Message passing and approximate message passing

Arian Maleki

Columbia University

What is the problem?

Given pdf $\mu(x_1, x_2, \ldots, x_n)$ we are interested in

- $\arg\max_{x_1, x_2, \ldots, x_n} \mu(x_1, x_2, \ldots, x_n)$
- Calculating marginal pdf $\mu_i(x_i)$
- Calculating joint distribution $\mu_S(x_S)$ for $S \subset \{1, 2, \ldots, n\}$

These problems are important because

- Many inference problems are in these forms
- Many applications in image procession, communication, machine learning, signal processing

This talk: Marginalizing distribution

- Most tools are applied to other problems

This talk: marginalization

Problem: Given pdf $\mu(x_1, x_2, \ldots, x_n)$ find $\mu_i(x_i)$

- Simplification: all x_j's are binary random variables

Is it difficult?

- Yes. No polynomial time algorithm
- Complexity of variable elimination: 2^{n-1}

Solution:

- No "good solution" for "generic μ"
- Consider structured μ

Structured μ

Problem: Given pdf $\mu(x_1, x_2, \ldots, x_n)$ find $\mu_i(x_i)$

- Simplification: all x_j's are binary random variables

Example 1: $x_1, x_2, \ldots, x_n$ are independent

- Easy marginalization
- Not interesting

Example 2: Independent subsets

$$\mu(x_1, x_2, \ldots, x_n) = \mu_1'(x_1, \ldots, x_k)\mu_2'(x_{k+1}, \ldots, x_n)$$

- More interesting and less easy
- Complexity still exponential in terms of size of subsets
- Does not cover many interesting cases

Structured μ:Cont'd

Problem: Given pdf $\mu(x_1, x_2, \ldots, x_n)$ find $\mu_i(x_i)$

- Simplification: all x_j's are binary random variables

Example 3: more interesting structure:

$$\mu(x_1, x_2, \ldots, x_n) = \mu'_1(x_{S_1})\mu'_2(x_{S_2}) \ldots \mu'_\ell(x_{S_\ell})$$

- Unlike independence, $S_i \cap S_j \neq \emptyset$
- Covers many interesting practical problems
- Let $|S_i| \ll n, \forall i$. Is marginalization easier than a generic μ ?
  - Not clear!! We introduce factor graph to address this question

Factor graph

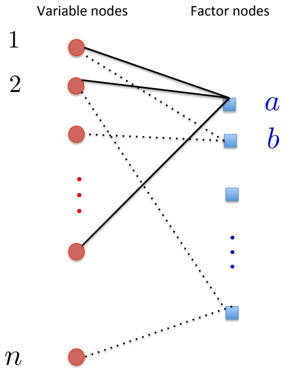
Slight generalization

$$\mu(x_1, x_2, \ldots, x_n) = \frac{1}{Z} \prod_{a \in F} \psi_a(x_{S_a})$$

- $\psi_a : \{0,1\}^{|S_a|} \to \mathbb{R}^+$ not necessarily pdf
- Z: normalizing constant. Called "partition function"
- $\psi_a(x_{S_a})$: factors

Factor graph:

- Variable node: n nodes corresponding to $x_1, x_2, \ldots, x_n$
- Function node: $|F|$ nodes corresponding to $\psi_a(x_{S_1}), \psi_b(x_{S_2}), \ldots$
- Edge between a and variable node i iff $x_i \in S_a$



Factor graph Cont'd

$$\mu(x_1, x_2, \ldots, x_n) = \frac{1}{Z} \prod_{a \in F} \psi_a(x_{S_a})$$

Factor graph:

- Variable node: n nodes corresponding to $x_1, x_2, \ldots, x_n$
- Function node: F nodes corresponding to $\psi_a(x_{S_1}), \psi_b(x_{S_2}), \ldots$
- Edge between a and variable node i iff $x_i \in S_a$

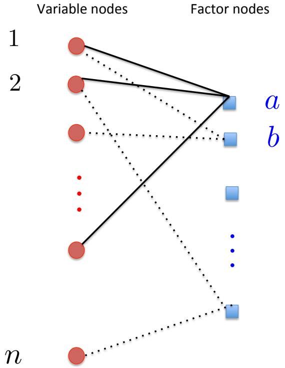
Some notation:

