

Compressive Sensing and Structured Sparse Recovery

An Liu (刘安)

College of Information Science & Electronic Engineering
Zhejiang University

Who am I?



Instructor: 刘安
Office: 第11教学楼201
Email: anliu@zju.edu.cn



北京大学 理学学士
电子科学与技术专业

2000.09 ~
2004.06

北京市优秀毕业生
北京大学优秀毕业生



北京大学 工学博士
通信与信息系统专业
导师:项海格

2004.09 ~
2011.01

北京大学信息科学
技术学院学术十杰



美国科罗拉多大学访问学生
通信与信号处理实验室
导师: Youjian Liu

2008.10 ~
2010.10



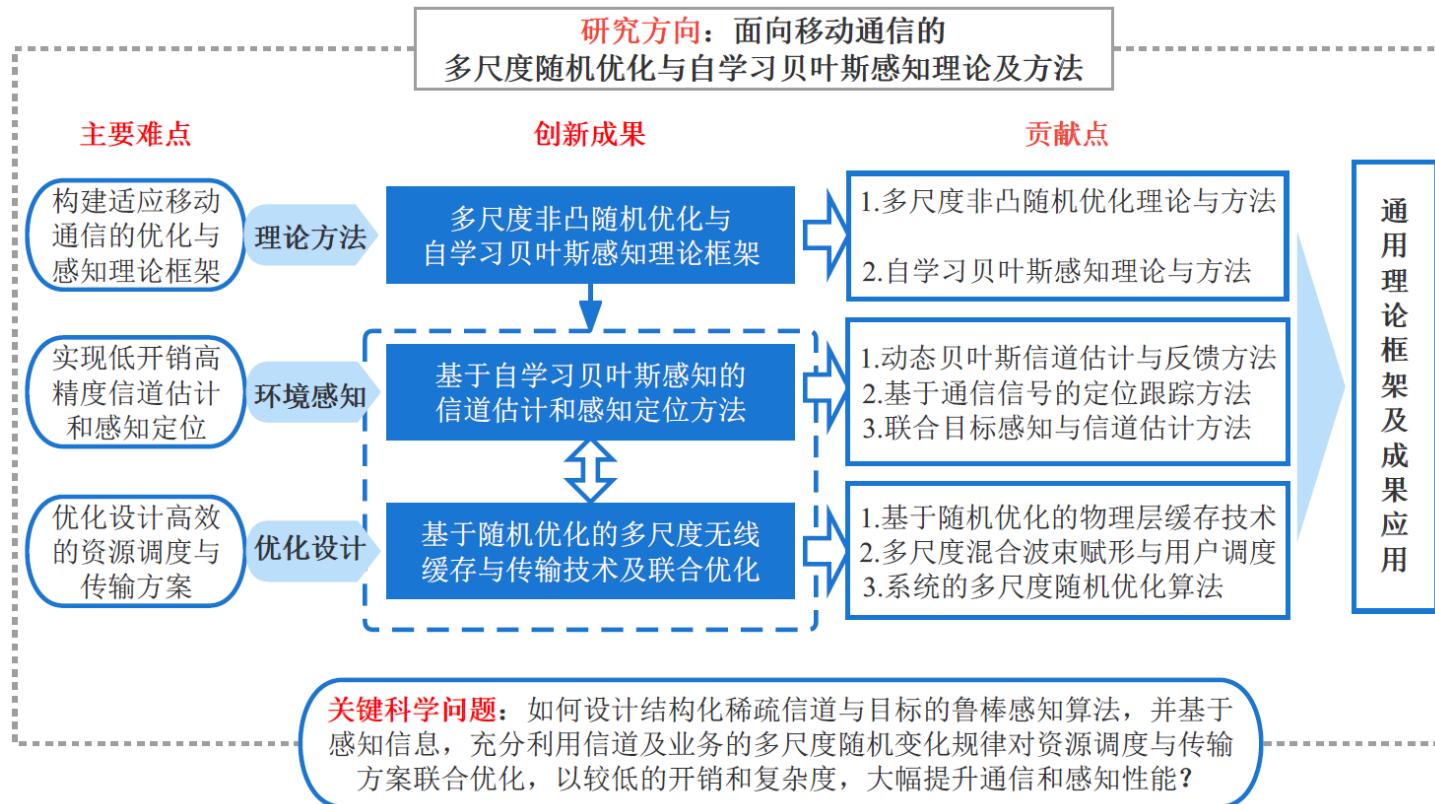
2011年至2018年2月在香港科技大学先后担任如下职位：

- 2011.03 ~ 2014.01 博士后 (合作导师: Prof. Vincent Lau, IEEE Fellow)
- 2014.01 ~ 2015.01 访问助理教授 (Visiting Assistant Professor)
- 2015.01 ~ 2018.02 研究助理教授 (Research Assistant Professor)



2018年3月至今在浙江大学担任百人计划研究员

研究领域：通信信号处理、无线通信与网络





代表性学术成果一览

共发表IEEE期刊与会议论文170余篇

- IEEE期刊论文100余篇
- 主要发表在TSP、TWC、TIT、JSAC等通信、信号处理、信息论领域的顶级期刊

英文专著章节3章，授权专利9项，包括美国专利1项

期刊简称	期刊全称	影响因子
IEEE TSP	IEEE Transactions on Signal Processing	4. 875
IEEE TWC	IEEE Transactions on Wireless Communications	8. 346
IEEE TIT	IEEE Transactions on Information Theory	2. 978
IEEE JSAC	IEEE Journal on Selected Areas in Communications	13. 081
IEEE JSTSP	IEEE Journal of Selected Topics in Signal Processing	7. 695
IEEE COMST	IEEE Communications Surveys & Tutorials	33. 84
IEEE WCL	IEEE Wireless Communications Letters	5. 281

国际重要学术组织的主要任职情况



职位	期刊/会议/组织名称	任职时间
Editor (编委)	IEEE TSP (IF = 4.875)	2020. 02 - 今
Editor (编委)	IEEE TWC (IF = 8.346)	2020. 01 - 今
Editor (编委)	IEEE WCL (IF = 5.281)	2017. 5 - 2022. 12
技术委员会委员	IEEE SPCOM 技术委员会 (SPCOM 技术委员会承担 IEEE 信号处理学会评奖、学术会议组织等重要职责，委员总数不超过 30 个)	2023. 01 - 今
Founding Member	IEEE <u>ComSoc</u> ISAC-ETI (IEEE 通信学会通信感知一体化新兴技术倡议委员会)	2021 - 今



课程评分方式

- Class participation: 10%
- Homework: 40% (5 assignments)
- Final Project: 50%

基本教材



本课程没有教材，各章节会列出一些参考文献，主要参考文献如下：

- [1] Yonina C. Eldar, "Compressed Sensing: Theory and Applications". Cambridge, U.K.: Cambridge Univ. Press, 2012.
- [2] R. Baraniuk, M. A. Davenport, M. F. Duarte, C. Hegde, J. Laska, M. Sheikh, and W. Yin, "An introduction to compressive sensing," 2011 [Online]. Available: <http://cnx.org/content/col11133/1.5/>
- [3] F. R. Kschischang, B. J. Frey and H. - Loeliger, "Factor graphs and the sum-product algorithm," in IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 498-519, Feb 2001.
- [4] D. G. Tzikas, A. C. Likas, and N. P. Galatsanos, "The Variational Approximation for Bayesian Inference," Signal Processing Magazine IEEE, vol. 25, no. 6, pp. 131–146, 2008.
- [5] J. Ma, X. Yuan, and L. Ping, "On the Performance of Turbo Signal Recovery with Partial DFT Sensing Matrices," IEEE Signal Processing Letters, vol. 22, no. 10, pp. 1580–1584, Oct 2015.
- [6] A. Liu, G. Liu, L. Lian, V. K. N. Lau and M. Zhao, "Robust Recovery of Structured Sparse Signals With Uncertain Sensing Matrix: A Turbo-VBL Approach," in IEEE Transactions on Wireless Communications, vol. 19, no. 5, pp. 3185-3198, May 2020.



Part I: Parameter Estimation

- Bayesian Decision Theory (& MAP)
- Maximum Likelihood Estimation
- Least Square Estimation
- MMSE Estimation
- Linear MMSE Estimation
- Bayesian Estimation
- Expectation Maximization
- Factor Graph and Message Passing
- Music
- Performance Bounds



Part II: Compressive Sensing (CS)

- Compressive Sensing in a Nutshell
- Optimization Based CS recovery Algorithm
 - ℓ_1 -norm minimization, Reweighted ℓ_1 -norm minimization, LASSO, Generalized LASSO
 - RIP and Performance Guarantee
- Greedy CS Recovery Algorithms
 - OMP, CoSaMP, SP
- Message Passing CS Recovery Algorithms
 - AMP, Big-AMP for bilinear models
- Applications to Wireless Communications



Part III: Bayesian Inference

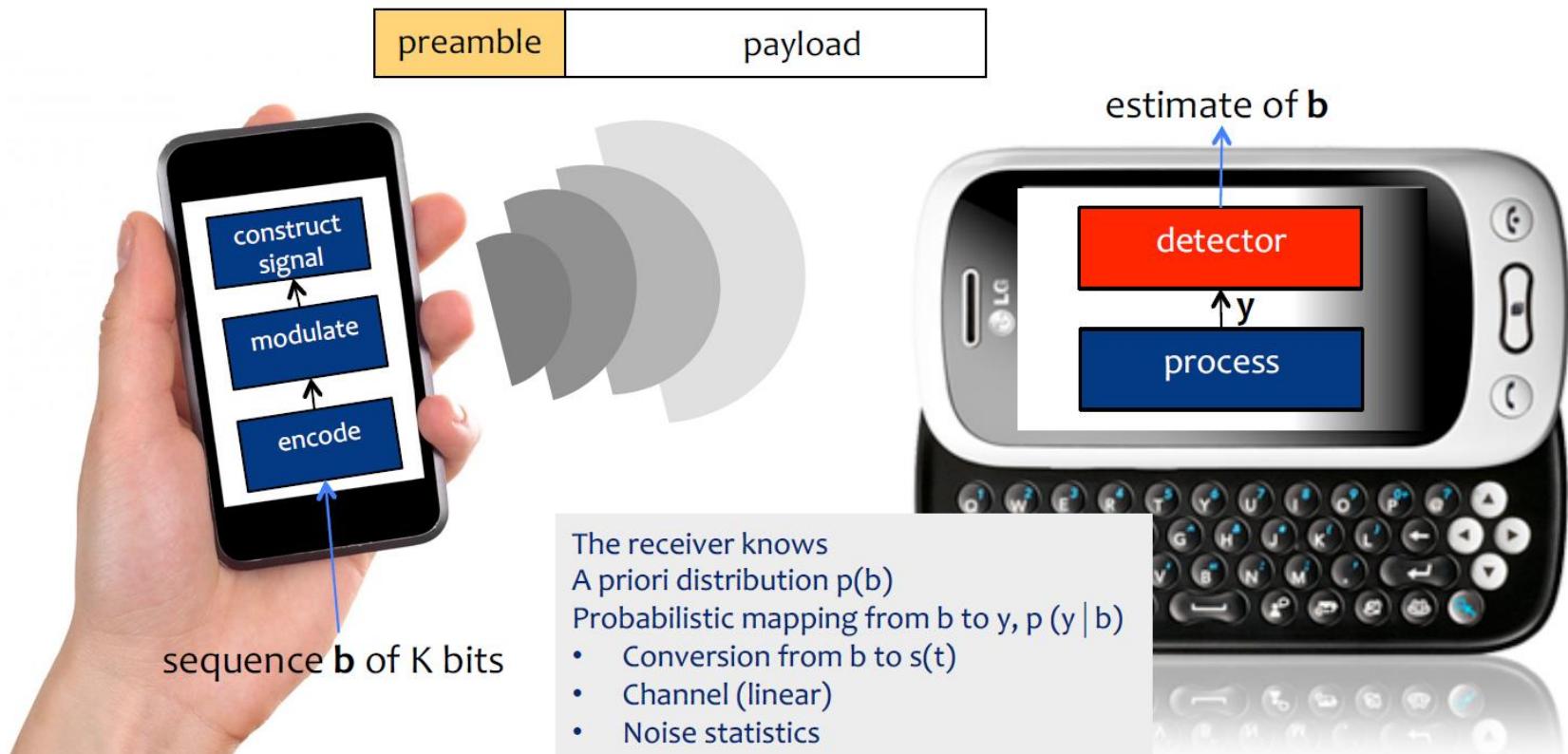
- Variational Bayesian Inference
- Sparse Bayesian Learning
- Turbo CS
- Turbo VBI
- Dynamic Bayesian Learning
- Applications to Wireless Communications



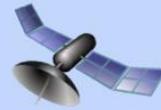
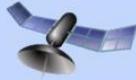
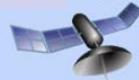
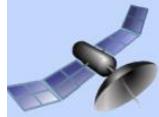
Applications

receiver design

- Data detection problem: recover \mathbf{b} from \mathbf{y} (optimally)
- Many variations: codes, mapping, channels, antennas, users



cooperative localization



Scenario

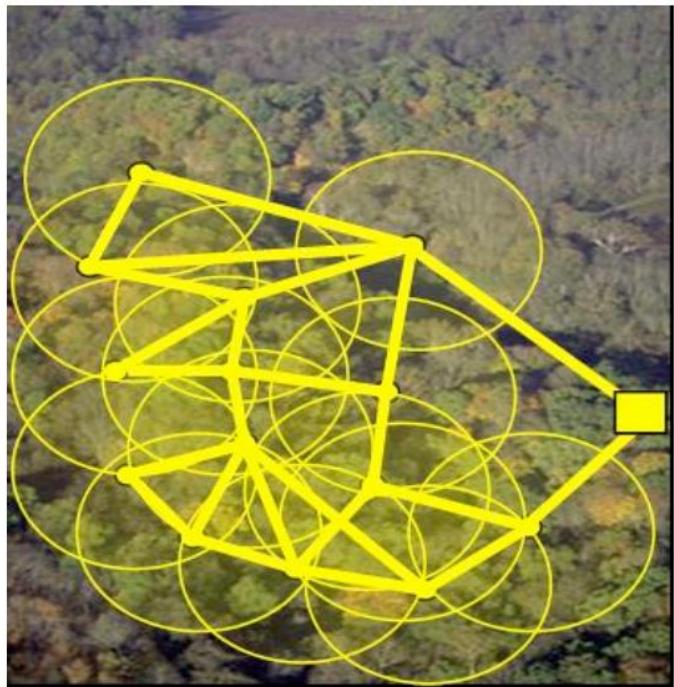
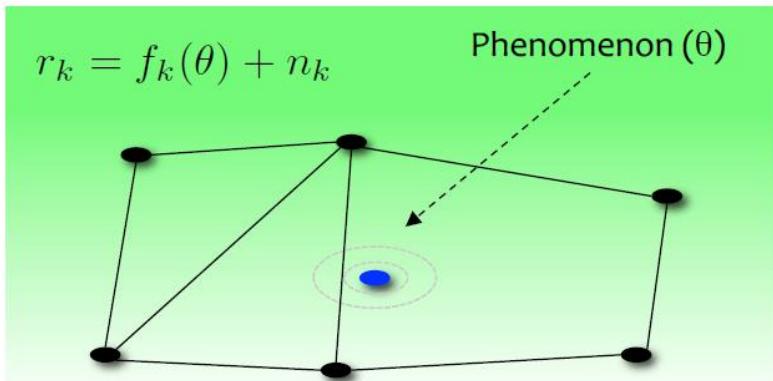
- N mobile nodes (agents) with **unknown** positions $x_1(t), \dots, x_N(t)$
- M reference nodes (anchors) with **known** positions $x_A(t), x_B(t), x_C(t), \dots$
- **Measurements** $z_{ij}(t)$ between $x_i(t)$ and $x_j(t)$, where j is mobile or reference

Goals

- For every mobile node to determine its own position, given all measurements up to now



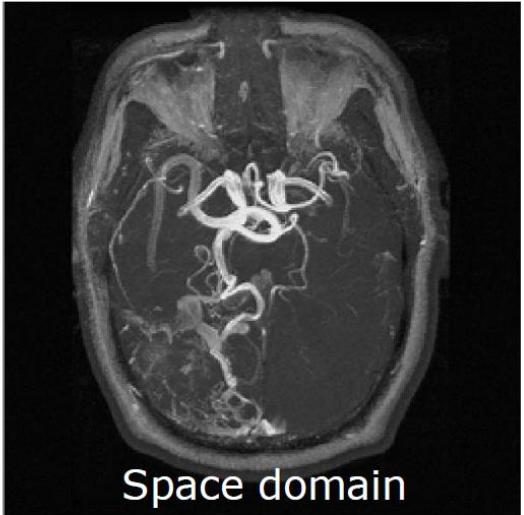
wireless sensor network



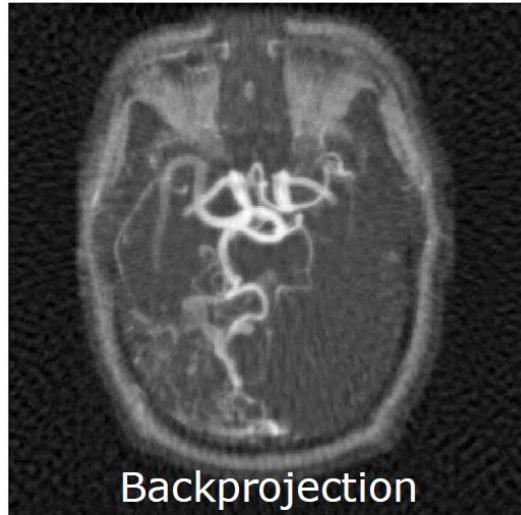
Goal:

