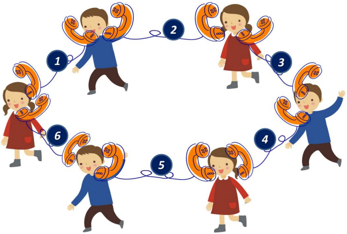


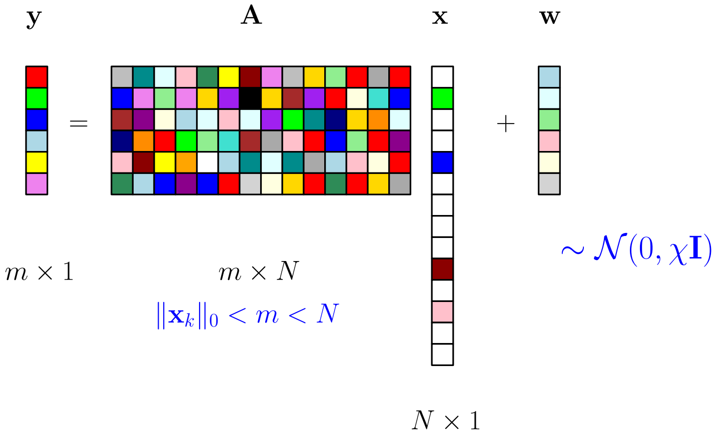
Probabilistic Sparse Recovery: Approximate Message Passing Algorithm

Geethu Joseph

SPC Lab, IISc

June 4, 2016

Sparse Signal Recover Problem



Goal: recover the signal ***x*** from observation ***y***

Sparse Recovery

Geometric[1]
LASSO, l_p norm-based

Combinatorial[2]
IHT, CoSaMP , SP, OMP

Probabilistic[3]
AMP, SBL

[1] [Donoho '04],[Candes-Tao '04, '06], [Rudelson-Vershynin'06]
[2] [Needell et al. '08] [Rubinstein et al. '09]
[3] [Zhou et al. '09] [Donoho et al. '09]

Probabilistic Approach

1. Impose a prior distribution on $\boldsymbol{x}$
2. Noise distribution gives $p(\boldsymbol{y}|\boldsymbol{x})$
3. Find the MAP/MMSE estimate using conditional pdf

$$\hat{\boldsymbol{x}}_{\mathrm{MAP}} = \max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y})$$
$$\hat{\boldsymbol{x}}_{\mathrm{MMSE}} = \mathbb{E}\{\boldsymbol{x}|\boldsymbol{y}\}$$

Motivation

- Impose Laplacian prior on $\boldsymbol{x}$, x_i are iid

$$p(\boldsymbol{x}) \cong \exp\left\{\frac{\lambda}{\chi}\|\boldsymbol{x}\|_1\right\}$$

- Noise distribution

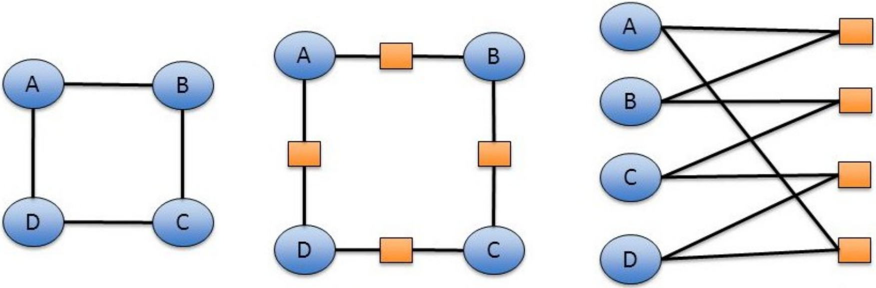
$$p(\boldsymbol{y}|\boldsymbol{x}) \;\cong\; \exp\left\{-\frac{1}{2\chi}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2\right\}$$

- Laplacian MAP = LASSO

$$\begin{aligned}\hat{\boldsymbol{x}} &= \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y})\\ &= \arg\min_{\boldsymbol{x}} -\log p(\boldsymbol{x},\boldsymbol{y})\\ &= \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2 + \lambda\|\boldsymbol{x}\|_1\end{aligned}$$

Factor Graphs

- Joint probability of many variables factors into little pieces
- Nodes: random variables, factors
- No normalization: determine all probabilities



MAP/MMSE using factor graphs

- Construction of bipartite graph
  - Variable nodes: $x_1, x_2, \ldots, x_N$
  - Factor nodes: $p(y_1|\boldsymbol{x}), p(y_2|\boldsymbol{x}), \ldots p(y_m|\boldsymbol{x})$
  - An edge between a variable node i and a factor node a if $A_{ai} \neq 0$
- **Goal:** Obtain marginal $p(x_i|\boldsymbol{y})$ or MAP estimate, $\arg\max_{\boldsymbol{x}} = p(\boldsymbol{x}|\boldsymbol{y})$