- ∂a: neighbors of function node a:

$$\partial a = \{i : x_i \in S_a\}.$$

- ∂i: neighbors of variable node i

$$\partial i = \{b \in F : i \in S_b\}.$$



Factor graph example

$$\mu(x_1, x_2, x_3) = \frac{1}{Z}\psi_a(x_1, x_2)\psi_b(x_2, x_3)$$

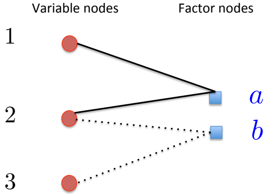
Some notation:

- ∂a: neighbors of function node a:

$$\begin{aligned} \partial a &= \{1, 2\} \\ \partial b &= \{2, 3\} \end{aligned}$$

- ∂i: neighbors of variable node i

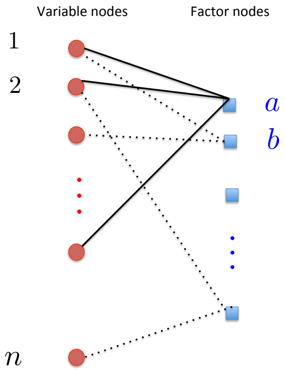
$$\begin{aligned} \partial 1 &= \{a\} \\ \partial 2 &= \{a, b\} \end{aligned}$$



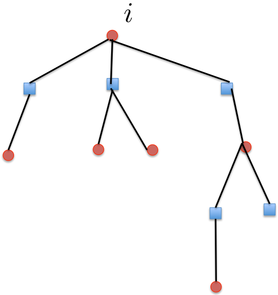
Why factor graph?

Factor graphs simplifies understanding structures

- Independence: graph has disconnected pieces
- Tree graphs: Easy marginalization
- Loops (specially short loops): generally make the problem more complicated



Marginalization on tree structured graphs



Tree: Graph with no loops

- Marginalization by variable elimination is efficient on such graphs (distributions)
  - Linear in n
  - Exponential in size of factors
- Proof: on the board

Summary of variable elimination on trees

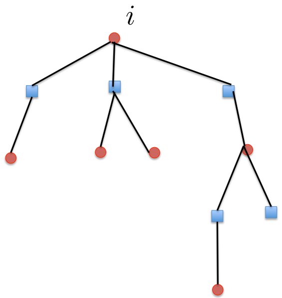
Notation

- $\mu(x_1, x_2, \ldots, x_n) = \prod_{a \in F} \psi_a(x_{\partial a})$
- $\mu_{j \to a}$: belief from variable node j to factor a
- $\hat{\mu}_{a \to j}$: belief from factor node a to variable j

We also have

$$\mu_{a \to j}(x_j) = \sum_{\{x_j : j \in \partial a \setminus j\}} \psi_a(x_{\partial a}) \prod_{\ell \in \partial a \setminus j} \hat{\mu}_{\ell \to a}(x_\ell)$$

$$\hat{\mu}_{j \to a}(x_j) = \prod_{b \in \partial j \setminus a} \mu_{b \to j}(x_j)$$

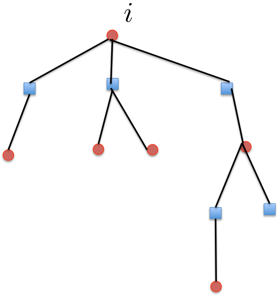


Belief propagation algorithm on trees

Based on variable elimination on trees consider iterative algorithm:

$$\nu_{a\to j}^{t+1}(x_j) = \sum_{\{x_j : j\in\partial a\setminus j\}} \psi_a(x_{\partial a}) \prod_{\ell\in\partial a\setminus j} \hat{\nu}_{\ell\to a}^{t}(x_\ell)$$

$$\hat{\nu}_{j\to a}^{t+1}(x_j) = \prod_{b\in\partial j\setminus a} \nu_{b\to j}^{t}(x_j)$$



- $\nu_{a\to j}^{t}$ and $\hat{\nu}_{j\to a}^{t}$ are beliefs of nodes at time t
- Each node propagates its belief to other nodes