Jointly compute estimate of θ

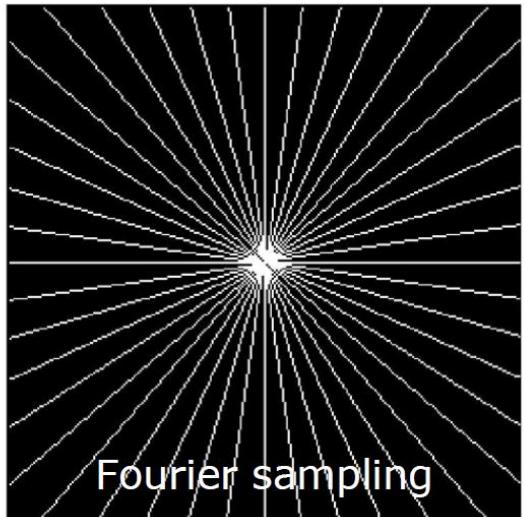
Magnetic Resonance Imaging



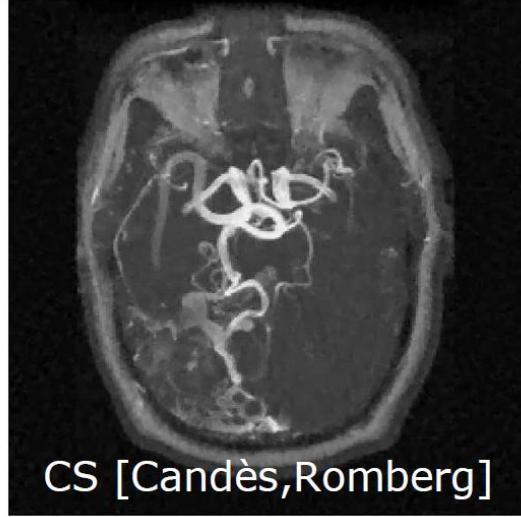
Space domain



Backprojection



Fourier sampling



CS [Candès,Romberg]



Analog-to-Information Conversion

[with E. Candès and J. Romberg]

DARPA A/I Project:

Efficient sampling of high-bandwidth signals

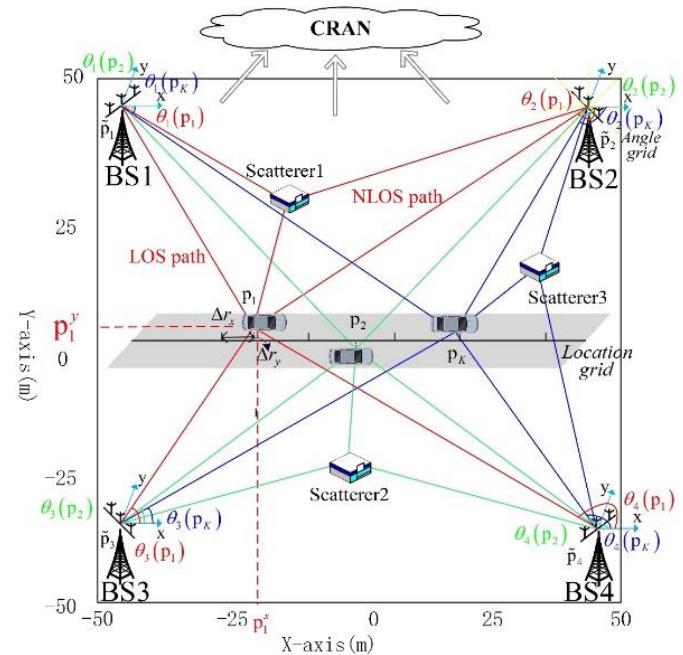
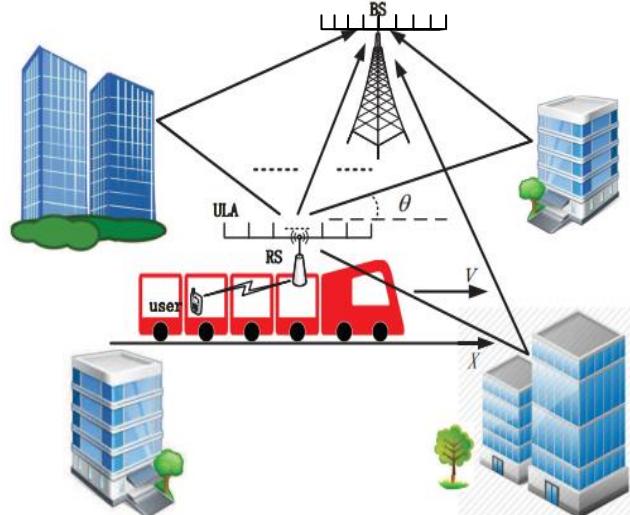
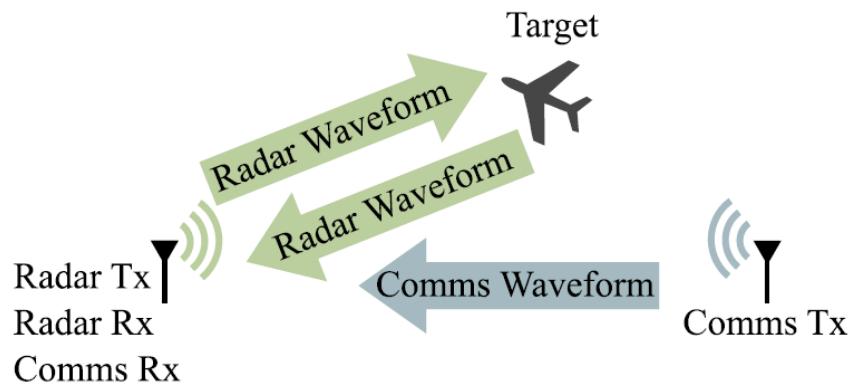
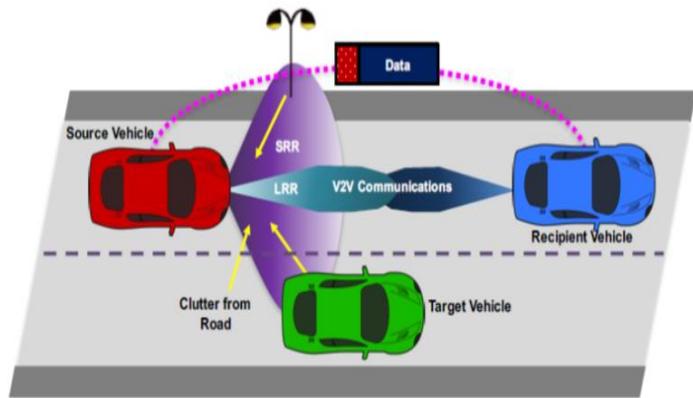
- sparse models allow sampling far below Nyquist rate
- new architectures for incoherent measurements



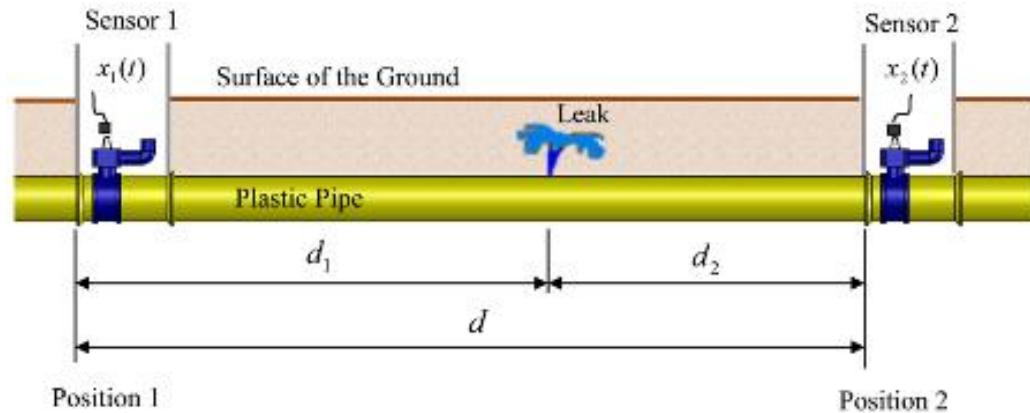
$$x(t) \approx \sum_{k=1}^K \alpha_k \psi_k(t)$$

$$y_m \approx \langle x, \phi_m \rangle = \int_{-\infty}^{\infty} x(t) \phi_m(t) dt$$

Sensing via Communication Signals



We want to detect the pipe leakage or blockage by measuring the frequency domain pressure response at the receiver.



Leak Identification in Smart Water Systems



Part I: Parameter Estimation

- Bayesian Decision Theory (& MAP)
- Maximum Likelihood Estimation
- Least Square Estimation
- MMSE Estimation
- Linear MMSE Estimation
- Bayesian Estimation
- Expectation Maximization
- Factor Graph and Message Passing
- Music
- Performance Bounds



Bayesian Decision Theory (& MAP)



Bayesian decision theory

- ▶ framework for computing optimal decisions on problems involving uncertainty (probabilities)
- ▶ basic concepts:
 - world:
 - has states or classes, drawn from a state or class random variable \mathbf{X}
 - fish classification, $\mathbf{X} \in \{\text{bass, salmon}\}$
 - student grading, $\mathbf{X} \in \{\text{A, B, C, D, F}\}$
 - medical diagnosis $\in \{\text{disease A, disease B, ..., disease M}\}$
 - observer:
 - measures observations (features), drawn from a random process \mathbf{Y}
 - fish classification, $\mathbf{Y} = (\text{scale length, scale width}) \in \mathbb{R}^2$
 - student grading, $\mathbf{Y} = (\text{HW}_1, \dots, \text{HW}_n) \in \mathbb{R}^n$
 - medical diagnosis $\mathbf{Y} = (\text{symptom 1, ..., symptom n}) \in \mathbb{R}^n$



Bayesian decision theory

- decision function:
 - observer uses the observations to make decisions about the state of the world x
 - if $y \in \Omega$ and $x \in \Psi$ the decision function is the mapping

$$g : \Omega \rightarrow \Psi$$

such that

$$g(y) = \hat{x}$$

and \hat{x} is a prediction of the state x

- loss function:
 - is the cost $L(\hat{x}, x)$ of deciding for \hat{x} when the true state is x
 - usually this is zero if there is no error and positive otherwise
- goal: to determine the optimal decision function for the loss $L(.,.)$

Examples

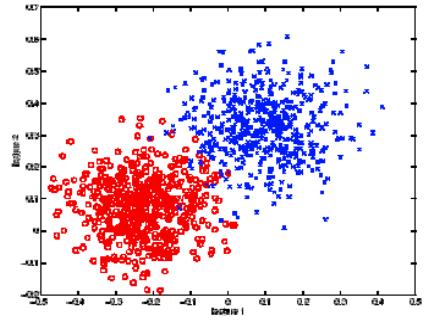
► Example 1: classification problems

- the observer tries to infer the state of the world

$$g(y) = \hat{x}, \hat{x} \in \{1, \dots, M\}$$

- we will also mostly consider the “0-1” loss function

$$L(g(y), x) = \begin{cases} 1, & g(y) \neq x \\ 0, & g(y) = x \end{cases}$$



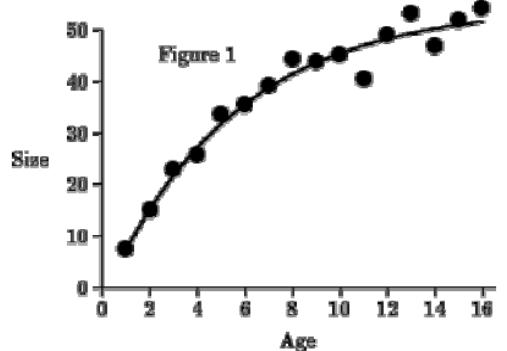
► Example 2: Estimation of a continuous variable

- the observer tries to predict a continuous x

$$g(y) \in \mathbb{R}$$

- is basically the same, for a suitable loss function, e.g. squared error

$$L(g(y), x) = \|g(y) - x\|^2$$





Tools for solving BDT problem

► probabilistic representations

- joint distribution
- Likelihood distributions
- prior probabilities

► properties of probabilistic inference

- chain rule of probability
- marginalization
- independence
- Bayes rule



Tools for solving BDT problem

- ▶ in order to find optimal decision function we need a **probabilistic description** of the problem

- in the most general form this is the **joint distribution**

$$P_{X,Y}(x, y)$$

- but we frequently **decompose** it into a combination of two terms

$$P_{X,Y}(x, y) = P_{Y|X}(y|x) P(x)$$



- these are the “ Likelihood distribution” and “ prior probability”
 - prior probability
 - prior probability of state x before observer actually measures anything
 - reflects a “prior belief” that, if all else is equal, the world will be in state x with probability $P(x)$ - Likelihood is the **model for the observations** given the class or state of the world



The chain rule of probability

- ▶ is an important consequence of the definition of conditional probability

- note that, by recursive application of

$$P_{X,Y}(x, y) = P_{Y|X}(y|x) P(x)$$

- we can write

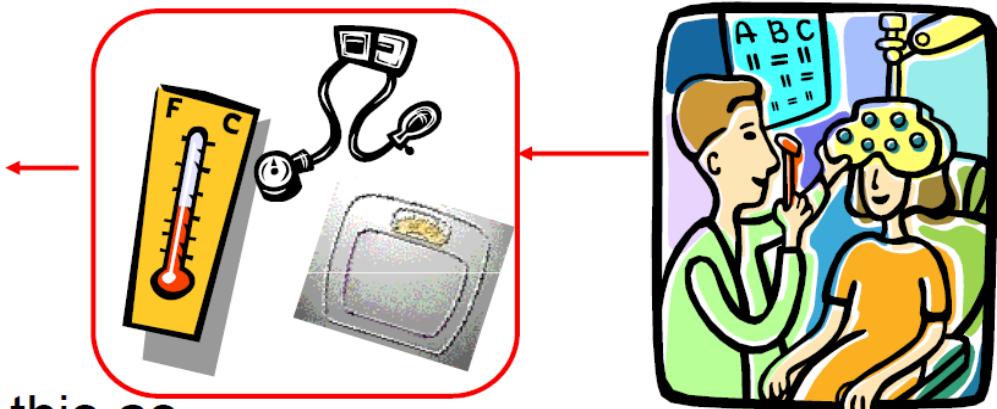
$$\begin{aligned} P_{X_1, X_2, \dots, X_n}(x_1, x_2, \dots, x_n) &= P_{X_1|X_2, \dots, X_n}(x_1 | x_2, \dots, x_n) \times \\ &\quad \times P_{X_2|X_3, \dots, X_n}(x_2 | x_3, \dots, x_n) \times \dots \\ &\quad \times \dots \times P_{X_{n-1}|X_n}(x_{n-1} | x_n) P_{X_n}(x_n) \end{aligned}$$

- ▶ this is called the chain rule of probability
- ▶ it allows us to modularize inference problems

Tools for solving BDT problems

► frequently we have problems with multiple random variables

- e.g. when in the doctor, you are mostly a collection of random variables
 - x_1 : temperature
 - x_2 : blood pressure
 - x_3 : weight
 - x_4 : cough



► we can summarize this as

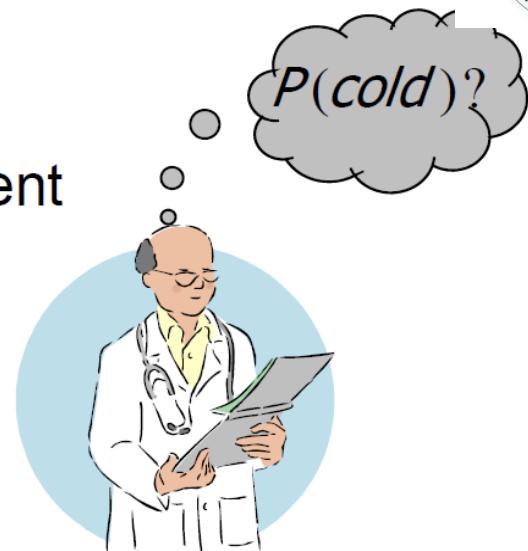
- a vector $\mathbf{X} = (x_1, \dots, x_n)$ of n random variables
- $P_{\mathbf{X}}(x_1, \dots, x_n)$ is the joint probability distribution

► but frequently we only care about a subset of \mathbf{X}

Marginalization

► what if I only want to know if the patient has a cold or not?

- does not depend on blood pressure and weight
- all that matters are fever and cough
- that is, we need to know $P_{X_1, X_4}(a, b)$



► we marginalize with respect to a subset of variables

- (in this case X_1 and X_4)
- this is done by summing (or integrating) the others out

$$P_{X_1, X_4}(x_1, x_4) = \sum_{x_3, x_4} P_{X_1, X_2, X_3, X_4}(x_1, x_2, x_3, x_4)$$

$$P_{X_1, X_4}(x_1, x_4) = \int \int P_{X_1, X_2, X_3, X_4}(x_1, x_2, x_3, x_4) dx_2 dx_3$$



Marginalization

► important equation:

- seems trivial, but for large models is a major computational asset for probabilistic inference
- for any question, there are lots of variables which are irrelevant
- direct evaluation is frequently intractable
- typically, we combine with the chain rule to explore independence relationships that will allow us to reduce computation

► independence:

- X and Y are independent random variables if

$$P_{X|Y}(x | y) = P_X(x)$$



Independence

- ▶ combined with marginalization, enables efficient computation

- e.g to compute $P_Y(\text{sick})$
- 1) marginalization

$$P_Y(\text{sick}) = \sum_s \int P_{Y|X_1,S}(\text{sick} | x, s) dx$$

- 2) chain rule

$$P_Y(\text{sick}) = \sum_s \int P_{Y|X_1,S}(\text{sick} | x, s) P_{S|X_1}(s | x) P_{X_1}(x) dx$$

- 3) independence

$$P_Y(\text{sick}) = \int P_{Y|X_1}(\text{sick} | x) P_{X_1}(x) \sum_s P_{S|X_1}(s | x) dx$$

- dividing and grouping terms (divide and conquer) makes the integral simpler



Tools for solving BDT problems

► Bayes rule

$$P_{X|Y}(x|y) = \frac{P_{Y|X}(y|x) P(x)}{P(y)}$$

- is the **central equation of Bayesian inference**
- allows us to “switch” the relation between the variables
- this is **extremely useful**
- e.g. for medical diagnosis doctor needs to know

$$P_{X|Y}(\text{disease } x | \text{symptom } y)$$

- this is very complicated because it is not causal
- we are asking for the probability of cause given consequence



Tools for solving BDT problems

- Bayes rule transforms it into the probability of consequence given cause

$$P_{X|Y}(\text{disease } x | \text{symptom } y) =$$

$$= \frac{P_{Y|X}(\text{symptom } y | \text{disease } x) P_x(\text{disease } x)}{P_Y(\text{symptom } y)}$$

and some other stuff

- note that $P_{Y|X}(\text{symptom } y | \text{disease } x)$ is easy – you can get it out of any medical textbook
- what about the other stuff?
 - $P_x(\text{disease } x)$ does not depend on the patient – you can get it by collecting statistics over the entire population
 - $P_Y(\text{symptom } y)$ is a combination of the two (marginalization)

$$P_Y(\text{symptom } y) = \sum_x P_{Y|X}(\text{symptom } y | \text{disease } x) P_x(\text{disease } x)$$



Bayesian decision theory

► recall that we have

- X – state of the world
- Y – **observations**
- $g(y)$ – **decision function**
- $L(g(y), x)$ – **loss** of predicting x with $g(y)$