Solution: Belief propagation

Belief Propagation

- **Intuition**: Peer pressure - A node determines a final belief distribution by listening to its neighbors.
- Keep passing messages (functions of the variable) until a stable belief state is reached
- Two versions
  - Sum-product: to compute marginals (MMSE), $p(\boldsymbol{x})$
  - Max-product: to compute maximizer (MAP), $\arg\max_{\boldsymbol{x}} p(\boldsymbol{x})$

Message Passing

- From a variable node to factor node

$$m_{i\to a} = p(x_i) \prod_{b\in\partial i\setminus a} m_{b\to i}(x_i)$$

- From factor node to variable node:
  1. **Sum-product**: estimate marginals

$$m_{a\to i} = \int_{\boldsymbol{x}_{-i}} p(y_a|\boldsymbol{x}) \prod_{j\in\partial a\setminus i} m_{j\to a}(x_j) d\boldsymbol{x}_{-i}$$

     - ★ **Marginal**: $p(x_i|\boldsymbol{y}) = p(x_i)\prod_{b\in\partial i} m_{b\to i}(x_i)$

  2. **Max-product**: scoring functions whose maxima are most likely states

$$m_{a\to i} = \max_{\boldsymbol{x}_{-i}} p(y_a|\boldsymbol{x}) \prod_{j\in\partial a\setminus i} m_{j\to a}(x_j)$$

     - ★ **Maximum**: $\hat{x}_i = \arg\max_{x_i} p(x_i)\prod_{b\in\partial i} m_{b\to i}(x_i)$

AMP Algorithms

- AMP: Belief propagation with carefully constructed approximations
  - The original AMP[4]: Laplacian MAP, with *iid* Gaussian noise, and *iid* matrix $\boldsymbol{A}$
  - The Bayesian AMP
  - The generalized AMP
  - Expectation-Maximization Bernoulli-Gaussian AMP
- Very fast: $\approx$ 15 iterations, simple operations

[4] [Donoho, Maleki, Montanari '09]

Design: Laplacian MAP estimate

1. Construct a joint distribution on data and write down the corresponding sum-product algorithm
2. Approximate the sum-product messages by the families with two scalar parameters
3. Use law of large numbers to approximate messages for the large system limit

Step 1: Basic Messages

- Recall: With $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$,

$$p(\boldsymbol{x}) \cong \prod_{i=1}^{N} \exp\{-\sigma^{-2}\lambda|x_i|\}$$

$$p(\boldsymbol{y}|\boldsymbol{x}) \cong \prod_{a=1}^{m} \exp\{-\frac{1}{2\chi}\|y_a - z_a\|\}$$

- Messages in negative log domain are

$$m_{i\to a}^{(t+1)}(x_i) = \lambda|x_i| + \sum_{b\in[m]\setminus a} m_{b\to i}^{(t)}(x_i)$$

$$m_{a\to i}^{(t)}(x_i) = \min_{x_{j\in[N]\setminus i}} \frac{1}{2}(y_a - z_a)^2 + \sum_{j\in[N]\setminus i} m_{j\to a}^{(t)}(x_j)$$

- Each iteration: $2Nm$ functions on real line

Assumptions

- $x_i \sim o(1)$
- $A_{ai}, iid \sim O(1/\sqrt{m})$
- $\boldsymbol{A}$ is normalized
  - Column sum: $\sum_{a=1}^{m} A_{ai} = 0$
  - l_2 norm of column $\sum_{a=1}^{m} A_{ai}^2 = 1$
- m scales with N

Step 2: Approximation to Family of 2 Scalars

- Approximation to quadratic function
  - Expand the messages using Taylor series, around 0

$$\begin{aligned} m_{a\to i}^{(t)}(x_i) &= -\alpha_{a\to i}^{(t)} A_{ai} x_i + \beta_{a\to i}^{(t)} (A_{ai} x_i)^2 \\ m_{i\to a}^{(t+1)}(x_i) &= \frac{1}{2\gamma_{i\to a}^{(t+1)}} \left(x_i - x_{i\to a}^{(t+1)} \right)^2 \end{aligned}$$

- Each iteration: $2Nm$ real numbers functions of $\{\boldsymbol{y}, \boldsymbol{A}\}$

$$m_{i\to a}^{(t+1)}(x_i) \to \begin{bmatrix} x_{i\to a}^{(t+1)} & \gamma_{i\to a} \end{bmatrix}$$

$$m_{a\to i}^{(t)}(x_i) \to \begin{bmatrix} \alpha_{a\to i}^{(t)} & \beta_{a\to i}^{(t)} \end{bmatrix}$$

- Memory requirements $\sim O(mN)$

Further Simplification!

