated pixels values lower value than 0.3 to 0
- The true $(\Sigma\Delta)$ for digits **0,1,4,5,6** are **208,80,166,316,** and **220**, respectively

- SMV case: Comparing the reconstruction performance in terms of $r_D - r_{FA}$

Algorithm	Digit 0	Digit 1	Digit 4	Digit 5	Digit 6
C-SBL	0.9530	0.9954	0.9834	0.9690	0.9847
BSBL	0.9204	0.9819	0.9617	0.9479	0.9116
PCSBL	0.7622	0.9544	0.8961	0.8270	0.8468
CLUSS	0.4265	0.6689	0.6421	0.5878	0.7179
MTCS	0.3030	0.4828	0.4779	0.4074	0.5989
MFOCUSS	0.3652	0.6510	0.5197	0.4961	0.5734
BPDN	0.3701	0.6360	0.5212	0.4967	0.5859

- SMV case: Learning the clustering pattern via our measure of clumpinesss $(\Sigma\Delta)$

Algorithm	Digit 0	Digit 1	Digit 4	Digit 5	Digit 6
C-SBL	244	78	176	324	196
BSBL	320	84	200	344	262
PCSBL	634	126	346	632	344
CLUSS	1018	370	570	804	446
MTCS	1774	1022	1194	1496	760
MFOCUSS	1532	878	936	1220	684
BPDN	1478	818	908	1192	646

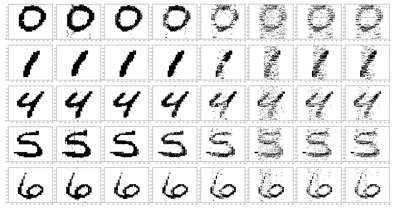
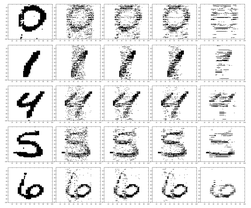
- MMV case: Comparing the reconstruction performance in terms of $r_D - r_{FA}$

Algorithm	Digit 0	Digit 1	Digit 4	Digit 5	Digit 6
C-SBL	0.9174	0.9822	0.9827	0.9762	0.9528
TMSBL	0.3439	0.5571	0.5218	0.4431	0.6147
MSBL	0.3688	0.6241	0.5515	0.4913	0.6497
MTCS	0.1776	0.2642	0.3192	0.3730	0.3795
MFOCUSS	0.3778	0.6517	0.5490	0.5282	0.6202

- MMV case: Learning the clustering pattern via our measure of clumpinesss $(\Sigma\Delta)$

Algorithm	Digit 0	Digit 1	Digit 4	Digit 5	Digit 6
C-SBL	274	80	176	324	200
TMSBL	1810	1054	1166	1568	752
MSBL	1630	796	1034	1360	650
MTCS	1228	688	834	1022	576
MFOCUSS	1522	806	936	1210	636

- SMV case: Columns from left to right show the true supports, support reconstruction via C-SBL, BSBL (h=4), PCSBL, CLUSSMCMC, MTCS, MFOCUSS, and BPDN algorithms, respectively



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