► the expected value of the loss is called the **risk**

$$\underline{Risk} = E_{X,Y} [L(g(Y), X)]$$

- which can be written as

$$Risk = \int \int P_{X,Y}(x, y) L(g(y), x) dx dy$$

Bayesian decision theory

- from this

$$Risk = \int \int P_{X,Y} (x, y) L(g(y), x) dx dy$$

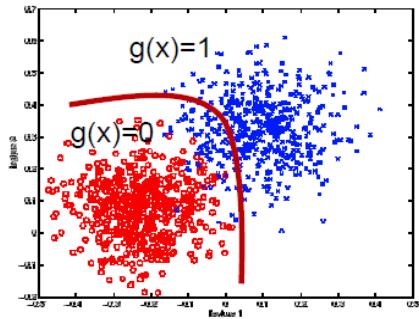
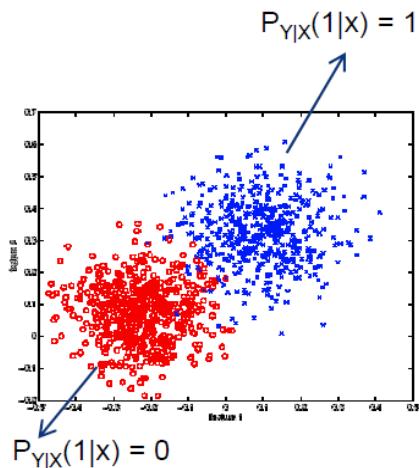
- by chain rule

$$\begin{aligned} Risk &= \int P_Y(y) \int P_{X|Y}(x|y) L(g(y), x) dx dy \\ &= \int P_Y(y) R(y) dy = E_Y[R(y)] \end{aligned}$$

- where

$$R(y) = \int P_{X|Y}(x|y) L(g(y), x) dx$$

- is the conditional risk, given the observation y



Bayesian decision theory

- since, by definition,

$$L(g(y), x) \geq 0, \quad \forall x, y$$

- it follows that

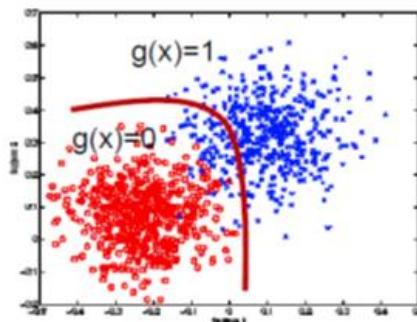
$$R(y) = \int P_{X|Y}(x|y) L(g(y), x) dx \geq 0, \quad \forall y$$

- hence

$$\text{Risk} = E_Y [R(y)]$$

is minimum if we minimize $R(y)$ at all y i.e., if we use pick the decision function

$$g^*(y) = \operatorname{argmin}_{g(y)} \int P_{X|Y}(x|y) L(g(y), x) dx$$



Bayesian decision theory

- this is the Bayes decision rule

$$g^*(y) = \operatorname{argmin}_{g(y)} \int P_{X|Y}(x|y) L(g(y), x) dx$$

- the associated risk

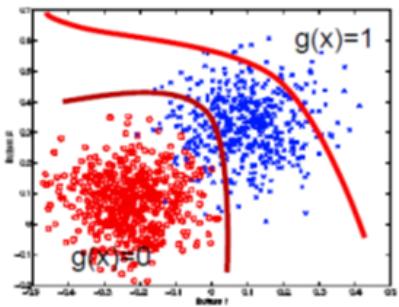
$$R^* = \int \int P_{X,Y}(x, y) L(g^*(y), x) dx dy$$

- or

$$R^* = \int P_Y(y) \int P_{X|Y}(x|y) L(g^*(y), x) dx dy$$

- is the Bayes risk, and cannot be beaten

which decision rule has smaller conditional risk?





Example

- ▶ let's consider the “0-1” loss

$$L(g(y), x) = \begin{cases} 1, & g(y) \neq x \\ 0, & g(y) = x \end{cases}$$

- ▶ for the “0-1” loss the optimal decision rule is the maximum a-posteriori probability rule

$$g^*(y) = \operatorname{argmax}_x P_{X|Y}(x|y)$$

- ▶ what is the associated risk?

$$R^* = \int P_{X,Y}(x \neq g^*(y), y) dy$$

Prove the above results by yourself!



MAP rule

► in summary

- for the zero/one loss, the following three decision rules are optimal and equivalent
 - 1) $\underline{g^*(y) = \operatorname{argmax}_x P_{X|Y}(x|y)}$
 - 2) $\underline{g^*(y) = \operatorname{argmax}_x P_{Y|X}(y|x) P_X(x)}$
 - 3) $\underline{g^*(y) = \operatorname{argmax}_x \log P_{Y|X}(y|x) + \log P_X(x)}$
- 1) is usually hard to use, 3) is frequently easier than 2)



Maximum Likelihood Estimation

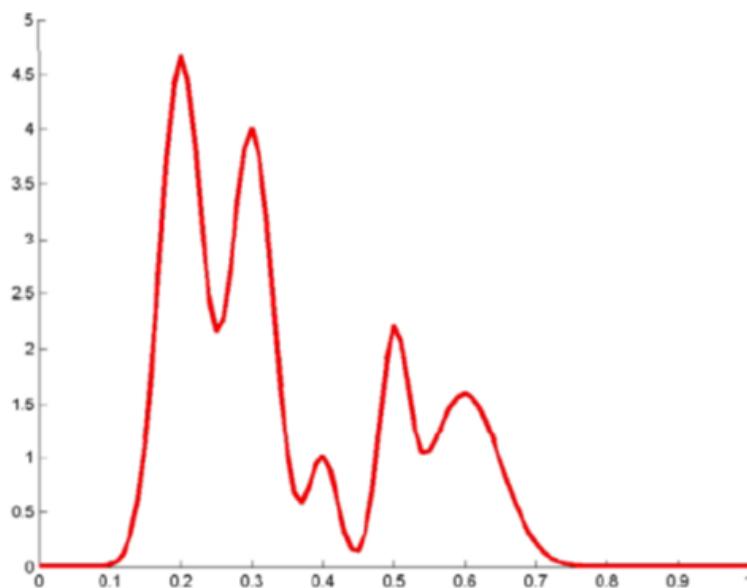


Bayesian decision theory

- advantages:
 - BDR is optimal and cannot be beaten
 - Bayes keeps you honest
 - models reflect causal interpretation of the problem, this is how we think
 - natural decomposition into “what we knew already” (prior) and “what data tells us” (CCD)
 - no need for heuristics to combine these two sources of info
 - BDR is, almost invariably, intuitive
 - Bayes rule, chain rule, and marginalization enable modularity, and scalability to very complicated models and problems
- problems:
 - BDR is optimal only insofar the models are correct.

Important

- **warning:** at this point all optimality claims for the BDR cease to be valid!!
- the BDR is guaranteed to achieve the minimum loss when we use the true probabilities
- when we “plug in” the probability estimates, we could be implementing a classifier that is quite distant from the optimal
 - e.g. if the $P_{Y|X}(y|x)$ look like the example above
 - I could never approximate them well by parametric models (e.g. Gaussian)





Maximum likelihood

- this seems pretty serious
 - how should I get these probabilities then?
- we rely on the maximum likelihood (ML) principle
- this has three steps:
 - 1) we choose a parametric model for all probabilities
 - to make this clear we denote the vector of parameters by Θ and the Likelihood distributions by

$$P_{X_i}(x_i | \Theta)$$

- note that this means that Θ is NOT a random variable (otherwise it would have to show up as subscript)
- it is simply a parameter, and the probabilities are a function of this parameter



Maximum likelihood

- three steps:
 - 2) we assemble a collection of datasets
 $D = \{x_1, x_2, \dots, x_n\}$ set of examples drawn independently
 - 3) we select the parameters to be the ones that maximize the probability of the data from that class
- like before, it does not really make any difference to maximize probabilities or their logs

$$\begin{aligned}\Theta^* &= \arg \max_{\Theta} P_X(D; \Theta) \\ &= \arg \max_{\Theta} \log P_X(D; \Theta)\end{aligned}$$



Maximum likelihood

- let's consider the Gaussian example

$$f(T) = \frac{1}{\sigma_T \sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{T-\bar{T}}{\sigma_T} \right)^2}$$

- given a sample $\{T_1, \dots, T_N\}$ of independent points
- the likelihood is

$$L(T_1, T_2, \dots, T_N | \bar{T}, \sigma_T) = L = \prod_{i=1}^N \left[\frac{1}{\sigma_T \sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{T_i - \bar{T}}{\sigma_T} \right)^2} \right]$$
$$L = \frac{1}{(\sigma_T \sqrt{2\pi})^N} e^{-\frac{1}{2} \sum_{i=1}^N \left(\frac{T_i - \bar{T}}{\sigma_T} \right)^2}$$



Maximum likelihood

- and the log-likelihood is

$$\Lambda = \ln L = -\frac{N}{2} \ln(2\pi) - N \ln \sigma_T - \frac{1}{2} \sum_{i=1}^N \left(\frac{T_i - \bar{T}}{\sigma_T} \right)^2$$

- the derivative with respect to the mean is zero when

$$\frac{\partial(\Lambda)}{\partial \bar{T}} = \frac{1}{\sigma_T^2} \sum_{i=1}^N (T_i - \bar{T}) = 0$$

- or

$$\bar{T} = \frac{1}{N} \sum_{i=1}^N T_i$$

- note that this is just the sample mean



Maximum likelihood

- and the log-likelihood is

$$\Lambda = \ln L = -\frac{N}{2} \ln(2\pi) - N \ln \sigma_T - \frac{1}{2} \sum_{i=1}^N \left(\frac{T_i - \bar{T}}{\sigma_T} \right)^2$$

- the derivative with respect to the variance is zero when

$$\frac{\partial(\Lambda)}{\partial \sigma_T} = -\frac{N}{\sigma_T} + \frac{1}{\sigma_T^3} \sum_{i=1}^N (T_i - \bar{T})^2 = 0$$

- or

$$\hat{\sigma}_T^2 = \frac{1}{N} \sum_{i=1}^N (T_i - \bar{T})^2$$

- note that this is just the sample variance



Estimators

- when we talk about estimators, it is important to keep in mind that
 - an estimate is a number
 - an estimator is a random variable

$$\hat{\theta} = f(X_1, \dots, X_n)$$

- an estimate is the value of the estimator for a given sample.
- if $\mathcal{D} = \{x_1, \dots, x_n\}$, when we say $\hat{\mu} = \frac{1}{n} \sum_j x_j$

what we mean is $\hat{\mu} = f(X_1, \dots, X_n) \Big|_{X_1=x_1, \dots, X_n=x_n}$ with

$$f(X_1, \dots, X_n) = \frac{1}{n} \sum_j X_j$$

← the X_i are random variables



Bias and variance

- we know how to produce estimators (by ML)
- how do we evaluate an estimator?
- Q₁: is the expected value equal to the true value?
- this is measured by the bias
 - if

$$\hat{\theta} = f(X_1, \dots, X_n)$$

then

$$Bias(\hat{\theta}) = E_{X_1, \dots, X_n} [f(X_1, \dots, X_n) - \theta]$$

- an estimator that has bias will usually not converge to the perfect estimate θ , no matter how large the sample is
- e.g. if θ is negative and the estimator is $f(X_1, \dots, X_n) = \frac{1}{n} \sum_j X_j^2$ the bias is clearly non-zero



Bias and variance

- the estimators is said to be **biased**
 - this means that it is **not expressive** enough to approximate the true value arbitrarily well
 - this will be clearer when we talk about density estimation
- Q₂: assuming that the estimator converges to the true value, **how many sample points do we need?**
 - this can be measured by the **variance**

$$Var(\hat{\theta}) = E_{X_1, \dots, X_n} \left\{ (f(X_1, \dots, X_n) - E_{X_1, \dots, X_n} [f(X_1, \dots, X_n)])^2 \right\}$$

- the variance **usually decreases** as one collects more training examples



Example

- ML estimator for the mean of a Gaussian $N(\mu, \sigma^2)$

$$\begin{aligned} Bias(\hat{\mu}) &= E_{X_1, \dots, X_n} [\hat{\mu} - \mu] = E_{X_1, \dots, X_n} [\hat{\mu}] - \mu \\ &= E_{X_1, \dots, X_n} \left[\frac{1}{n} \sum_i X_i \right] - \mu \\ &= \frac{1}{n} \sum_i E_{X_1, \dots, X_n} [X_i] - \mu \\ &= \frac{1}{n} \sum_i E_{X_i} [X_i] - \mu \\ &= \mu - \mu = 0 \end{aligned}$$

- the estimator is unbiased



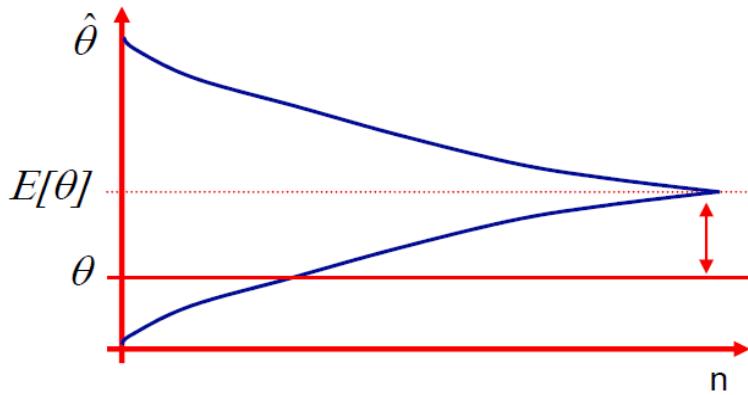
Example

- variance of ML estimator for mean of a Gaussian $N(\mu, \sigma^2)$

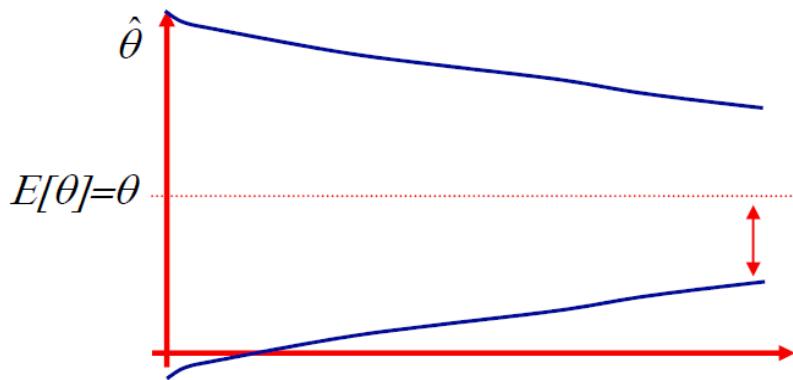
$$\begin{aligned}Var(\hat{\mu}) &= E_{X_1, \dots, X_n} \left\{ (\hat{\mu} - E_{X_1, \dots, X_n} [\hat{\mu}])^2 \right\} = E_{X_1, \dots, X_n} \left\{ (\hat{\mu} - \mu)^2 \right\} \\&= E_{X_1, \dots, X_n} \left\{ \left(\frac{1}{n} \sum_i X_i - \mu \right)^2 \right\} \\&= \frac{1}{n^2} E_{X_1, \dots, X_n} \left\{ \left(\sum_i (X_i - \mu) \right)^2 \right\} \\&= \frac{1}{n^2} E_{X_1, \dots, X_n} \left\{ \sum_{ij} (X_i - \mu)(X_j - \mu) \right\}\end{aligned}$$

Bias and variance

- we really care about the conjunction of the two factors
 - working hard to decrease variance if bias is large is useless



- working hard to decrease bias if variance is large is useless





Mean squared error

- one possibility to account for both bias and variance is to minimize the mean squared error

– if

$$\hat{\theta} = f(X_1, \dots, X_n)$$

– then

$$MSE(\hat{\theta}) = E_{X_1, \dots, X_n} [\{f(X_1, \dots, X_n) - \theta\}^2]$$

- the connection to bias and variance follows from

$$MSE(\hat{\theta}) = E \left[\{ \hat{\Theta} - E[\hat{\Theta}] + E[\hat{\Theta}] - \theta \}^2 \right]$$

$$= E \left[\{ \hat{\Theta} - E[\hat{\Theta}] \}^2 \right] + 2E \left[\{ \hat{\Theta} - E[\hat{\Theta}] \} \{ E[\hat{\Theta}] - \theta \} \right]$$

$$+ E \left[\{ E[\hat{\Theta}] - \theta \}^2 \right]$$



Mean squared error

- $$\begin{aligned} MSE(\hat{\theta}) &= E\left[\left\{\hat{\Theta} - E[\hat{\Theta}] + E[\hat{\Theta}] - \theta\right\}^2\right] \\ &= E\left[\left\{\hat{\Theta} - E[\hat{\Theta}]\right\}^2\right] + 2E\left(\left\{\hat{\Theta} - E[\hat{\Theta}]\right\}\left\{E[\hat{\Theta}] - \theta\right\}\right) \\ &\quad + E\left[\left\{E[\hat{\Theta}] - \theta\right\}^2\right] \\ &= \text{var}(\hat{\Theta}) + 2E\left(\left\{\hat{\Theta} - E[\hat{\Theta}]\right\}\left\{E[\hat{\Theta}] - \theta\right\}\right) + \left\{E[\hat{\Theta}] - \theta\right\}^2 \\ &= \text{var}(\hat{\Theta}) + 2\left\{E[\hat{\Theta}] - E[\hat{\Theta}]\right\}\left\{E[\hat{\Theta}] - \theta\right\} + Bias^2(\hat{\Theta}) \end{aligned}$$

– and

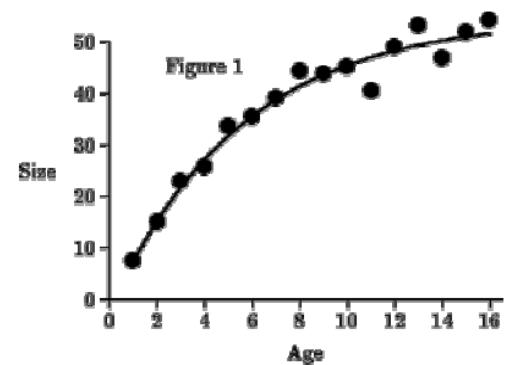
$$MSE(\hat{\Theta}) = \text{var}(\hat{\Theta}) + Bias^2(\hat{\Theta})$$



Least squares

- there are interesting connections between ML estimation and least squares methods
- e.g. in a regression problem we have
 - two random variables X and Y
 - a dataset of examples
 $\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\}$
 - a parametric model of the form

$$y = f(x; \Theta) + \varepsilon$$



- where Θ is a parameter vector, and ε a random variable that accounts for noise
- e.g. $\varepsilon \sim N(0, \sigma^2)$



Least squares

- assuming that the family of models is known, e.g.

$$f(x; \Theta) = \sum_{i=0}^K \theta_i x^i$$

- this is really just a problem of parameter estimation
- where the data is distributed as

$$P_{Z|X}(D | x; \Theta) = G(z, f(x; \Theta), \sigma^2)$$

- note that X is always known, and the mean is a function of x and Θ
- in the homework, you will show that

$$\Theta^* = [\Gamma^T \Gamma]^{-1} \Gamma^T y$$



Least squares

- where

$$\Gamma = \begin{bmatrix} 1 & \dots & x_1^K \\ & \vdots & \\ 1 & \dots & x_n^K \end{bmatrix}$$

- conclusion:
 - least squares estimation is really just ML estimation under the assumption of
 - Gaussian noise
 - independent sample
 - $\varepsilon \sim N(0, \sigma^2)$
- once again, probability makes the assumptions explicit



(Linear) Minimum Mean-Square Error Estimation

Linear Minimum Mean-Square Error Estimation (LMMSE)



- We have two jointly distributed random vectors \mathbf{X} and \mathbf{Y} .
- We observe \mathbf{Y} and we wish to “guess” the value of \mathbf{X} in some optimal sense.
- Analogously to what done before, we define the following error function:
Mean-Square-Error (MSE)

$$\text{mse} = \mathbb{E} \left[\left\| \mathbf{X} - \hat{\mathbf{X}} \right\|_2^2 \right]$$

- We seek an estimator $\hat{\mathbf{X}}$ in the form of an affine function of the observation \mathbf{Y} , that is,

$$\hat{\mathbf{X}} = \mathbf{A}\mathbf{Y} + \mathbf{b}$$

Linear Minimum Mean-Square Error Estimation (LMMSE)



- First, notice that for any mean vectors \mathbf{m}_x and \mathbf{m}_y and any estimator $\hat{\mathbf{X}}$, we can always reduce the problem to a zero-mean case by considering $\mathbf{X}' = \mathbf{X} - \mathbf{m}_x$, $\mathbf{Y}' = \mathbf{Y} - \mathbf{m}_y$. If $\hat{\mathbf{X}}'$ is the minimum MSE (MMSE) estimator for \mathbf{X}' given \mathbf{Y}' , then

$$\hat{\mathbf{X}} = \mathbf{m}_x + \hat{\mathbf{X}}'$$

is the optimal estimator for \mathbf{X} given \mathbf{Y} .