Hope: Converge to a "good" fixed point

$$\nu_{b\to j}^{t}(x_j) \underset{t\to\infty}{\to} \mu_{b\to j}(x_j)$$

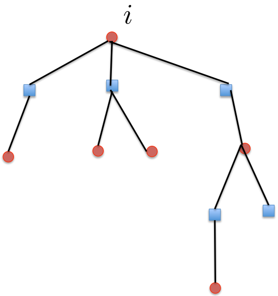
$$\hat{\nu}_{j\to b}^{t}(x_j) \underset{t\to\infty}{\to} \mu_{j\to b}(x_j)$$

Belief propagation algorithm on trees

Belief propagation:

$$\nu_{a\to j}^{t+1}(x_j) = \sum_{\{x_j : j\in\partial a\setminus j\}} \psi_a(x_{\partial a}) \prod_{\ell\in\partial a\setminus j} \hat{\nu}_{\ell\to a}^{t}(x_\ell)$$

$$\hat{\nu}_{j\to a}^{t+1}(x_j) = \prod_{b\in\partial j\setminus a} \nu_{b\to j}^{t}(x_j)$$



Why BP and not variable elimination?

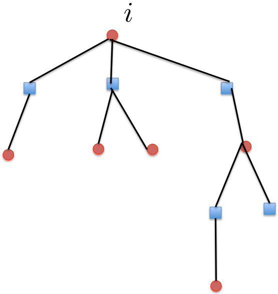
- On trees: no good reason
- loopy graphs: can be easily applied

Belief propagation algorithm on trees

Belief propagation:

$$\nu_{a\to j}^{t+1}(x_j) = \sum_{\{x_j : j\in\partial a\setminus j\}} \psi_a(x_{\partial a}) \prod_{\ell\in\partial a\setminus j} \hat{\nu}_{\ell\to a}^{t}(x_\ell)$$

$$\hat{\nu}_{j\to a}^{t+1}(x_j) = \prod_{b\in\partial j\setminus a} \nu_{b\to j}^{t}(x_j)$$



Lemma: On a tree graph BP converges to the marginal distributions in finite number of iterations, i.e,

$$\nu_{b\to j}^{t}(x_j) \underset{t\to\infty}{\to} \mu_{b\to j}(x_j)$$

$$\hat{\nu}_{j\to b}^{t}(x_j) \underset{t\to\infty}{\to} \mu_{j\to b}(x_j)$$

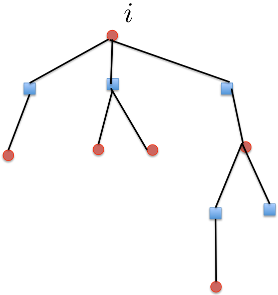
Summary of belief propagation on trees

Equivalent to variable elimination

Exact on trees

Converges in finite number of iterations:

- 2 times the maximum depth of the tree.

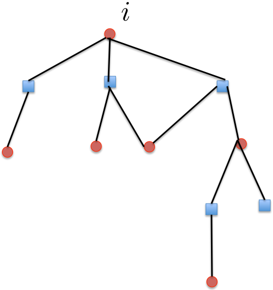


Loopy belief propagation

Apply belief propagation to loopy graph:

$$\nu_{a\to j}^{t+1}(x_j) = \sum_{\{x_j : j\in\partial a\setminus j\}} \psi_a(x_{\partial a}) \prod_{\ell\in\partial a\setminus j} \hat{\nu}_{\ell\to a}^{t}(x_\ell)$$

$$\hat{\nu}_{j\to a}^{t+1}(x_j) = \prod_{b\in\partial j\setminus a} \nu_{b\to j}^{t}(x_j)$$



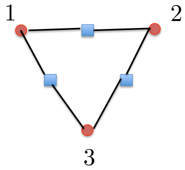
Wait until the algorithm converges:

- Announce $\prod_{a\in\partial j} \nu_{a\to j}^{\infty}(x_j)$ as $\mu_j(x_j)$

Challenges for loopy BP

Does not converge necessarily

- Example :
  $\mu(x_1, x_2, x_3) = \mathbb{I}(x_1 \neq x_2)\mathbb{I}(x_2 \neq x_3)\mathbb{I}(x_3 \neq x_1)$
  - $t = 1$ start with $x_1 = 0$
  - $t = 2 \Rightarrow x_2 = 1$
  - $t = 3 \Rightarrow x_3 = 0$
  - $t = 4 \Rightarrow x_4 = 1$



Even if converges, not necessarily the marginal distribution

Can have more than one fixed point

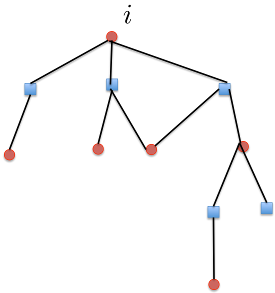
Advantages of loopy BP

In many applications, works really well

- Coding theory
- Machine vision
- compressed sensing

  ⋮

In some cases, loopy BP can be analyzed theoretically



Some theoretical results on loopy BP

BP equations have at least one fixed point

- Proof uses Brouwer's theorem
- Does not necessarily converge to any of the fixed points

BP is exact on trees

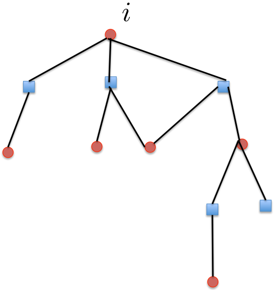
BP is "accurate" for "locally tree like" graphs

Gallager 1963, Luby et al. 2001, Richardson and Urbanke 2001

Several methods exist for verifying correctness of BP

- Computation trees
- Dobrushin condition

  ⋮



Locally tree like graphs

BP is "accurate"for "locally tree like" graphs

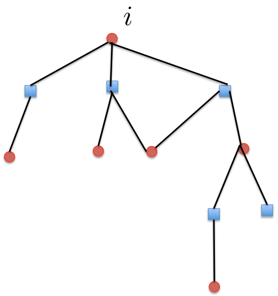
Heuristic: sparse graphs (without many edges) can be "locally tree like"

How to check?

- If $\mathrm{girth}(G) = O(\log(n))$

  $\mathrm{girth}(G)$ is the length of the shortest cycle

- Random (d_v, d_c) graph is locally tree like with high probability



Dobrushin condition

sufficient condition for convergence of BP

Influence of node j on node i:

$$C_{ij} = \sup_{\underline{x},\underline{x'}} \{ \| \mu(x_i \mid x_{V \setminus i,j}) - \mu(x_i \mid x'_{V \setminus i,j}) \|_{TV} : x_l = x'_l \ \forall l \neq j \}$$

Intuition: if the influence of the nodes on each other is very small, then oscillations should not occur

Theorem

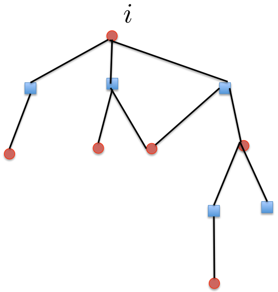
If $\gamma = \sup_i (\sum_j C_{ij}) < 1$, then BP marginals converge to the true marginals.

Summary of BP

Belief propagation:

$$\nu_{a\to j}^{t+1}(x_j) = \sum_{\{x_j : j\in\partial a\setminus j\}} \psi_a(x_{\partial a}) \prod_{\ell\in\partial a\setminus j} \hat{\nu}_{\ell\to a}^{t}(x_\ell)$$

$$\hat{\nu}_{j\to a}^{t+1}(x_j) = \prod_{b\in\partial j\setminus a} \nu_{b\to j}^{t}(x_j)$$



Exact on trees

Accurate on locally tree like graphs

No generic proving technique that works on all problems

Successful in many applications, even when we cannot prove convergence

More challenges

What if μ is a pdf on $\mathbb{R}^n$?