- Relation between messages:

$$r_{a\to i}^{(t)} = \frac{\alpha_{a\to i}^{(t)}}{\beta_{a\to i}^{(t)}} = y_a - \sum_{j\in[N]\setminus i} A_{aj} x_{j\to a}^{(t)}$$

$$x_{i\to a}^{(t+1)} = \eta\left(\sum_{b\in[m]\setminus a} A_{bi} r_{b\to i}^{(t)}; \theta_t\right)$$

- Law of large numbers: $\frac{\lambda}{\sum_{b\in[m]\setminus a} \beta_{b\to i}^{(t)} A_{bi}^2} \approx \theta_t$
- $\eta(a;\theta)$ is soft thresholding function

$$\eta(a;\theta) = \begin{cases} a-\theta & a>\theta \\ 0 & -\theta\le a\le\theta \\ a+\theta & a<\theta \end{cases} \tag{1}$$

Step 3: Large System Limit

- Approximation:

$$r_{a\to i}^{(t)} \approx r_a^{(t)} + \delta r_{a\to i}^{(t)} + \mathcal{O}(1/N)$$
$$x_{i\to a}^{(t)} \approx x_i^{(t)} + \delta x_{i\to a}^{(t)} + \mathcal{O}(1/N)$$

- Approximation steps involve
  1. Taylor series expansion of $x_{i\to a}^{(t+1)}$
  2. Law of large numbers
- Memory requirements $\sim O(m+N)$
- On convergence, declare estimate as $x_i^{(t)}$

AMP Algorithm

- After simplifications,

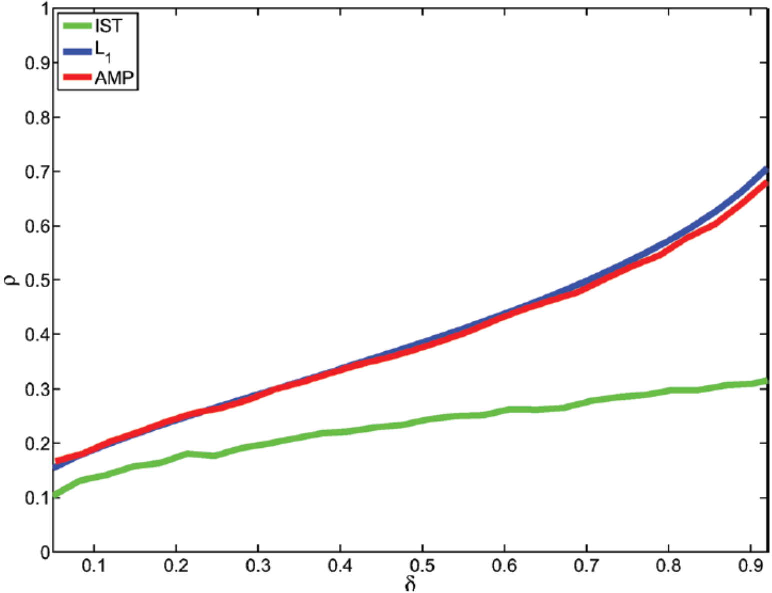
$$\boxed{\begin{aligned} \boldsymbol{x}^{t+1} &= \eta\left(\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^{(t)}; \theta_t\right) \\ \boldsymbol{r}^t &= \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} + b_t\boldsymbol{r}^{(t-1)} \end{aligned}}$$

where $b_t \approx \frac{1}{m}\sum_{i=1}^{N}\eta'(\boldsymbol{x}_i^{(t-1)} + [\boldsymbol{A}^T\boldsymbol{r}^{(t-1)}]_i; \theta_{t-1})$

- Similar to iterative soft thresholding:

$$\begin{aligned} \boldsymbol{x}^{t+1} &= \eta\left(\boldsymbol{x}^t + \boldsymbol{A}^T\boldsymbol{r}^{(t)}; \theta_t\right) \\ \boldsymbol{r}^t &= \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} \end{aligned}$$

Phase Transition Curve



Covergence

Theorem

[a] *For Laplacian prior, and* $\boldsymbol{A}$ *is Gaussian with iid entries, then (for n large enough) AMP converge within relative distance* ϵ *from a minimizer in* $t = O(log(1/\epsilon))$ *iterations.*

[a] [Bayati, Montanari '11]