- We restrict to the zero-mean case, and seek $\hat{\mathbf{X}}$ in the form $\mathbf{A}\mathbf{Y}$. The orthogonality principle yields the condition

$$(\mathbf{X} - \hat{\mathbf{X}}, \mathbf{Y}) = \mathbb{E} [(\mathbf{X} - \hat{\mathbf{X}})^\top \mathbf{Y}] = \text{tr} (\mathbb{E} [(\mathbf{X} - \hat{\mathbf{X}})\mathbf{Y}^\top]) = 0$$

that, explicitly, writes

$$\text{tr} (\mathbb{E} [\mathbf{XY}^\top] - \mathbf{A}\mathbb{E} [\mathbf{YY}^\top]) = 0$$

Linear Minimum Mean-Square Error Estimation (LMMSE)



- We can solve for \mathbf{A} as follows:

$$\mathbf{A}\mathbb{E}[\mathbf{Y}\mathbf{Y}^T] = \mathbb{E}[\mathbf{X}\mathbf{Y}^T] \Rightarrow \mathbf{A} = \mathbb{E}[\mathbf{X}\mathbf{Y}^T](\mathbb{E}[\mathbf{Y}\mathbf{Y}^T])^{-1}$$

- In conclusions, in the general case (non-zero mean), we let

$$\Sigma_{xy} = \text{cov}(\mathbf{X}, \mathbf{Y}) = \mathbb{E}[(\mathbf{X} - \mathbf{m}_x)(\mathbf{Y} - \mathbf{m}_y)^T], \quad \Sigma_y = \text{cov}(\mathbf{Y})$$

and we obtain the linear MMSE estimator (**Wiener filter**) of \mathbf{X} from \mathbf{Y} as

$$\hat{\mathbf{X}} = \mathbf{m}_x + \Sigma_{xy} \Sigma_y^{-1} (\mathbf{Y} - \mathbf{m}_y)$$

Linear Minimum Mean-Square Error Estimation (LMMSE)



- The error covariance matrix is given by

$$\begin{aligned}\text{cov}(\mathbf{X} - \hat{\mathbf{X}}) &= \mathbb{E}[(\mathbf{X} - \hat{\mathbf{X}})(\mathbf{X} - \hat{\mathbf{X}})^T] \\ &= \boldsymbol{\Sigma}_x - \boldsymbol{\Sigma}_{xy}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Sigma}_{xy}^T\end{aligned}$$

- The total MMSE, is given by $\mathbb{E}[\|\mathbf{X} - \hat{\mathbf{X}}\|_2^2] = \text{tr}(\text{cov}(\mathbf{X} - \hat{\mathbf{X}}))$.

Notice: The estimation error vector $\mathbf{X} - \hat{\mathbf{X}}$ is uncorrelated with the observation vector \mathbf{Y} .



Minimum Mean-Square Error Estimation (MMSE)

- With the same setting as before, we now seek an estimator $\hat{\mathbf{X}} = g(\mathbf{Y})$, in the space of all (measurable) functions of the observation \mathbf{Y} .

Theorem . *The MMSE estimator of \mathbf{X} given \mathbf{Y} is the conditional mean*

$$\hat{\mathbf{X}} = \mathbb{E}[\mathbf{X}|\mathbf{Y}]$$

Proof.

We use the orthogonality principle: the optimal estimator $\hat{\mathbf{X}}$ must satisfy

$$\mathbb{E} \left[(\mathbf{X} - \hat{\mathbf{X}})^T g(\mathbf{Y}) \right] = 0, \quad \text{for all functions } g$$

Let's check with the conditional mean:

$$\begin{aligned}\mathbb{E} [(\mathbf{X} - \mathbb{E}[\mathbf{X}|\mathbf{Y}])^T g(\mathbf{Y})] &= \mathbb{E} [\mathbb{E} [(\mathbf{X} - \mathbb{E}[\mathbf{X}|\mathbf{Y}])^T g(\mathbf{Y}) | \mathbf{Y}]] \\ &= \mathbb{E} [\mathbb{E} [\mathbf{X}^T g(\mathbf{Y}) | \mathbf{Y}] - \mathbb{E}[\mathbf{X}|\mathbf{Y}]^T g(\mathbf{Y})] \\ &= \mathbb{E} [\mathbb{E} [\mathbf{X}|\mathbf{Y}]^T g(\mathbf{Y}) - \mathbb{E}[\mathbf{X}|\mathbf{Y}]^T g(\mathbf{Y})] \\ &= 0\end{aligned}$$



The Gaussian case

- If \mathbf{X}, \mathbf{Y} are jointly Gaussian, then the linear MMSE estimator and the optimal MMSE estimator coincide.
- In order to see this, recall Theorem 14

$$f_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y}) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma_{x|y})}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{m}_{x|y})^\top \Sigma_{x|y}^{-1} (\mathbf{x} - \mathbf{m}_{x|y})\right)$$

where the conditional mean value is given by

$$\mathbf{m}_{x|y} = \mathbb{E}[\mathbf{X}|\mathbf{Y} = \mathbf{y}] = \mathbf{m}_x + \Sigma_{xy}\Sigma_y^{-1}(\mathbf{y} - \mathbf{m}_y)$$

and the conditional covariance matrix is given by

$$\Sigma_{x|y} = \mathbb{E}[(\mathbf{X} - \mathbf{m}_{x|y})(\mathbf{X} - \mathbf{m}_{x|y})^\top | \mathbf{Y} = \mathbf{y}] = \Sigma_x - \Sigma_{xy}\Sigma_y^{-1}\Sigma_{yx}$$



The Gaussian case

- Hence, the (general) MMSE estimator of \mathbf{X} given \mathbf{Y} coincides with the linear MMSE estimator (Wiener filter) in the Gaussian case:

$$\hat{\mathbf{X}} = \mathbb{E}[\mathbf{X}|\mathbf{Y}] = \mathbf{m}_x + \boldsymbol{\Sigma}_{xy}\boldsymbol{\Sigma}_y^{-1}(\mathbf{Y} - \mathbf{m}_y)$$

- MMSE decomposition:

$$\mathbf{X} = \hat{\mathbf{X}} + (\mathbf{X} - \hat{\mathbf{X}}) = \hat{\mathbf{X}} + \mathbf{V}$$

where the MMSE estimator $\hat{\mathbf{X}}$ and the estimation error vector \mathbf{V} are **independent**,

$$\hat{\mathbf{X}} \sim \mathcal{N}(\mathbf{m}_x, \boldsymbol{\Sigma}_{xy}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{\Sigma}_{yx}), \quad \mathbf{V} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{x|y})$$



Bayesian Estimation



Maximum likelihood

- parameter estimation in three steps:
 - 1) choose a parametric model for probabilities
to make this clear we denote the vector of parameters by Θ

$$P_X(x; \Theta)$$

note that this means that Θ is NOT a random variable

- 2) assemble $\mathcal{D} = \{x_1, \dots, x_n\}$ of examples drawn independently
- 3) select the parameters that maximize the probability of the data

$$\begin{aligned}\Theta^* &= \arg \max_{\Theta} P_X(D; \Theta) \\ &= \arg \max_{\Theta} \log P_X(D; \Theta)\end{aligned}$$

- $P_X(\mathcal{D}; \Theta)$ is the likelihood of parameter Θ with respect to the data



Bayesian parameter estimation

- the main difference with respect to ML is that in the Bayesian case Θ is a random variable
- basic concepts
 - training set $D = \{x_1, \dots, x_n\}$ of examples drawn independently
 - probability density for observations given parameter

$$P_{X|\Theta}(x | \theta)$$

- prior distribution for parameter configurations

$$P_\Theta(\theta)$$

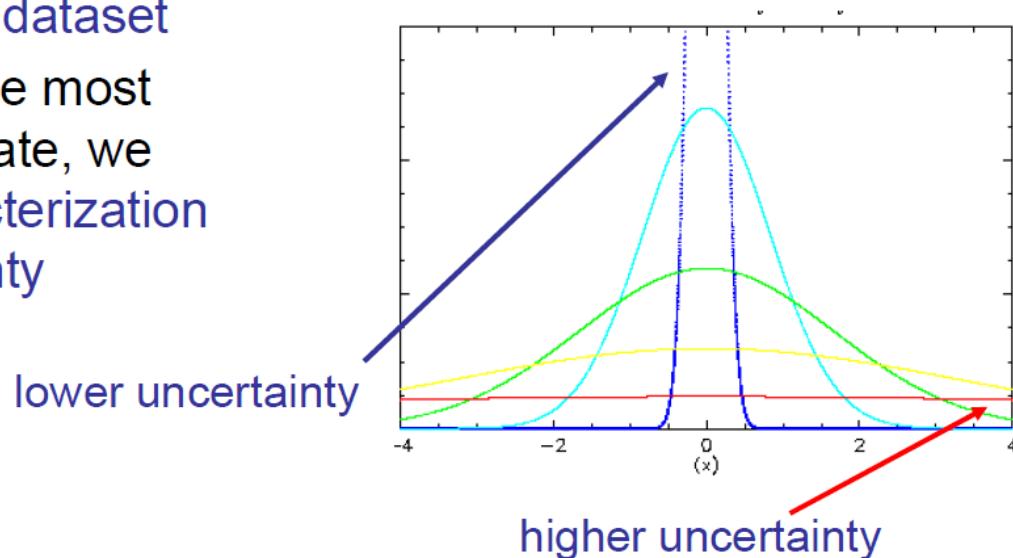
that encodes prior beliefs about them

- goal: to compute the posterior distribution

$$P_{\Theta|X}(\theta | D)$$

Bayes vs ML

- there are a number of significant differences between Bayesian and ML estimates
- D_1 :
 - ML produces a number, the best estimate
 - to measure its goodness we need to measure bias and variance
 - this can only be done with repeated experiments
 - Bayes produces a complete characterization of the parameter from the single dataset
 - in addition to the most probable estimate, we obtain a characterization of the uncertainty





Bayes vs ML

- D₂: optimal estimate
 - under ML there is one “best” estimate
 - under Bayes there is no “best” estimate
 - only a random variable that takes different values with different probabilities
 - technically speaking, it makes no sense to talk about the “best” estimate
- D₃: predictions
 - remember that we do not really care about the parameters themselves
 - they are needed only in the sense that they allow us to build models
 - that can be used to make predictions (e.g. the BDR)
 - unlike ML, Bayes uses ALL information in the training set to make predictions



Bayesian solution

- Gaussian likelihood (observations)

$$P_{T|\mu}(D | \mu) = G(D, \mu, \sigma^2) \quad \sigma^2 \text{ is known}$$

- Gaussian prior (what we know)

$$P_\mu(\mu) = G(\mu, \mu_0, \sigma_0^2)$$

- μ_0, σ_0^2 are known hyper-parameters
- we need to compute
 - posterior distribution for μ

$$P_{\mu|T}(\mu | D) = \frac{P_{T|\mu}(D | \mu)P_\mu(\mu)}{P_T(D)}$$



Bayesian solution

- posterior distribution

$$P_{\mu|T}(\mu | D) = \frac{P_{T|\mu}(D | \mu)P_\mu(\mu)}{P_T(D)}$$

- note that
 - this is a probability density
 - we can ignore constraints (terms that do not depend on μ)
 - and normalize when we are done
- we only need to work with

$$\begin{aligned} P_{\mu|T}(\mu | D) &\propto P_{T|\mu}(D | \mu)P_\mu(\mu) \\ &\propto \prod_i P_{X|\mu}(x_i | \mu)P_\mu(\mu) \end{aligned}$$



Bayesian solution

- plugging in the Gaussians

$$P_{\mu|T}(\mu | D) \propto \prod_i P_{X|\mu}(x_i | \mu) P_\mu(\mu)$$

$$\propto \prod_i G(x_i, \mu, \sigma^2) G(\mu, \mu_0, \sigma_0^2)$$

$$\propto \exp \left\{ - \sum_i \frac{(x_i - \mu)^2}{2\sigma^2} - \frac{(\mu - \mu_0)^2}{2\sigma_0^2} \right\}$$

$$\propto \exp \left\{ - \sum_i \frac{\mu^2 - 2x_i\mu + x_i^2}{2\sigma^2} - \frac{\mu^2 - 2\mu\mu_0 + \mu_0^2}{2\sigma_0^2} \right\}$$

$$\propto \exp \left\{ - \left(\frac{n}{2\sigma^2} + \frac{1}{2\sigma_0^2} \right) \mu^2 + 2 \left(\frac{\sum_i x_i}{2\sigma^2} + \frac{\mu_0}{2\sigma_0^2} \right) \mu - \left(\frac{\sum_i x_i^2}{2\sigma^2} + \frac{\mu_0^2}{2\sigma_0^2} \right) \right\}$$



Bayesian solution

$$P_{\mu|T}(\mu | D) \propto \exp \left\{ - \left(\frac{n}{2\sigma^2} + \frac{1}{2\sigma_0^2} \right) \mu^2 + 2 \left(\frac{\sum_i x_i}{2\sigma^2} + \frac{\mu_0}{2\sigma_0^2} \right) \mu \right\}$$

- this is a **Gaussian**, we just need to put it in the standard quadratic form to know its **mean** and **variance**
- use the **completing the squares** trick

$$ax^2 + 2bx + c = a \left(x^2 + 2 \frac{b}{a} x + \frac{c}{a} \right)$$

$$= a \left(x^2 + 2 \frac{b}{a} x + \left(\frac{b}{a} \right)^2 - \left(\frac{b}{a} \right)^2 + \frac{c}{a} \right) = a \left(x + \frac{b}{a} \right)^2 + c - \frac{b^2}{a}$$



Bayesian solution

$$P_{\mu|T}(\mu | D) \propto \exp \left\{ - \left(\frac{n}{2\sigma^2} + \frac{1}{2\sigma_0^2} \right) \mu^2 + 2 \left(\frac{\sum x_i}{2\sigma^2} + \frac{\mu_0}{2\sigma_0^2} \right) \mu \right\}$$

- in this case

$$ax^2 + 2bx + c = a \left(x + \frac{b}{a} \right)^2 + c - \frac{b^2}{a} \propto a \left(x + \frac{b}{a} \right)^2$$

- we have

$$P_{\mu|T}(\mu | D) \propto \exp \left\{ - \left(\frac{n}{2\sigma^2} + \frac{1}{2\sigma_0^2} \right) \left[\mu - \sqrt{\left(\frac{\sum x_i}{2\sigma^2} + \frac{\mu_0}{2\sigma_0^2} \right)} \right]^2 \right\}$$



Bayesian solution

- and using

$$1/\left(\frac{n}{2\sigma^2} + \frac{1}{2\sigma_0^2}\right) = \frac{2\sigma^2\sigma_0^2}{(\sigma^2 + n\sigma_0^2)}$$

- we have

$$\begin{aligned} P_{\mu|T}(\mu | D) &\propto \exp \left\{ -\left(\frac{n}{2\sigma^2} + \frac{1}{2\sigma_0^2} \right) \left[\mu - \left(\frac{2\sigma^2\sigma_0^2}{\sigma^2 + n\sigma_0^2} \right) \left(\frac{\sigma_0^2 \sum_i x_i + \mu_0 \sigma^2}{2\sigma^2\sigma_0^2} \right) \right]^2 \right\} \\ &\propto \exp \left\{ -\left(\frac{2\sigma^2\sigma_0^2}{\sigma^2 + n\sigma_0^2} \right)^{-1} \left[\mu - \left(\frac{\sigma_0^2 \sum_i x_i + \mu_0 \sigma^2}{\sigma^2 + n\sigma_0^2} \right) \right]^2 \right\} \end{aligned}$$

- and

$$P_{\mu|T}(\mu | D) = G(\mu, \mu_n, \sigma_n^2), \quad \mu_n = \frac{\sigma_0^2 \sum_i x_i + \mu_0 \sigma^2}{\sigma^2 + n\sigma_0^2}, \quad \sigma_n^2 = \left(\frac{\sigma^2 \sigma_0^2}{\sigma^2 + n\sigma_0^2} \right)$$



Bayesian solution

- we had

$$\mu_n = \alpha_n \hat{\mu} + (1 - \alpha_n) \mu_0$$

$$\alpha_n \in [0,1], \quad \alpha_n \xrightarrow{n \rightarrow \infty} 1, \quad \alpha_n \xrightarrow{n \rightarrow 0} 0$$