Extension seems straightforward

$$\begin{aligned}
\nu_{a\to j}^{t+1}(x_j) &= \int_{\{x_j : j \in \partial a \backslash j\}} \psi_a(x_{\partial a}) \prod_{\ell \in \partial a \backslash j} \hat{\nu}_{\ell \to a}^t(x_\ell) d\underline{x} \\
\hat{\nu}_{j\to a}^{t+1}(x_j) &= \prod_{b \in \partial j \backslash a} \nu_{b \to j}^t(x_j)
\end{aligned}$$

But, calculations become complicated

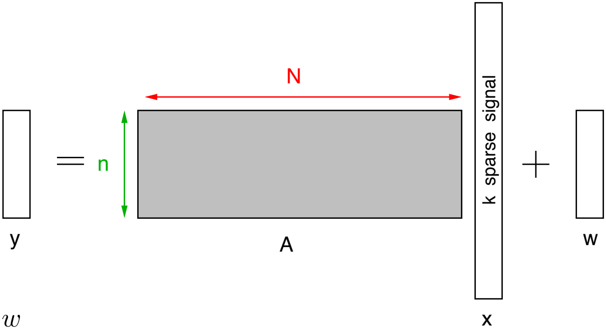
- Shall discretize the pdf with certain accuracy
- Do lots of numeric multiple integrtions

Exception: Gaussian graphical models

- $\mu(\underline{x}) = \frac{\exp(-\underline{x}^t J \underline{x} + c^t \underline{x})}{Z}$
- All messages become Gaussian
- NOT the focus of this talk

Approximate message passing

Model



$$y = Ax_o + w$$

- x_o: k-sparse vector in $\mathbb{R}^N$
- A: $n \times N$ design matrix
- y: measurement vector in $\mathbb{R}^n$
- w: measurement noise in $\mathbb{R}^n$

LASSO

Many useful heuristic algorithms:

- Noiseless: ℓ_1-minimization

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \|x\|_1 \\ \text{subject to} \quad & y = Ax \end{aligned}$$

$\|x\|_1 = \sum_i |x_i|$

- Noisy: LASSO

$$\underset{x}{\text{minimize}} \quad \frac{1}{2}\|y - Ax\|_2^2 + \lambda\|x\|_1$$

convex optimizations

Chen, Donoho, Saunders (96), Tibshirani (96)

LASSO

Many useful heuristic algorithms:

- Noiseless: ℓ_1-minimization

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \|x\|_1 \\ \text{subject to} \quad & y = Ax \end{aligned}$$

$\|x\|_1 = \sum_i |x_i|$

- Noisy: LASSO

$$\underset{x}{\text{minimize}} \quad \frac{1}{2}\|y - Ax\|_2^2 + \lambda\|x\|_1$$

convex optimizations

Chen, Donoho, Saunders (96), Tibshirani (96)

Algorithmic challenges of sparse recovery

Use convex optimization tools to solve LASSO

Computational complexity:

- interior point method:
  - generic tools
  - appropriate for $N < 5000$
- homotopy methods:
  - use the structure of the LASSO
  - appropriate for $N < 50000$
- First order methods
  - Low computational complexity per iteration
  - Require many iterations

Algorithmic challenges of sparse recovery

Use convex optimization tools to solve LASSO

Computational complexity:

- interior point method:
  - generic tools
  - appropriate for $N < 5000$
- homotopy methods:
  - use the structure of the LASSO
  - appropriate for $N < 50000$
- First order methods
  - Low computational complexity per iteration
  - Require many iterations

Algorithmic challenges of sparse recovery

Use convex optimization tools to solve LASSO

Computational complexity:

- interior point method:
  - generic tools
  - appropriate for $N < 5000$
- homotopy methods:
  - use the structure of the LASSO
  - appropriate for $N < 50000$
- First order methods
  - Low computational complexity per iteration
  - Require many iterations

Message passing for LASSO

Define:

$$\mu(\mathrm{d}x) = \frac{1}{Z} e^{-\beta\lambda\|x\|_1 - \frac{\beta}{2}\|y-Ax\|^2} \mathrm{d}x$$

- Z: normalization constant
- $\beta > 0$

As $\beta \to \infty$:

- μ concentrates around solution of LASSO: $\hat{x}^\lambda$

Marginalize μ to obtain $\hat{x}^\lambda$

Message passing for LASSO

Define:

$$\mu(\mathrm{d}x) = \frac{1}{Z} e^{-\beta\lambda\|x\|_1 - \frac{\beta}{2}\|y - Ax\|^2} \mathrm{d}x$$

- Z: normalization constant
- $\beta > 0$

As $\beta \to \infty$:

- μ concentrates around solution of LASSO: $\hat{x}^\lambda$

Marginalize μ to obtain $\hat{x}^\lambda$

Message passing for LASSO

Define:

$$\mu(\mathrm{d}x) = \frac{1}{Z} e^{-\beta\lambda\|x\|_1 - \frac{\beta}{2}\|y - Ax\|^2} \mathrm{d}x$$

- Z: normalization constant
- $\beta > 0$

As $\beta \to \infty$:

- μ concentrates around solution of LASSO: $\hat{x}^\lambda$

Marginalize μ to obtain $\hat{x}^\lambda$

Message passing algorithm

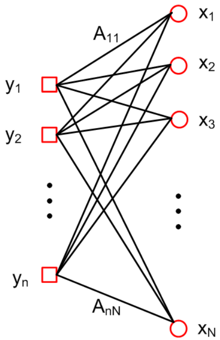
$$\mu(\mathrm{d}x) = \frac{1}{Z} e^{-\beta\lambda \sum_i |x_i| - \frac{\beta}{2} \sum_a (y_a - (Ax)_a)^2} \mathrm{d}x$$

Belief propagation update rules:

$$\nu^{t+1}_{i\to a}(s_i) \cong e^{-\beta\lambda |s_i|} \prod_{b\neq a} \hat{\nu}^t_{b\to i}(s_i)$$

$$\hat{\nu}^t_{a\to i}(s_i) \cong \int e^{-\frac{\beta}{2}(y_a - (As)_a)^2} \prod_{j\neq i} \mathrm{d}\nu^t_{j\to a}(s_j)$$

Pearl (82)



Issues of message passing algorithm

$$\nu_{i\to a}^{t+1}(s_i) \cong e^{-\beta\lambda|s_i|} \prod_{b\neq a} \hat{\nu}_{b\to i}^{t}(s_i)$$

$$\hat{\nu}_{a\to i}^{t}(s_i) \cong \int e^{-\frac{\beta}{2}(y_a-(As)_a)^2} \prod_{j\neq i} \mathrm{d}\nu_{j\to a}^{t}(s_j)$$

Challenges:

- messages are distributions over real line
- calculation of messages is difficult
  - Numeric integration
  - The number of variables in the integral is $N-1$
- The graph is not "tree like"
  - No good analysis tools

Challenges for message passing algorithm

$$\nu_{i\to a}^{t+1}(s_i) \cong e^{-\beta\lambda|s_i|} \prod_{b\neq a} \nu_{b\to i}^{t}(s_i)$$

$$\hat{\nu}_{a\to i}^{t}(s_i) \cong \int e^{-\frac{\beta}{2}(y_a-(As)_a)^2} \prod_{j\neq i} \mathrm{d}\nu_{j\to a}^{t}(s_j)$$

Challenges:

- messages are distributions over real line
- calculation of messages is difficult
- The graph is not "tree like"

Solution:

- blessing of dimensionality:
  - as $N \to \infty$: $\hat{\nu}_{a\to i}^{t}(s_i)$ converges to Gaussian
  - as $N \to \infty$: $\nu_{i\to a}^{t}(s_i)$ converges to

$$f_\beta(s;x,b) \equiv \frac{1}{z_\beta(x,b)}\, e^{-\beta|s|-\frac{\beta}{2b}(s-x)^2}\,.$$

Challenges for message passing algorithm

$$\nu^{t+1}_{i\to a}(s_i) \cong e^{-\beta\lambda|s_i|} \prod_{b\neq a} \nu^t_{b\to i}(s_i)$$

$$\hat{\nu}^t_{a\to i}(s_i) \cong \int e^{-\frac{\beta}{2}(y_a-(As)_a)^2} \prod_{j\neq i} \mathrm{d}\nu^t_{j\to a}(s_j)$$

Challenges:

- messages are distributions over real line
- calculation of messages is difficult
- The graph is not "tree like"