- the Bayesian solution is

$$\mu_n = \underbrace{\frac{n\sigma_0^2}{\sigma^2 + n\sigma_0^2} \mu_{ML}}_{\alpha_n} + \underbrace{\frac{\sigma^2}{\sigma^2 + n\sigma_0^2} \mu_0}_{1 - \alpha_n}$$

- note that

$$\alpha_n \in [0,1], \quad \alpha_n \xrightarrow{n \rightarrow \infty} 1, \quad \alpha_n \xrightarrow{n \rightarrow 0} 0$$

- this should be read as

$$P_{\text{Bayes}} = P_{\text{ML}} + P_{\text{prior}}$$

- Bayesian precision is greater than both that of ML and prior



Conjugate priors

► note that

- the prior $P_\mu(\mu) = G(\mu, \mu_0, \sigma_0^2)$ is Gaussian
- the posterior $P_{\mu|T}(\mu | D) = G(x, \mu_n, \sigma_n^2)$ is Gaussian

► whenever this is the case (posterior in the same family as prior) we say that

- $P_\mu(\mu)$ is a **conjugate prior** for the likelihood $P_{X|\mu}(X | \mu)$
- posterior $P_{\mu|T}(\mu | D)$ is the **reproducing density**

► HW: a number of likelihoods have conjugate priors

Likelihood	Conjugate prior
Bernoulli	Beta
Poisson	Gamma
Exponential	Gamma
Normal (known σ^2)	Gamma



Expectation Maximization



Expectation-maximization

- ▶ we have seen that EM is a framework for ML estimation with missing data
- ▶ i.e. problems where we have, two types of random variables
 - X observed random variable
 - Z hidden random variable
- ▶ goal:
 - given iid sample $D = \{x_1, \dots, x_n\}$
 - find parameters Ψ^* that maximize likelihood with respect to D

$$\begin{aligned}\Psi^* &= \arg \max_{\Psi} P_{\mathbf{X}}(\mathcal{D}; \Psi) \\ &= \arg \max_{\Psi} \int P_{\mathbf{X}|Z}(\mathcal{D}|z; \Psi) P_Z(z; \Psi) dz\end{aligned}$$



Expectation-maximization

- ▶ the set

$$D = \{x_1, \dots, x_n\}$$

is called the incomplete data

- ▶ the set

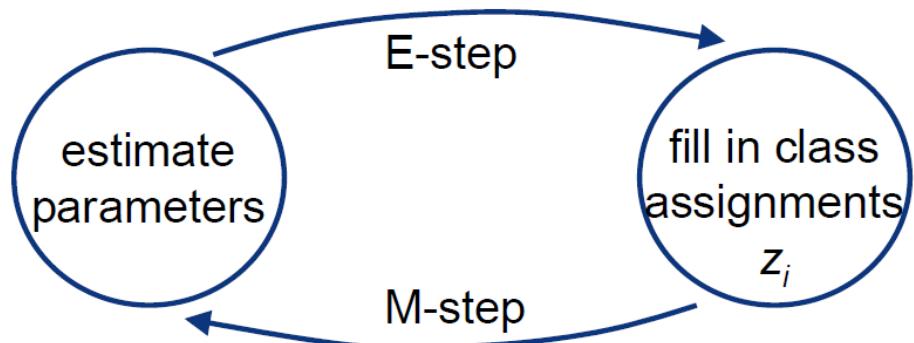
$$D_c = \{(x_1, z_1), \dots, (x_n, z_n)\}$$

is called the complete data

- ▶ we never get to see it, otherwise the problem would be trivial (standard ML)
- ▶ EM solves the problem by iterating between two steps

Expectation-maximization

- ▶ the basic idea is quite simple
 1. start with an initial parameter estimate $\Psi^{(0)}$
 2. **E-step:** given current parameters $\Psi^{(i)}$ and observations in D , “guess” what the values of the z_i are
 3. **M-step:** with the new z_i , we have a complete data problem, solve this problem for the parameters, i.e. compute $\Psi^{(i+1)}$
 4. go to 2.
- ▶ this can be summarized as





The Q function

- ▶ main idea: don't know what complete data likelihood is, but can compute its expected value given observed data
- ▶ this is the Q function

$$Q(\Psi; \Psi^{(n)}) = E_{Z|X; \Psi^{(n)}} [\log P_{X,Z}(\mathcal{D}, \{z_1, \dots, z_N\}; \Psi) | \mathcal{D}]$$

- ▶ and is a bit tricky:
 - it is the expected value of likelihood with respect to complete data (joint X and Z)
 - given that we observed incomplete data (X)
 - note that the likelihood is a function of Ψ (the parameters that we want to determine)
 - but to compute the expected value we need to use the parameter values from the previous iteration (because we need a distribution for Z|X)



Expectation-maximization

► E-step:

- given estimates $\Psi^{(n)} = \{\Psi^{(n)}_1, \dots, \Psi^{(n)}_C\}$
- compute expected log-likelihood of complete data

$$Q(\Psi; \Psi^{(n)}) = E_{Z|X; \Psi^{(n)}} [\log P_{X,Z}(\mathcal{D}, \{z_1, \dots, z_N\}; \Psi) | \mathcal{D}]$$

► M-step:

- find parameter set that maximizes this expected log-likelihood

$$\Psi^{(n+1)} = \arg \max_{\Psi} Q(\Psi; \Psi^{(n)})$$

► let's make this more concrete by looking at a toy example



EM convergence

- ▶ we are now ready to show that EM converges
- ▶ recall: the goal is to maximize $\log P_{\mathbf{X}}(\mathcal{D}; \Psi)$
- ▶ using

$$P_{\mathbf{X}, \mathbf{Z}}(\mathcal{D}, \mathbf{z}; \Psi) = P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi)P_{\mathbf{X}}(\mathcal{D}; \Psi)$$

- ▶ this can be written as

$$\log P_{\mathbf{X}}(\mathcal{D}; \Psi) = \log P_{\mathbf{X}, \mathbf{Z}}(\mathcal{D}, \mathbf{z}; \Psi) - \log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi)$$

- ▶ taking expectations on both sides and using the fact that the LHS does not depend on Z

$$\begin{aligned}\log P_{\mathbf{X}}(\mathcal{D}; \Psi) &= E_{\mathbf{Z}|\mathbf{X}; \Psi^{(n)}}[\log P_{\mathbf{X}, \mathbf{Z}}(\mathcal{D}, \mathbf{z}; \Psi)|\mathcal{D}] \\ &\quad - E_{\mathbf{Z}|\mathbf{X}; \Psi^{(n)}}[\log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi)|\mathcal{D}]\end{aligned}$$



EM convergence

- and plugging in the definition of the Q function

$$\begin{aligned}\log P_{\mathbf{X}}(\mathcal{D}; \Psi) &= E_{\mathbf{Z}|\mathbf{X}; \Psi^{(n)}}[\log P_{\mathbf{X}, \mathbf{Z}}(\mathcal{D}, \mathbf{z}; \Psi)|\mathcal{D}] \\ &\quad - E_{\mathbf{Z}|\mathbf{X}; \Psi^{(n)}}[\log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi)|\mathcal{D}] \\ &= Q(\Psi|\Psi^{(n)}) + H(\Psi|\Psi^{(n)})\end{aligned}$$

- where we have also introduced

$$\begin{aligned}H(\Psi|\Psi^{(n)}) &= -E_{\mathbf{Z}|\mathbf{X}; \Psi^{(n)}}[\log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi)|\mathcal{D}] \\ &= - \int P_{\mathbf{Z}|\mathbf{X}; \Psi^{(n)}}(\mathbf{z}|\mathcal{D}; \Psi^{(n)}) \log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi) d\mathbf{z}\end{aligned}$$



EM convergence

- ▶ the **key** to proving convergence is this equation

$$\log P_{\mathbf{X}}(\mathcal{D}; \Psi) = Q(\Psi | \Psi^{(i)}) + H(\Psi | \Psi^{(i)})$$

- ▶ note, in particular, that

$$\begin{aligned} & \log P_{\mathbf{X}}(\mathcal{D}; \Psi^{(n+1)}) - \log P_{\mathbf{X}}(\mathcal{D}; \Psi^{(n)}) = \\ &= Q(\Psi^{(n+1)} | \Psi^{(n)}) + H(\Psi^{(n+1)} | \Psi^{(n)}) \\ &\quad - [Q(\Psi^{(n)} | \Psi^{(n)}) + H(\Psi^{(n)} | \Psi^{(n)})] \\ &= Q(\Psi^{(n+1)} | \Psi^{(n)}) - Q(\Psi^{(n)} | \Psi^{(n)}) \\ &\quad + H(\Psi^{(n+1)} | \Psi^{(n)}) - H(\Psi^{(n)} | \Psi^{(n)}) \end{aligned}$$



EM convergence

- ▶ but, by definition of the M-step

$$\Psi^{(n+1)} = \arg \max_{\Psi} Q(\Psi | \Psi^{(n)})$$

- ▶ it follows that

$$Q(\Psi^{(n+1)} | \Psi^{(n)}) \geq Q(\Psi^{(n)} | \Psi^{(n)})$$

- ▶ and since

$$\begin{aligned} & \log P_{\mathbf{X}}(\mathcal{D}; \Psi^{(n+1)}) - \log P_{\mathbf{X}}(\mathcal{D}; \Psi^{(n)}) = \\ &= Q(\Psi^{(n+1)} | \Psi^{(n)}) - Q(\Psi^{(n)} | \Psi^{(n)}) \\ &\quad + H(\Psi^{(n+1)} | \Psi^{(n)}) - H(\Psi^{(n)} | \Psi^{(n)}) \end{aligned}$$

we have

$$\log P_{\mathbf{X}}(\mathcal{D}; \Psi^{(n+1)}) \geq \log P_{\mathbf{X}}(\mathcal{D}; \Psi^{(n)})$$



EM convergence

- ▶ we have

$$\log P_{\mathbf{X}}(\mathcal{D}; \boldsymbol{\Psi}^{(n+1)}) \geq \log P_{\mathbf{X}}(\mathcal{D}; \boldsymbol{\Psi}^{(n)})$$

- ▶ if

$$H(\boldsymbol{\Psi}^{(n+1)} | \boldsymbol{\Psi}^{(n)}) \geq H(\boldsymbol{\Psi}^{(n)} | \boldsymbol{\Psi}^{(n)})$$

- ▶ but, from

$$H(\boldsymbol{\Psi} | \boldsymbol{\Psi}^{(n)}) = -E_{\mathbf{Z}|\mathbf{X};\boldsymbol{\Psi}^{(n)}}[\log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \boldsymbol{\Psi}) | \mathcal{D}]$$

we have

$$\begin{aligned} & H(\boldsymbol{\Psi}^{(n+1)} | \boldsymbol{\Psi}^{(n)}) - H(\boldsymbol{\Psi}^{(n)} | \boldsymbol{\Psi}^{(n)}) \\ &= -E_{\mathbf{Z}|\mathbf{X};\boldsymbol{\Psi}^{(n)}} \left[\log \frac{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \boldsymbol{\Psi}^{(n+1)})}{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \boldsymbol{\Psi}^{(n)})} \Big| \mathcal{D} \right] \end{aligned}$$



EM convergence

- ▶ and, since the log is a concave function, by Jensen's
 $E[f(x)] \leq f(E[x])$
- ▶
$$\begin{aligned} H(\Psi^{(n+1)} | \Psi^{(n)}) - H(\Psi^{(n)} | \Psi^{(n)}) \\ = -E_{\mathbf{Z}|\mathbf{X};\Psi^{(n)}} \left[\log \frac{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi^{(n+1)})}{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi^{(n)})} \Big| \mathcal{D} \right] \\ \geq -\log E_{\mathbf{Z}|\mathbf{X};\Psi^{(n)}} \left[\frac{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi^{(n+1)})}{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi^{(n)})} \Big| \mathcal{D} \right] \\ = -\log \int P_{\mathbf{Z}|\mathbf{X};\Psi^{(n)}}(\mathbf{z}|\mathcal{D}; \Psi^{(n)}) \frac{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi^{(n+1)})}{P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi^{(n)})} d\mathbf{z} \\ = -\log 1 = 0 \end{aligned}$$



EM convergence

- ▶ this shows that

$$\log P_{\mathbf{X}}(\mathcal{D}; \boldsymbol{\Psi}^{(n+1)}) \geq \log P_{\mathbf{X}}(\mathcal{D}; \boldsymbol{\Psi}^{(n)})$$

- ▶ i.e. the log-likelihood of the incomplete data can only increase from iteration to iteration
- ▶ hence the algorithm converges
- ▶ note that there is no guarantee of convergence to a global minimum, only local



Geometric interpretation

- ▶ one can also derive a geometric interpretation from

$$\log P_{\mathbf{X}}(\mathcal{D}; \Psi) = Q(\Psi | \Psi^{(n)}) + H(\Psi | \Psi^{(n)})$$

- ▶ by noting that

$$\begin{aligned} H(\Psi | \Psi^{(n)}) &= -E_{\mathbf{Z}|\mathbf{X};\Psi^{(n)}}[\log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi)|\mathcal{D}] \\ &= -\int P_{\mathbf{Z}|\mathbf{X};\Psi^{(n)}}(\mathbf{z}|\mathcal{D}; \Psi^{(n)}) \log P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z}|\mathcal{D}; \Psi) d\mathbf{z} \end{aligned}$$

- ▶ is of the form

$$\begin{aligned} H(\Psi | \Psi^{(n)}) &= -\int p_n(\mathbf{z}) \log p(\mathbf{z}) d\mathbf{z} \\ &= \int p_n(\mathbf{z}) \log \frac{p_n(\mathbf{z})}{p(\mathbf{z})} d\mathbf{z} - \int p_n(\mathbf{z}) \log p_n(\mathbf{z}) d\mathbf{z} \end{aligned}$$



Geometric interpretation

- ▶ is of the form

$$\begin{aligned} H(\Psi|\Psi^{(n)}) &= - \int p_n(\mathbf{z}) \log p(\mathbf{z}) d\mathbf{z} \\ &= \int p_n(\mathbf{z}) \log \frac{p_n(\mathbf{z})}{p(\mathbf{z})} d\mathbf{z} - \int p_n(\mathbf{z}) \log p_n(\mathbf{z}) d\mathbf{z} \\ &= KL[p_n||p] + H[p_n] \end{aligned}$$

- ▶ where $KL[p||q]$ is the Kullback-Leibler divergence between p and q , and $H[p]$ the entropy of p
- ▶ it can be shown that these two quantities are never negative, from which $H(\Psi|\Psi^{(n)}) \geq 0$ and
- ▶ since

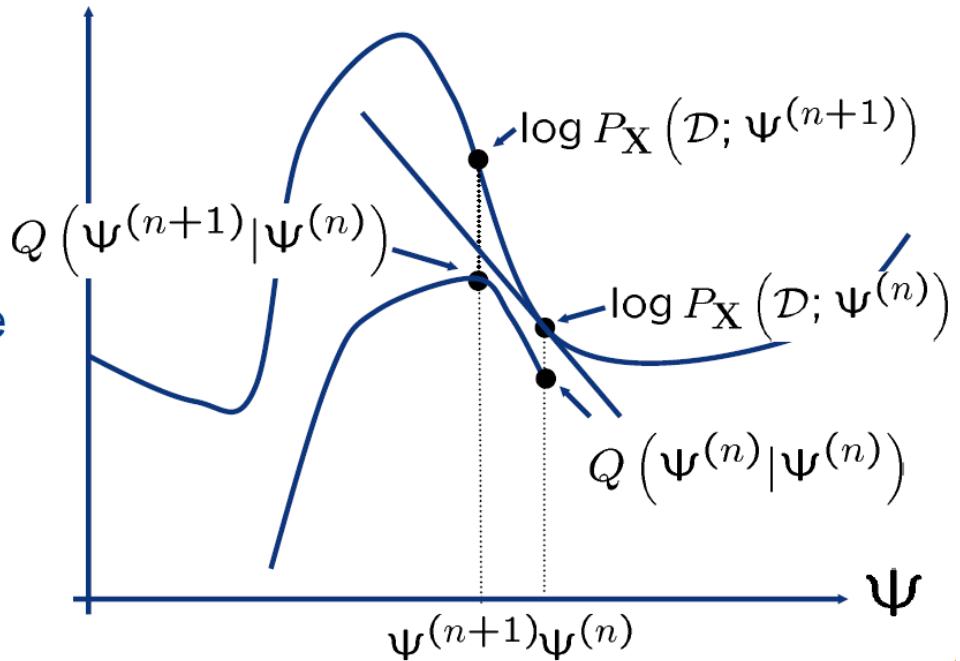
$$\log P_{\mathbf{X}}(\mathcal{D}; \Psi) = Q(\Psi|\Psi^{(n)}) + H(\Psi|\Psi^{(n)})$$

Geometric interpretation

- ▶ we have

$$\log P_X(\mathcal{D}; \Psi) \geq Q(\Psi | \Psi^{(n)})$$

- ▶ which means that the Q function is a lower bound to the log-likelihood of the observed data
- ▶ this allows an interpretation of the EM steps as
 - E-step: lower-bound the observed log-likelihood
 - M-step: maximize the lower bound





Geometric interpretation

- consider next the difference between cost and bound

$$\log P_{\mathbf{X}}(\mathcal{D}; \Psi) - Q(\Psi | \Psi^{(n)}) = H(\Psi | \Psi^{(n)})$$

- which can be written as

$$H(\Psi | \Psi^{(n)}) = KL[p_n || p] + H[p_n]$$

with

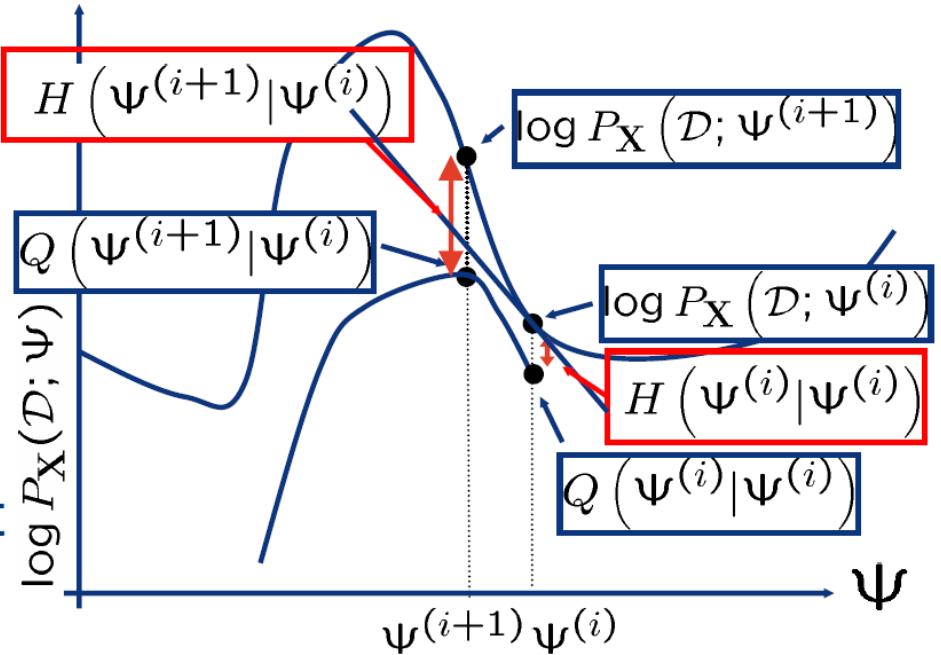
$$p_n(\mathbf{z}) = P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z} | \mathcal{D}; \Psi^{(n)}) \quad p(\mathbf{z}) = P_{\mathbf{Z}|\mathbf{X}}(\mathbf{z} | \mathcal{D}; \Psi)$$