Solution:

- blessing of dimensionality:
  - as $N \to \infty$: $\hat{\nu}^t_{a\to i}(s_i)$ converges to Gaussian
  - as $N \to \infty$: $\nu^t_{i\to a}(s_i)$ converges to

$$f_\beta(s;x,b) \equiv \frac{1}{z_\beta(x,b)}\, e^{-\beta|s|-\frac{\beta}{2b}(s-x)^2}\,.$$

Simplifying messages for large N

Simplification of $\hat{\nu}^t_{a\to i}(s_i)$

$$
\begin{aligned}
\hat{\nu}^t_{a\to i}(s_i) &\cong \int e^{-\frac{\beta}{2}(y_a-(As)_a)^2} \prod_{j\neq i} \mathrm{d}\nu^t_{j\to a}(s_j) \\
&\cong \mathbb{E}\exp\left(-\frac{\beta}{2}(y_a - A_{ai}s_i - \sum_{j\neq i} A_{aj}s_j)^2\right) \\
&\cong \mathbb{E}\exp\Big(-\frac{\beta}{2}(A_{ai}s_i - Z)^2\Big)
\end{aligned}
$$

$Z = y_a - \sum_{j\neq i} A_{aj}s_j \xrightarrow{d} W$; W is Gaussian

$$
\left|\mathbb{E}h_{s_i}(Z) - \mathbb{E}h_{s_i}(W)\right| \leq \frac{C'''_t}{N^{1/2}(\hat{\tau}^t_{a\to i})^{3/2}}\,,
$$

As $N \to \infty$, $\hat{\nu}^t_{a\to i}$ converges to a Gaussian distribution.

Theorem

Let $x^t_{j\to a}$ and $(\tau^t_{j\to a}/\beta)$ denote the mean and variance of distribution $\nu^t_{j\to a}$. Then there exists a constant C'_t such that

$$\begin{aligned}
\sup_{s_i} |\hat{\nu}^t_{a\to i}(s_i) - \hat{\phi}^t_{a\to i}(s_i)| &\leq \frac{C'_t}{N(\hat{\tau}^t_{a\to i})^3}\,, \\
\hat{\phi}^t_{a\to i}(\mathrm{d}s_i) &\equiv \sqrt{\frac{\beta A_{ai}^2}{2\pi\hat{\tau}^t_{a\to i}}}\, e^{\beta(A_{ai}s_i - z^t_{a\to i})^2/2\hat{\tau}^t_{a\to i}}\, \mathrm{d}s_i\,,
\end{aligned}$$

where

$$z^t_{a\to i} \equiv y_a - \sum_{j\neq i} A_{aj} x^t_{j\to a}, \qquad \hat{\tau}^t_{a\to i} \equiv \sum_{j\neq i} A_{aj}^2 \tau^t_{j\to a}.$$

Donoho, M., Montanari (11)

Shape of $\nu^t_{i\to a}$; intuition

Summary

- $\hat{\nu}^t_{b\to i}(s_i) \approx$ Gaussian
- $\nu^{t+1}_{i\to a}(s_i) \cong e^{-\beta\lambda|s_i|} \prod_{b\neq a} \nu^t_{b\to i}(s_i)$

Shape of $\nu^{t+1}_{i\to a}(s_i)$

- $\nu^{t+1}_{i\to a}(s_i) \equiv \frac{1}{z_\beta(x,b)}\, e^{-\beta|s|-\frac{\beta}{2b}(s-x)^2}$

Message passing (cont'd)

Theorem

Suppose that $\hat{\nu}^t_{a\to i} = \hat{\phi}^t_{a\to i}$*, with mean* $z^t_{a\to i}$ *and variance* $\hat{\tau}^t_{a\to i} = \hat{\tau}^t$*. Then at the next iteration we have*

$$|\nu^{t+1}_{i\to a}(s_i) - \phi^{t+1}_{i\to a}(s_i)| < C/n\,,$$
$$\phi^{t+1}_{i\to a}(s_i) \equiv \lambda\, f_\beta(\lambda s_i; \lambda \sum_{b\neq a} A_{bi} z^t_{b\to i}, \lambda^2(1+\hat{\tau}^t))\,,$$

and

$$f_\beta(s; x, b) \equiv \frac{1}{z_\beta(x,b)}\, e^{-\beta|s| - \frac{\beta}{2b}(s-x)^2}\,.$$