- hence

$$\begin{aligned} H(\Psi^{(n+1)} | \Psi^{(n)}) - H(\Psi^{(n)} | \Psi^{(n)}) &= \\ &= KL[p_n || p_{n+1}] + H[p_n] - KL[p_n || p_n] - H[p_n] \\ &= KL[p_n || p_{n+1}] \geq 0 \end{aligned}$$

Geometric interpretation

- ▶ note that since
 - by definition of M-step: $Q(\Psi^{(n+1)}|\Psi^{(n)}) \geq Q(\Psi^{(n)}|\Psi^{(n)})$
 - by non-negativity of KL: $H(\Psi^{(n+1)}|\Psi^{(n)}) \geq H(\Psi^{(n)}|\Psi^{(n)})$
- ▶ it follows that $\log P_X(\mathcal{D}; \Psi^{(n+1)}) \geq \log P_X(\mathcal{D}; \Psi^{(n)})$
- ▶ EM converges without need for step sizes
- ▶ this is not the case for gradient ascent which uses the linear approximation
- ▶ if we move too far, there will be overshoot





MAP parameter estimates

- ▶ so far we have concentrated on ML estimation
- ▶ EM can be equally applied to obtain MAP estimates, with a straightforward extension
- ▶ recall that for MAP the goal is

$$\begin{aligned}\Psi^* &= \arg \max_{\Psi} P_{\Psi|X}(\Psi|\mathcal{D}) \\ &= \arg \max_{\Psi} P_{X|\Psi}(\mathcal{D}|\Psi)P_{\Psi}(\Psi)\end{aligned}$$

- ▶ this is not very different from ML, we just multiply by $P_{\Psi}(\Psi)$
- ▶ still a problem of estimation from incomplete data, with

$$P_{X|\Psi}(\mathcal{D}|\Psi) = \int P_{X|Z,\Psi}(\mathcal{D}|z, \Psi)P_{Z|\Psi}(z|\Psi)dz$$



MAP parameter estimates

- and there is a complete data posterior

$$P_{\Psi|X,Z}(\Psi|\mathcal{D}, z)$$

- the E step is now to compute

$$\begin{aligned} E_{Z|X,\Psi}[\log P_{\Psi|X,Z}(\Psi|\mathcal{D}, z)|\mathcal{D}, \Psi^{(n)}] &= \\ &= E_{Z|X,\Psi}[\log P_{X,Z|\Psi}(\mathcal{D}, z|\Psi)|\mathcal{D}, \Psi^{(n)}] + \\ &\quad + E_{Z|X,\Psi}[\log P_\Psi(\Psi)|\mathcal{D}, \Psi^{(n)}] - \\ &\quad - E_{Z|X,\Psi}[\log P_{X,Z}(\mathcal{D}, z)|\mathcal{D}, \Psi^{(n)}] \\ &= Q(\Psi|\Psi^{(n)}) + \log P_\Psi(\Psi) - \\ &\quad - E_{Z|X,\Psi}[\log P_{X,Z}(\mathcal{D}, z)|\mathcal{D}, \Psi^{(n)}] \end{aligned}$$

- note that the last term does not depend on Ψ
- does not affect M-step, we can drop it



MAP parameter estimates

- ▶ hence the E-step does not really change

- ▶ E step: compute

$$Q(\Psi|\Psi^{(n)}) = E_{\mathbf{Z}|\mathbf{X},\Psi}[\log P_{\mathbf{X},\mathbf{Z}|\Psi}(\mathcal{D}, \mathbf{z}|\Psi)|\mathcal{D}, \Psi^{(n)}]$$

- ▶ and the M-step becomes

$$\Psi^{(n+1)} = \arg \max_{\Psi} \left\{ Q(\Psi|\Psi^{(n)}) + \log P_{\Psi}(\Psi) \right\}$$

- ▶ this is the MAP-EM algorithm

- ▶ note that M-step looks like a standard Bayesian estimate procedure, and typically is

- ▶ e.g. for mixtures, it is equivalent to computing Bayesian estimates for each component, under “soft-assignments”

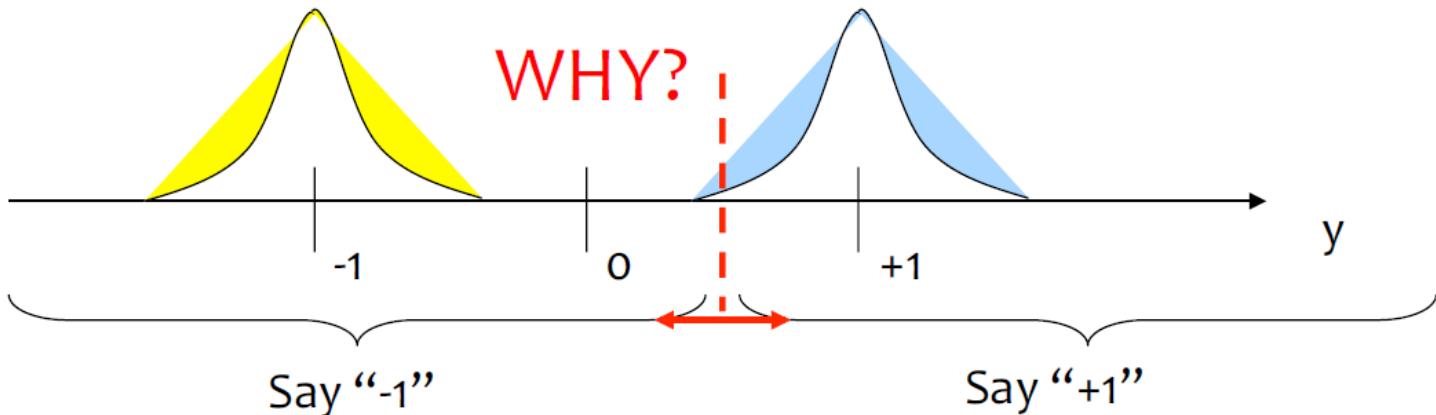


Factor Graph and Message Passing

F. R. Kschischang, B. J. Frey and H. -. Loeliger, "Factor graphs and the sum-product algorithm," in IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 498-519, Feb 2001, doi: 10.1109/18.910572.

Optimal detection

- Parameter to be estimated $\mathbf{b} \in \mathcal{B}$
 - known prior $p_B(\mathbf{b})$
- Observation $y \in \mathcal{Y}$
 - known $p_{Y|B}(y|\mathbf{b})$
- Estimator/detector $b^*(y): \mathcal{Y} \rightarrow \mathcal{B}$
 - For every possible y , associate an estimate
 - Designed to minimize a cost
- Example
 - $\mathbf{b} \in \{-1, +1\}$, $p_B(+1)=0.1$, $p_B(-1)=0.9$
 - $y=b+n$, $n \sim N_n(0,1) \Rightarrow p_{Y|B}(y|\mathbf{b}) = N_y(b, 1)$





MAP detector is optimal

- MAP detector is **optimal** in the following sense: among all detectors, the MAP detector minimizes the probability of making an error
- Two variations
 - Minimizing the error rate of the packet \mathbf{b} (FER)
$$\mathbf{b}^*(\mathbf{y}) = \arg \max_{\mathbf{b}} p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}|\mathbf{y})$$
 - Minimizing the error rate of the individual bits (BER)
$$b_k^*(\mathbf{y}) = \arg \max_{b_k \in \{0,1\}} p_{B_k|\mathbf{Y}}(b_k|\mathbf{y}), \forall k$$

Why two MAP detectors?

Problem

- Maximizing[‡] over all \mathbf{b} : complexity exponential in length of \mathbf{b}

$$\mathbf{b}^*(\mathbf{y}) = \arg \max_{\mathbf{b}} p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}|\mathbf{y})$$

Solution

- Minimize BER instead of FER

$$b_k^*(\mathbf{y}) = \arg \max_{b_k \in \{0,1\}} p_{B_k|\mathbf{Y}}(b_k|\mathbf{y}), \forall k$$

Problem

- It is hard to find $p(b_k | \mathbf{y})$

Solution

- How to find $p(b_k | \mathbf{y})$ is one of the topics of this tutorial
- $p(b_k | \mathbf{y})$ is **marginal** of $p(\mathbf{b} | \mathbf{y})$
- Factor graphs can help in computing marginals





How about continuous variables?

- Suppose $\mathbf{b} \in \mathbb{C}^N$
- Terminology: “detection” \rightarrow “estimation”
- Different cost functions lead to different estimators
- **MAP estimator** (for $\delta \rightarrow 0$)

$$c(\mathbf{b}, \mathbf{b}^*(\mathbf{y})) = \begin{cases} 0 & \|\mathbf{b} - \mathbf{b}^*(\mathbf{y})\| < \delta \\ 1 & \text{else} \end{cases}$$

$$\mathbf{b}^*(\mathbf{y}) = \arg \max_{\mathbf{b}} p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}|\mathbf{y})$$

- **MMSE estimator**

$$c(\mathbf{b}, \mathbf{b}^*(\mathbf{y})) = \|\mathbf{b} - \mathbf{b}^*(\mathbf{y})\|^2$$

$$\mathbf{b}^*(\mathbf{y}) = \int \mathbf{b} p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}|\mathbf{y}) d\mathbf{b}$$

Motivation

- Three important problems

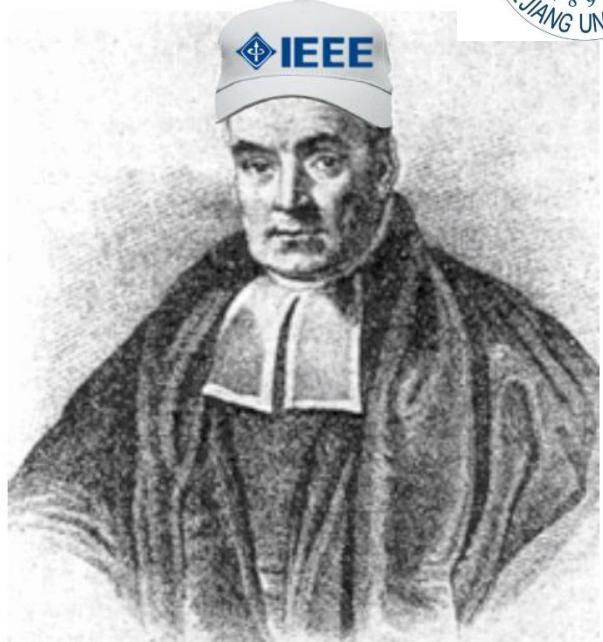
- MAP configuration

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$$

- Marginal posteriors

$$p(x_i|\mathbf{y}) = \sum_{\sim x_i} p(\mathbf{x}|\mathbf{y})$$

- Probability of \mathbf{y}
 $p(\mathbf{y})$



- Complexity exponential in dimension of \mathbf{x}
- Complexity can be reduced by exploiting conditional independence
- Bayesian graphical models is a tool to do this **systematically**



Example

- Consider the following distribution over binary variables

$$p(x_1, x_2, x_3, x_4, \mathbf{y}) = \prod_{i=1}^4 p(x_i) \times \prod_{i=2}^4 p(y_i|x_1, x_i)$$

- Solve the three problems

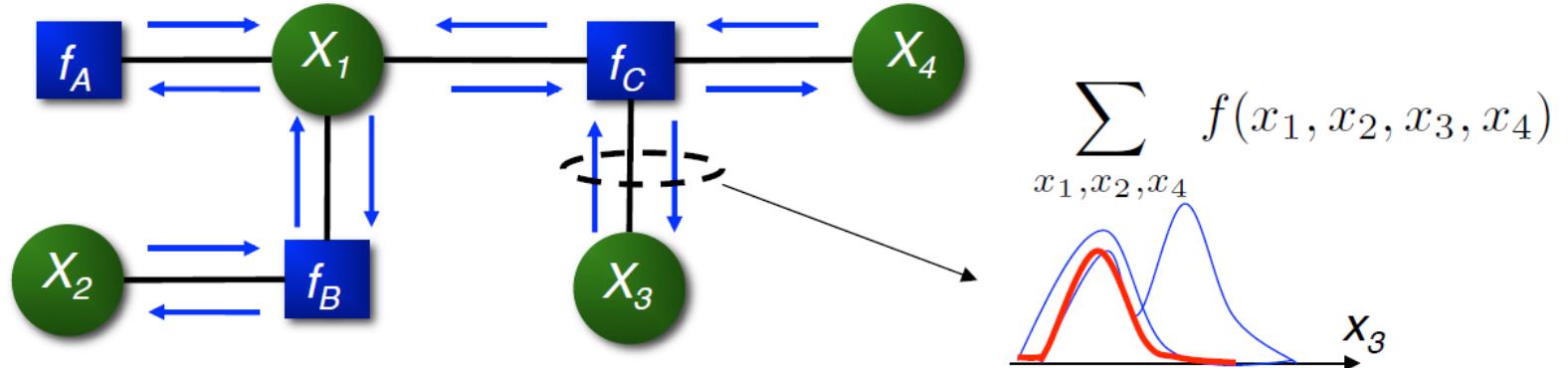
$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} p(\mathbf{x}, \mathbf{y})$$

$$\hat{x}_i = \arg \max_{x_i} \underbrace{\max_{\sim x_i} p(\mathbf{x}, \mathbf{y})}_{g(x_i)}$$

$$p(x_i, \mathbf{y}) = \sum_{\sim x_i} p(\mathbf{x}, \mathbf{y})$$

Factor graphs: high level

- Factorization $f(x_1, x_2, x_3, x_4) = f_A(x_1)f_B(x_1, x_2)f_C(x_1, x_3, x_4) \geq 0$



- The sum-product algorithm
 - Variable vertex to factor vertex

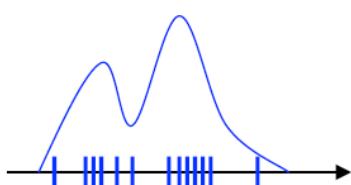
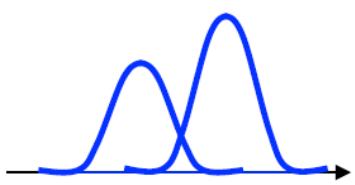
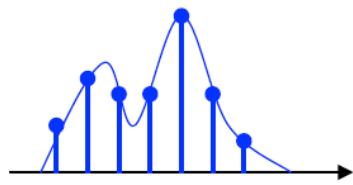
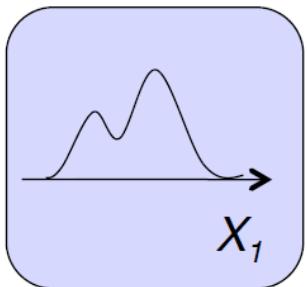
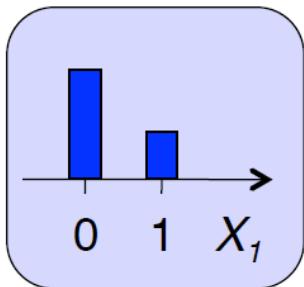
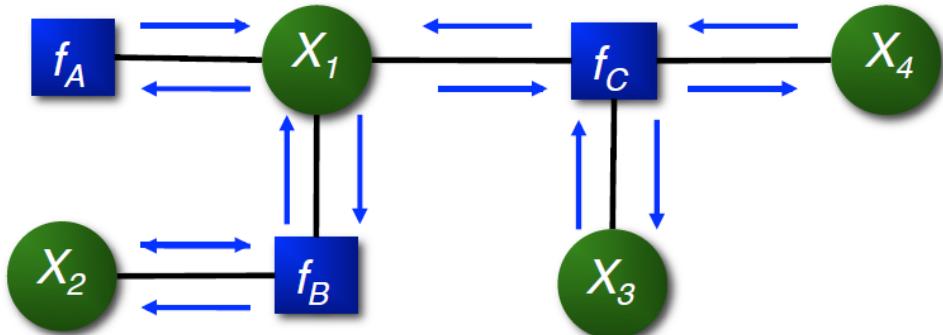
$$\mu_{X_1 \rightarrow f_C}(x_1) = \mu_{f_A \rightarrow X_1}(x_1) \times \mu_{f_B \rightarrow X_1}(x_1)$$
 - Factor vertex to variable vertex

$$\mu_{f_C \rightarrow X_1}(x_1) = \sum_{x_3, x_4} f_C(x_1, x_3, x_4) \times \mu_{X_3 \rightarrow f_C}(x_3) \times \mu_{X_4 \rightarrow f_C}(x_4)$$
- Marginals

$$\begin{aligned} g_{X_3}(x_3) &= \mu_{f_C \rightarrow X_3}(x_3) \times \mu_{X_3 \rightarrow f_C}(x_3) \\ &= \sum_{x_1, x_2, x_4} f(x_1, x_2, x_3, x_4) \end{aligned}$$

Factor graphs: high level

- Messages are **functions**



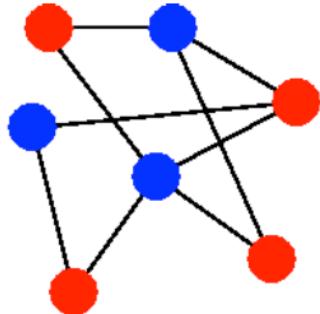
- Messages are often **normalized and transformed** (e.g., to LLR)
- When factor graph has **cycles**, SPA may fail
- Replacing “sum” with “max” yields max-marginals (MPA)

$$\sum_{x_1, x_2, x_4} f(x_1, x_2, x_3, x_4) \longrightarrow \max_{x_1, x_2, x_4} f(x_1, x_2, x_3, x_4)$$

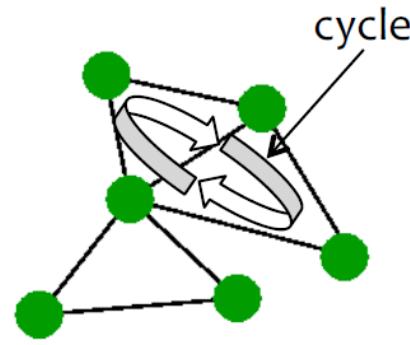
What is a factor graph?