Donoho, M., Montanari (11)

Next asymptotic: $\beta \to \infty$

Reminder:

- $x^t_{j\to a}$ mean of $\nu^t_{j\to a}$
- $z^t_{a\to i}$ mean of $\hat{\nu}^t_{a\to i}$
- $z^t_{a\to i} \equiv y_a - \sum_{j\neq i} A_{aj} x^t_{j\to a}$

Can we write $x^t_{j\to a}$ in terms of $z^t_{a\to i}$?

$$x^t_{j\to a} = \frac{1}{z_\beta} \int s_i \mathrm{e}^{-\beta|s_i| - \beta(s_i - \sum_{b\neq a} A_{bi} z^t_{b\to i})^2}$$

- Can be done, but ...

Laplace method simplifies it $(\beta \to \infty)$

$$x^t_{j\to a} = \eta(\sum_{b\neq a} A_{bi} z^t_{b\to i}).$$

Summary

$x^t_{j\to a}$ mean of $\nu^t_{j\to a}$
$z^t_{a\to i}$ mean of $\hat{\nu}^t_{a\to i}$

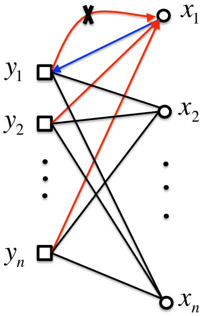
$$z^t_{a\to i} = y_a - \sum_{j\neq i} A_{aj} x^t_{j\to a}$$
$$x^{t+1}_{i\to a} = \eta(\sum_{b\neq a} A_{bi} z^t_{b\to i}).$$

Comparison of MP and IST

MP

$$z^t_{a\to i} = y_a - \sum_{j\neq i} A_{aj} x^t_{j\to a}$$

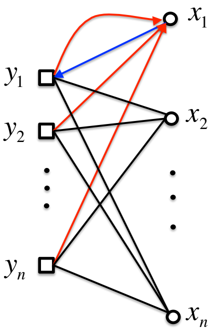
$$x^{t+1}_{i\to a} = \eta\Big(\sum_{b\neq a} A_{bi} z^t_{b\to i}\Big).$$



IST

$$z^t_{a\to i} = y_a - \sum_{j\neq i} A_{aj} x^t_{j\to a}$$

$$x^{t+1}_{i\to a} = \eta\Big(\sum_{b} A_{bi} z^t_{b\to i}\Big).$$



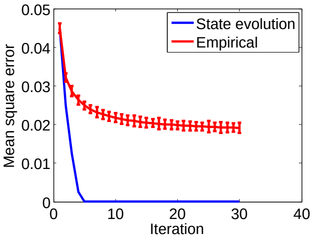
We can further simplify the messages

$$
\begin{aligned}
x^{t+1} &= \eta(x^t + A^T z^t; \lambda^t) \\
z^t &= y - Ax^t + \frac{\|x^t\|_0}{n} z^{t-1}
\end{aligned}
$$

IST versus AMP

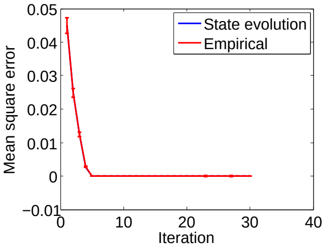
IST:

$$x^{t+1} = \eta(x^t + A^T z^t; \lambda^t)$$
$$z^t = y - Ax^t$$



AMP:

$$x^{t+1} = \eta(x^t + A^T z^t; \lambda^t)$$
$$z^t = y - Ax^t + \frac{1}{n}\|x^t\|_0 z^{t-1}$$



Theoretical guarantees

We can predict the performance of AMP at every iteration in asymptotic regime

Idea:

- $x^t + A^T z^t$ can be modeled as signal plus Gaussian noise
- We keep track of noise variance
- I will discuss this part in the "Risk Seminar"