- **Graphs**

- Graph $G=(V,E)$
- Vertices, edges
- Bipartite graph



A) A Bipartite Graph



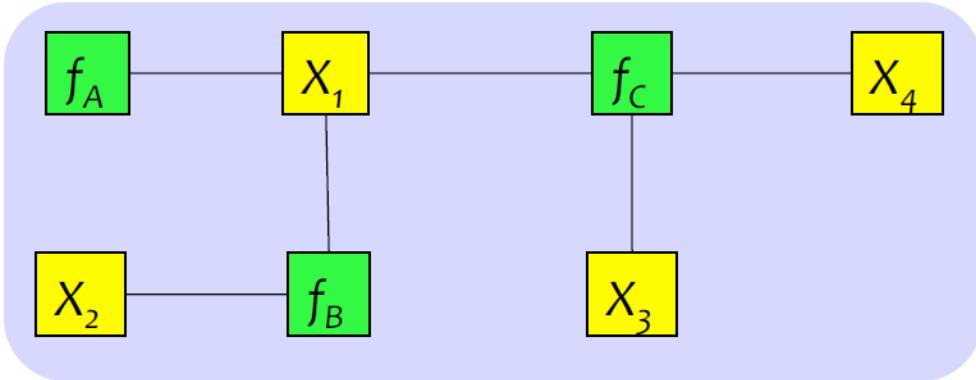
B) A non-Bipartite Graph

- **Factor graph**

A non-unique graph that represents the factorization of a real-valued function $f(\mathbf{x})$

Factor graph: construction

- Factorization $f(x_1, x_2, x_3, x_4) = f_A(x_1) f_B(x_1, x_2) f_C(x_1, x_3, x_4)$
- Factor graph



Construction

1. Vertex for every variable x_i (labeled with X_i)
2. Vertex for every factor f_k (labeled with f_k)
3. Edge between vertex x_i and vertex f_k when x_i appears in f_k

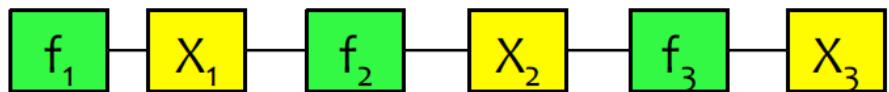


Example

1. Draw the FG of the following distribution

$$p(x_1, x_2, x_3, x_4, \mathbf{y}) = \prod_{i=1}^4 p(x_i) \times \prod_{i=2}^4 p(\mathbf{y}_i | x_1, x_i)$$

2. Write down a function corresponding to the following FG

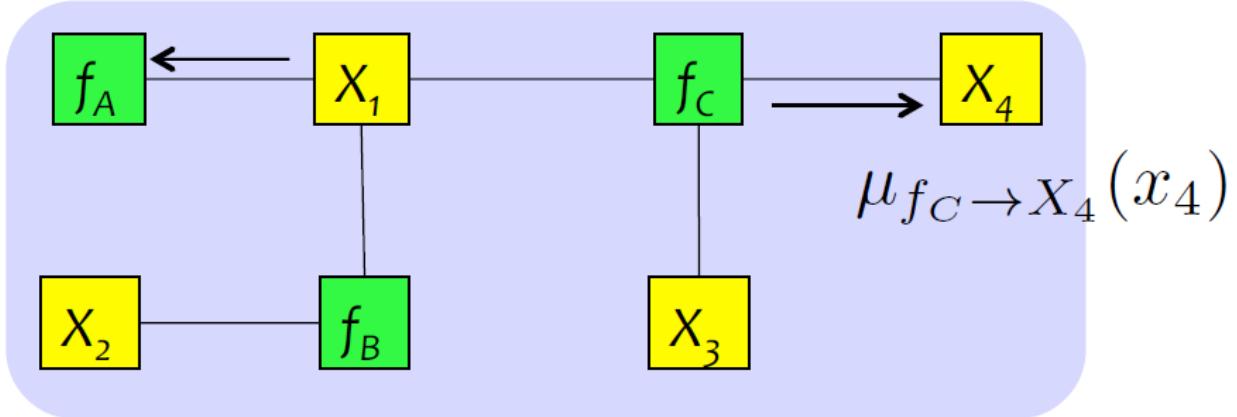


3. Can you come up with a reasonable distribution?

Message passing on factor graphs

- Messages are functions

$$\mu_{X_1 \rightarrow f_A}(x_1)$$



- If X_4 is binary and X_1 ternary

$$\mu_{X_1 \rightarrow f_A}(\cdot) = [\mu_{X_1 \rightarrow f_A}(-1) \quad \mu_{X_1 \rightarrow f_A}(0) \quad \mu_{X_1 \rightarrow f_A}(+1)]$$

$$\mu_{f_C \rightarrow X_4}(\cdot) = [\mu_{f_C \rightarrow X_4}(0) \quad \mu_{f_C \rightarrow X_4}(1)]$$



Sum-product and max-product algorithms

- There are many ways to compute messages
 - MAP configuration: max-product algorithm (MPA)
 - Marginal distributions: sum-product algorithm (SPA)
 - Normalization constant: sum-product algorithm

- Sum-marginal of a function $f(\mathbf{x})$

$$g_{X_k}^{\text{sum}}(x_k) = \sum_{\sim\{x_k\}} f(\mathbf{x})$$

- Max-marginal of a function $f(\mathbf{x})$

$$g_{X_k}^{\text{max}}(x_k) = \max_{\sim\{x_k\}} f(\mathbf{x})$$

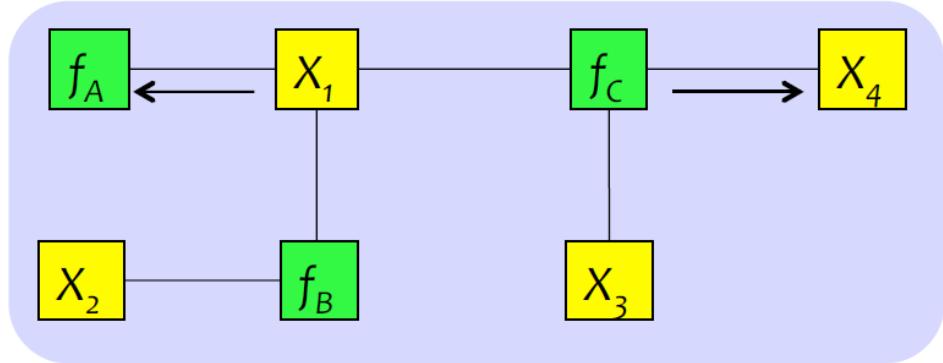
When $f(\mathbf{x}) = p(\mathbf{x}, \mathbf{y})$
Then

$$g_{X_4}^{\text{sum}}(x_4) = p(x_4, \mathbf{y})$$

and

$$g_{X_k}^{\text{max}}(x_k) = \max_{\sim\{x_k\}} p(\mathbf{x}, \mathbf{y})$$

The sum-product algorithm (=belief propagation)



1. Message from variable to factor vertex

$$\mu_{X_1 \rightarrow f_A}(x_1) = \mu_{f_B \rightarrow X_1}(x_1) \times \mu_{f_C \rightarrow X_1}(x_1)$$

2. Message from factor to variable vertex

$$\mu_{f_C \rightarrow X_4}(x_4) = \sum_{x_1, x_3} f_C(x_1, x_3, x_4) \mu_{X_1 \rightarrow f_C}(x_1) \mu_{X_3 \rightarrow f_C}(x_3)$$

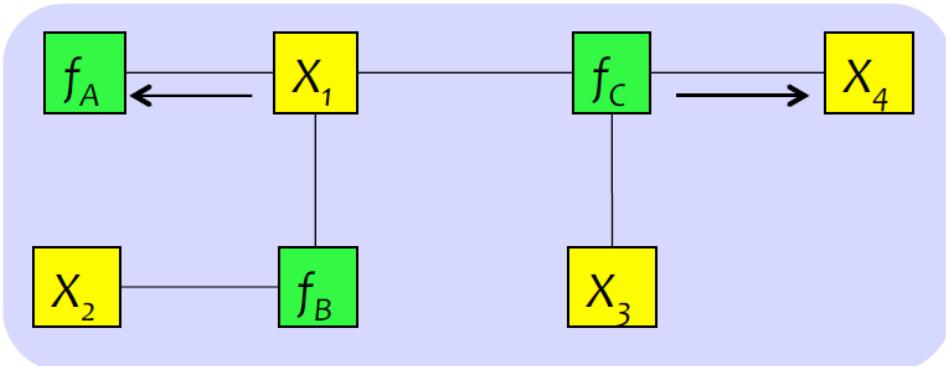
3. Sum-marginal

$$g_{X_1}^{\text{sum}}(x_1) = \mu_{f_A \rightarrow X_1}(x_1) \times \mu_{X_1 \rightarrow f_A}(x_1)$$

$$g_{X_4}^{\text{sum}}(x_4) = \mu_{f_C \rightarrow X_4}(x_4) \times \mu_{X_4 \rightarrow f_C}(x_4)$$

Initialization: Messages are initialized from leaves. Outgoing message only computed when incoming messages are available. Alternative: always transmit messages.

The max-product algorithm



1. Message from variable to factor vertex

$$\mu_{X_1 \rightarrow f_A}(x_1) = \mu_{f_B \rightarrow X_1}(x_1) \times \mu_{f_C \rightarrow X_1}(x_1)$$

2. Message from factor to variable vertex

$$\mu_{f_C \rightarrow X_4}(x_4) = \max_{x_1, x_3} f_C(x_1, x_3, x_4) \mu_{X_1 \rightarrow f_C}(x_1) \mu_{X_3 \rightarrow f_C}(x_3)$$

3. Max-marginal

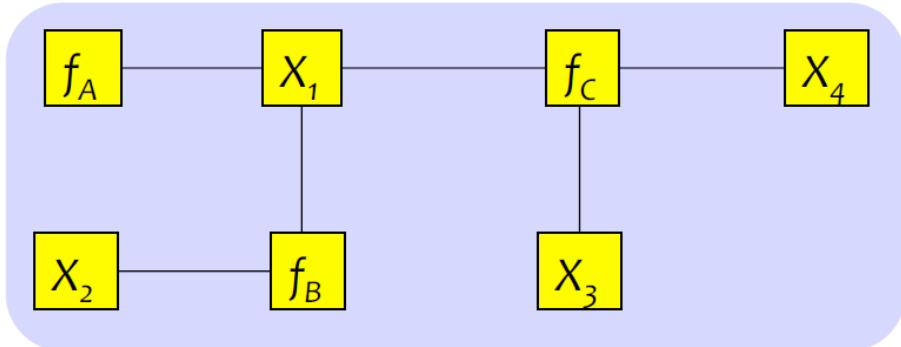
$$g_{X_1}^{\max}(x_1) = \mu_{f_A \rightarrow X_1}(x_1) \times \mu_{X_1 \rightarrow f_A}(x_1)$$

$$g_{X_4}^{\max}(x_4) = \mu_{f_C \rightarrow X_4}(x_4) \times \mu_{X_4 \rightarrow f_C}(x_4)$$

Initialization: Messages are initialized from leaves. Outgoing message only computed when incoming messages are available. Alternative: always transmit messages.

The sum-product algorithm

- $f(x_1, x_2, x_3, x_4) = f_A(x_1)f_B(x_1, x_2, x_3)f_C(x_3, x_4)$
 - Assume all variables are defined over the set {0,1}
 - Functions can be represented as a vector of size 2: $[\mu(0), \mu(1)]$



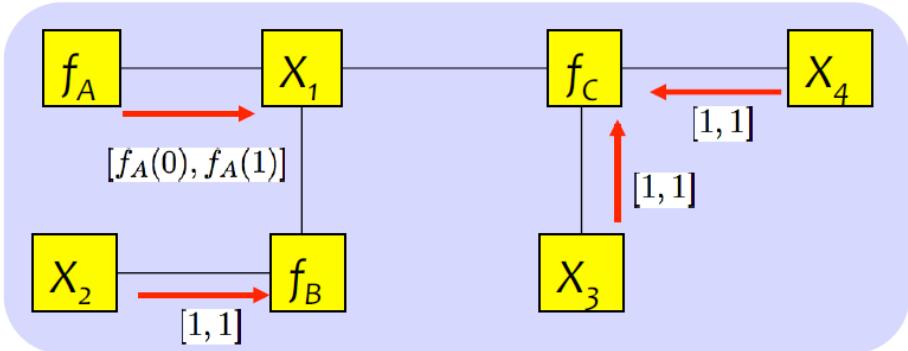
3 phases

1. Initialization
2. Message computation
3. Termination

$$\mu_{f_k \rightarrow X_i}(x_i) = \sum_{\sim\{x_i\}} \left(f_k(x_1, \dots, x_D) \prod_{j \neq i} \mu_{X_j \rightarrow f_k}(x_j) \right)$$

Phase 1: initialization

- Messages start from the edge of the graph
 - Variable vertices of degree 1 send the message “1” over the domain: $\mathbf{m}=[1,1]$
 - Factor vertices of degree 1 (say $f_k(x_i)$) send message: $\mathbf{m}=[f_k(0), f_k(1)]$



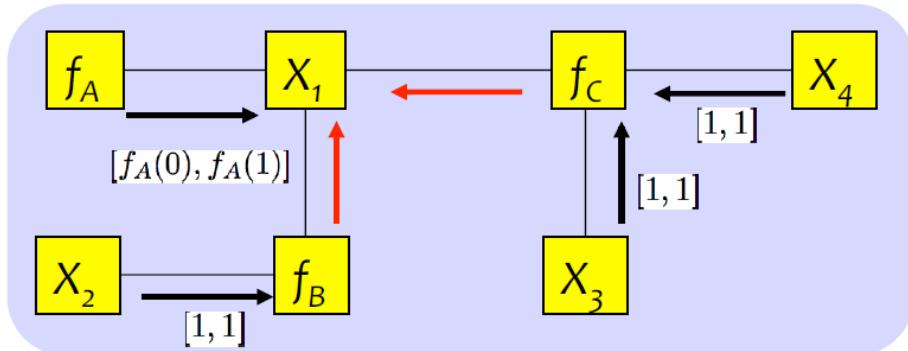
3 phases

1. Initialization
2. Message computation
3. Termination

$$\mu_{f_k \rightarrow X_i}(x_i) = \sum_{\sim\{x_i\}} \left(f_k(x_1, \dots, x_D) \prod_{j \neq i} \mu_{X_j \rightarrow f_k}(x_j) \right)$$

Phase 2: message computation

- Compute outgoing message when all other incoming messages are available



$$\mu_{f_B \rightarrow X_1}(x_1) = \sum_{x_2 \in \{0,1\}} f_B(x_1, x_2) \mu_{X_2 \rightarrow f_B}(x_2)$$

$$\begin{aligned} \mu_{f_C \rightarrow X_1}(x_1) &= \sum_{x_3, x_4 \in \{0,1\}} f_C(x_1, x_3, x_4) \mu_{X_3 \rightarrow f_C}(x_3) \mu_{X_4 \rightarrow f_C}(x_4) \\ &= \sum_{x_3, x_4 \in \{0,1\}} f_C(x_1, x_3, x_4) \end{aligned}$$

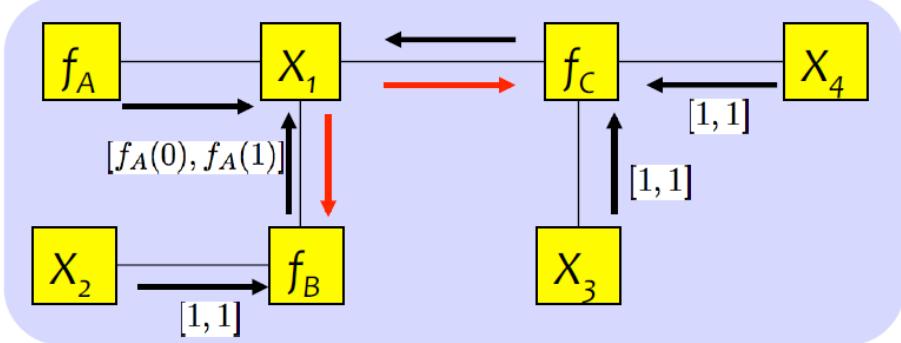
3 phases

1. Initialization
2. Message computation
3. Termination

$$\mu_{f_k \rightarrow X_i}(x_i) = \sum_{\sim\{x_i\}} \left(f_k(x_1, \dots, x_D) \prod_{j \neq i} \mu_{X_j \rightarrow f_k}(x_j) \right)$$

Phase 2: message computation

- Compute outgoing message when all other incoming messages are available



$$\mu_{X_1 \rightarrow f_C}(x_1) = \mu_{f_A \rightarrow X_1}(x_1) \mu_{f_B \rightarrow X_1}(x_1)$$

$$\mu_{X_1 \rightarrow f_B}(x_1) = \mu_{f_A \rightarrow X_1}(x_1) \mu_{f_C \rightarrow X_1}(x_1)$$

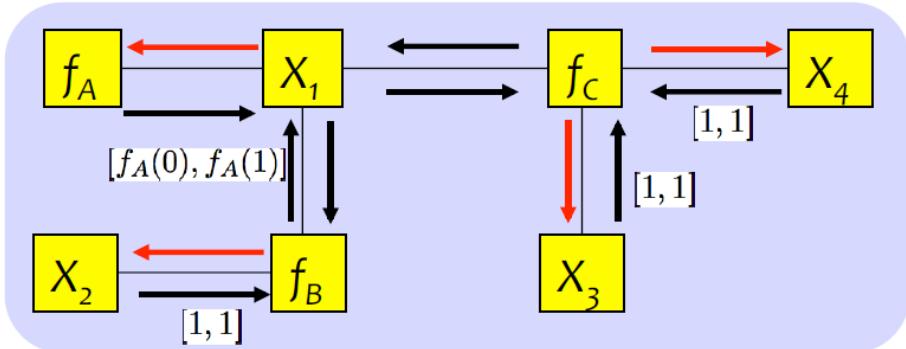
3 phases

1. Initialization
2. Message computation
3. Termination

$$\mu_{f_k \rightarrow X_i}(x_i) = \sum_{\sim\{x_i\}} \left(f_k(x_1, \dots, x_D) \prod_{j \neq i} \mu_{X_j \rightarrow f_k}(x_j) \right)$$

Phase 2: message computation

- Compute outgoing message when all other incoming messages are available



$$\mu_{f_C \rightarrow X_3}(x_3) = \sum_{x_1, x_4 \in \{0,1\}} f_C(x_1, x_3, x_4) \mu_{X_1 \rightarrow f_C}(x_1) \mu_{X_4 \rightarrow f_C}(x_4)$$

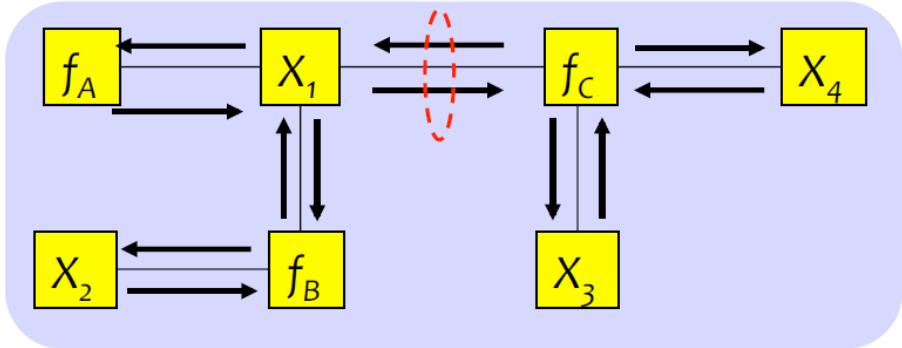
3 phases

1. Initialization
2. Message computation
3. Termination

$$\mu_{f_k \rightarrow X_i}(x_i) = \sum_{\sim\{x_i\}} \left(f_k(x_1, \dots, x_D) \prod_{j \neq i} \mu_{X_j \rightarrow f_k}(x_j) \right)$$

Phase 3: termination

- Compute outgoing message when all other incoming messages are available



$$\begin{aligned}\mu_{X_1 \rightarrow f_C}(x_1) \mu_{f_C \rightarrow X_1}(x_1) &= \sum_{x_2, x_3, x_4} f(x_1, x_2, x_3, x_4) \\ &= \mu_{X_1 \rightarrow f_A}(x_1) \mu_{f_A \rightarrow X_1}(x_1)\end{aligned}$$

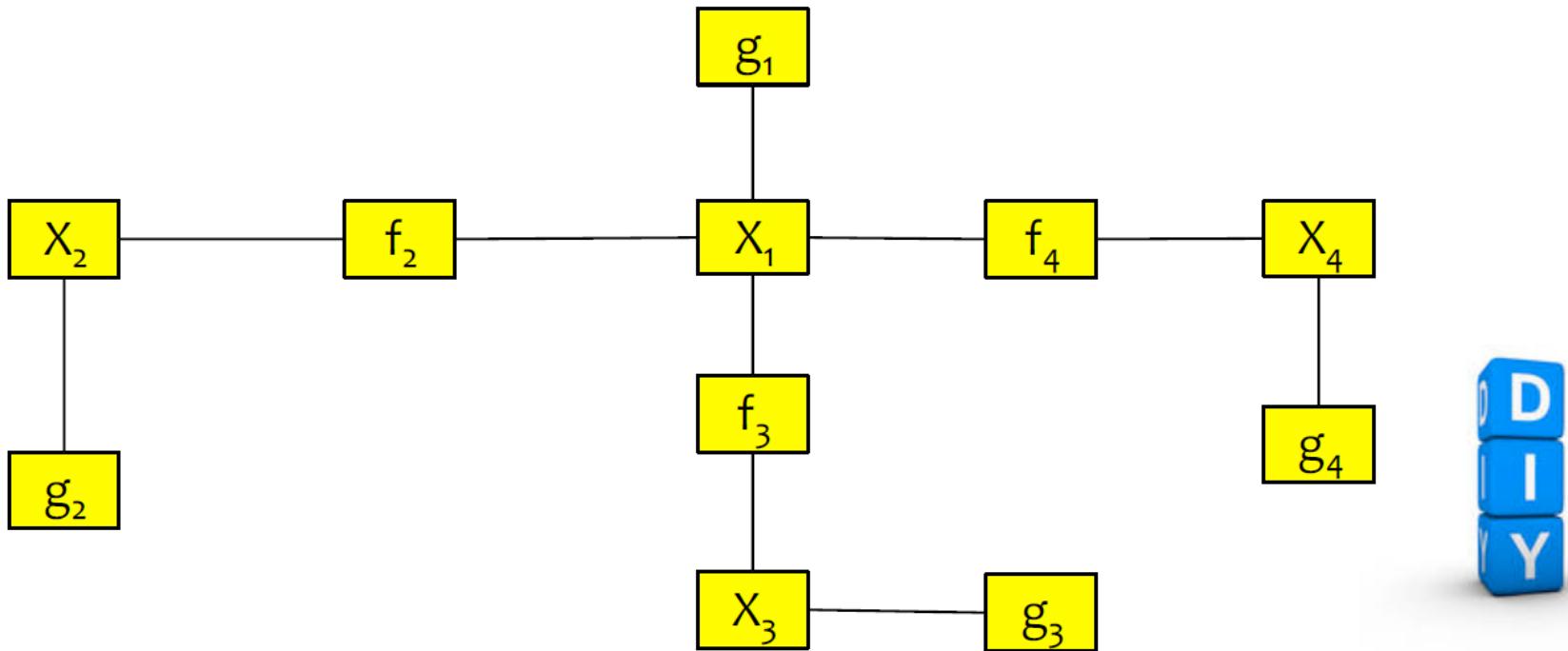
3 phases

1. Initialization
2. Message computation
3. Termination

$$g_{X_i}(x_i) = \mu_{f_k \rightarrow X_i}(x_i) \times \mu_{X_i \rightarrow f_k}(x_i)$$

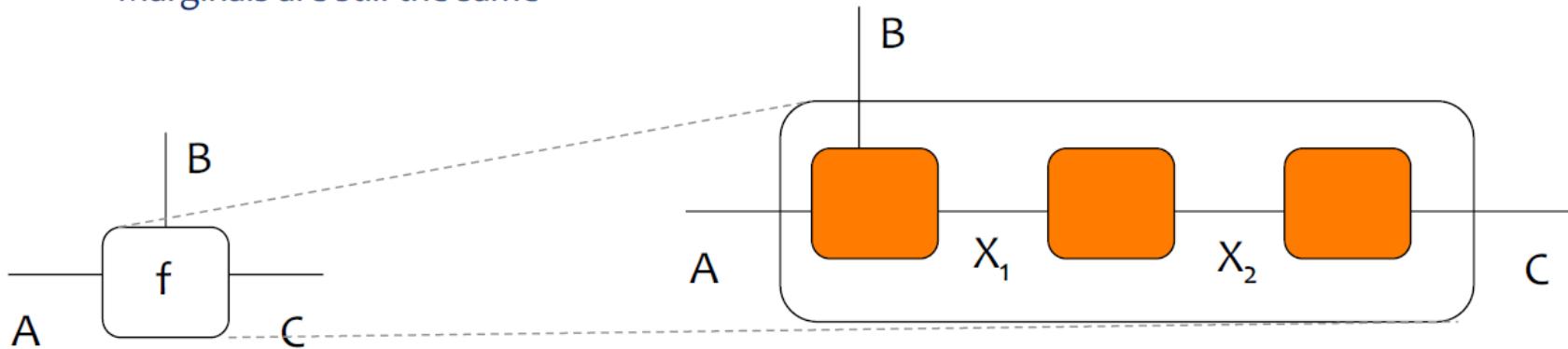
Example

- Consider $p(x_1, x_2, x_3, x_4, \mathbf{y}) = \prod_{i=1}^4 p(x_i) \times \prod_{i=2}^4 p(\mathbf{y}_i|x_1, x_i)$
- Write down the sum-product rules $g_i(x_i)$
- Verify the solution



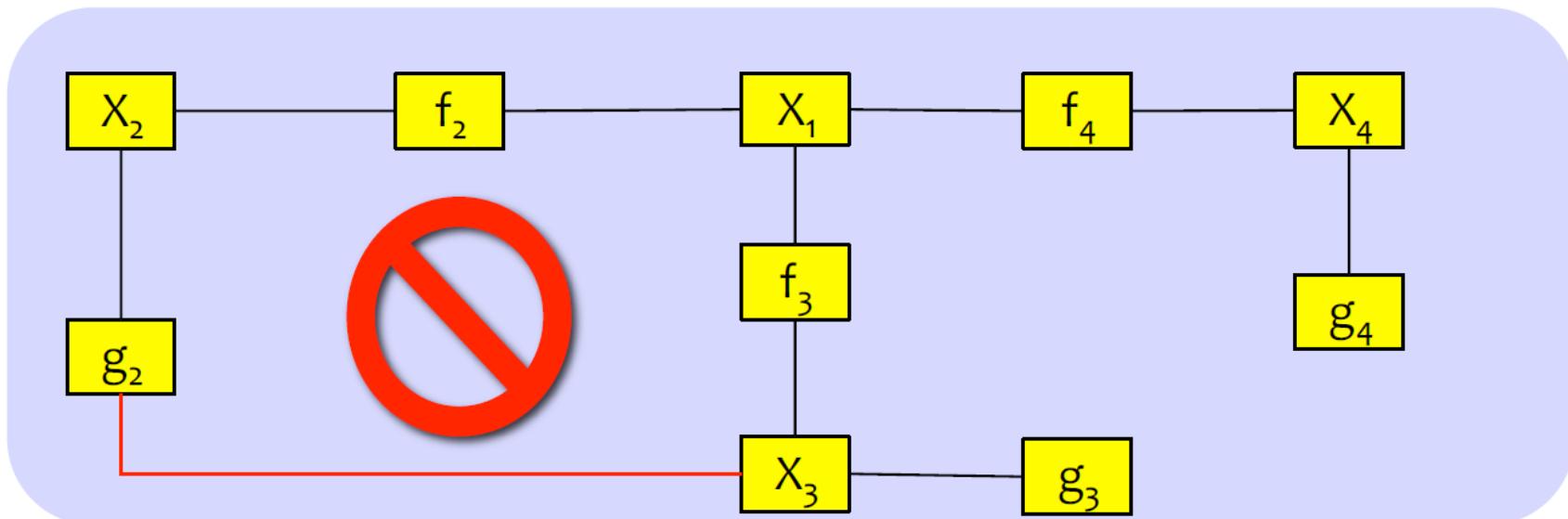
Aside: opening of vertices

- Given a vertex $f(A, B, C)$ in a larger factor graph of a joint distribution $p(A, B, C, \dots | Y=y)$
- We can replace $f(A, B, C)$ with a factorization $g(A, B, C, X_1, X_2)$ where
 - X_1, X_2 do not appear elsewhere
 - Summing out X_1 and X_2 yields again $f(A, B, C)$
 - Internal factor graph is a tree
- Outcome of [S/M]PA is not affected:
 - Messages over A, B and C are still the same
 - Marginals are still the same



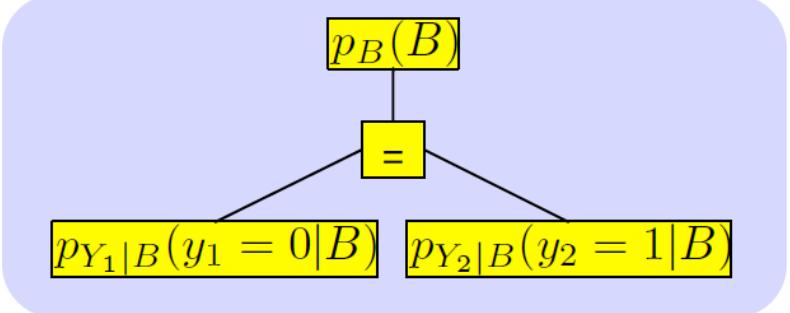
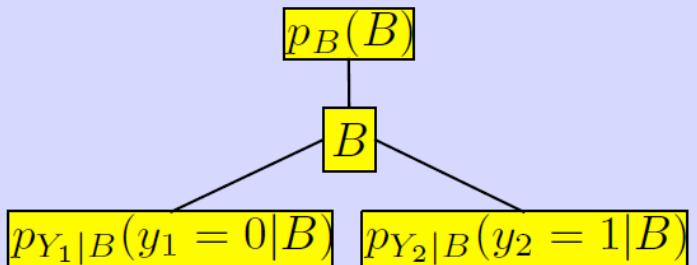
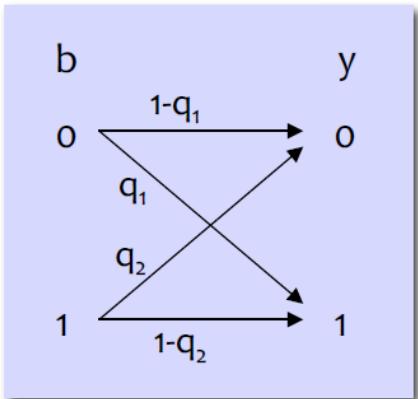
Factor graphs with cycles

- **Problem:** When the factor graph has cycles: [S/M]PA gets stuck
- **Solution:** add artificial messages (e.g., set to “1”)
- **New Problem 1:** XPA keeps running forever
- **Solution:** stop XPA after some time, and compute marginals
- **New Problem 2:** XPA becomes very unstable; messages tend to very small or very large values; marginals are generally completely incorrect
- **Conclusion:** don't use XPA for factor graphs with cycles!



Running example

- Repetition code: $\mathbf{b}=[00]$, or $\mathbf{b}=[11]$, b a priori uniform
- Binary asymmetrical channel
- $\mathbf{y}=[01]$
- Find
 - MAP estimate of b (mode of MAPD $p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y})$)
 - likelihood of receiving \mathbf{y} ($p_{\mathbf{Y}}(\mathbf{y})$, $M=[q_1, q_2]$)
- Factorization $p_{B,Y}(b,y) = p_{\mathbf{Y}|B}(\mathbf{y}|b)p_B(b)$
 $= p_{Y_1|B}(y_1|b)p_{Y_2|B}(y_2|b)p_B(b)$
- Factor graphs



Problem 1 - likelihood of the model

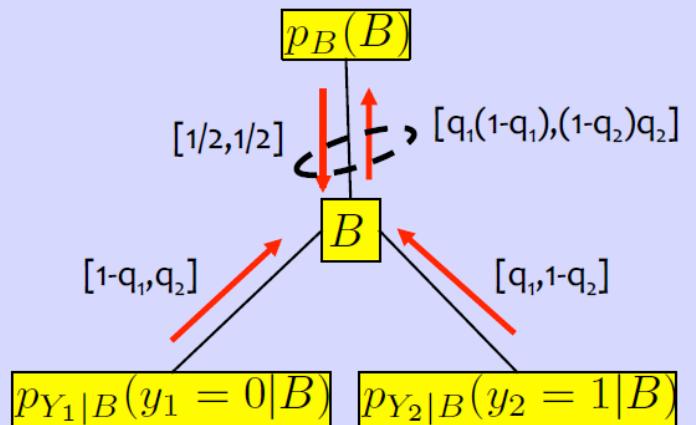
Recipe

1. Factorize the joint distribution $p_{X,Y}(x,y) = p_{Y|X}(y|x)p_X(x)$ (add variables if necessary)
2. Create a factor graph of this factorization (y is given, it is a parameter, not a variable!)
3. Perform the SPA, this yields marginals $p_{X_k,Y}(x_k,y)$
4. Take any k , sum over x_k , this gives $p_Y(y)$

$$p_{B,Y}(b=0, y=[0,1]) = 0.5 q_1(1-q_1)$$

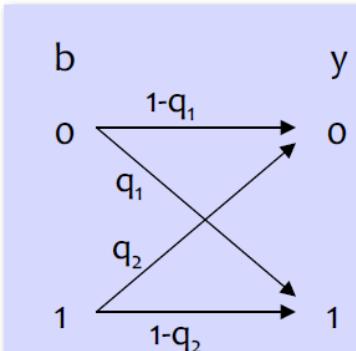
$$p_{B,Y}(b=1, y=[0,1]) = 0.5 q_2(1-q_2)$$

$$p_Y(y=[0,1]) = 0.5(q_1(1-q_1)+(1-q_2)q_2)$$



Recall:

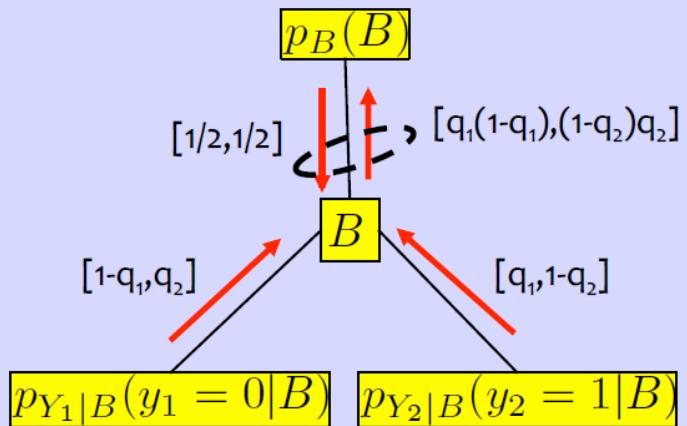
- $p(a|b) \propto p(a,b)$
- $p(a) = \sum_b p(a,b)$



Problems 2,3 - marginal a posteriori distributions

Recipe

- Factorize the joint distribution $p_{X,Y}(x,y) = p_{Y|X}(y|x)p_X(x)$ (add variables if necessary)
- Create a factor graph of this factorization (y is given, it is a parameter, not a variable!)
- Perform the SPA, this yields marginals $p_{X_k,Y}(x_k,y)$
- For the MAPD of X_k , normalizing $p_{X_k,Y}(x_k,y)$ gives $p_{X_k|Y}(x_k|y)$



$$p_{B,Y}(b=0, y=[0,1]) = 0.5 q_1(1-q_1)$$

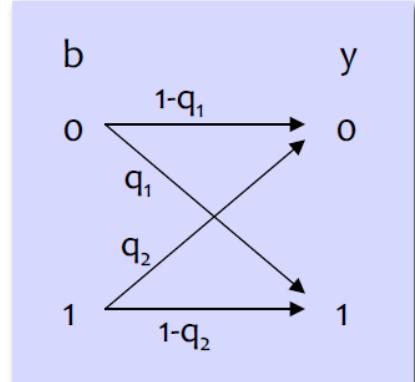
$$p_{B,Y}(b=1, y=[0,1]) = 0.5 q_2(1-q_2)$$

$$p_{B|Y}(0|[0,1]) = q_1(1-q_1) / (q_1(1-q_1) + (1-q_2)q_2)$$

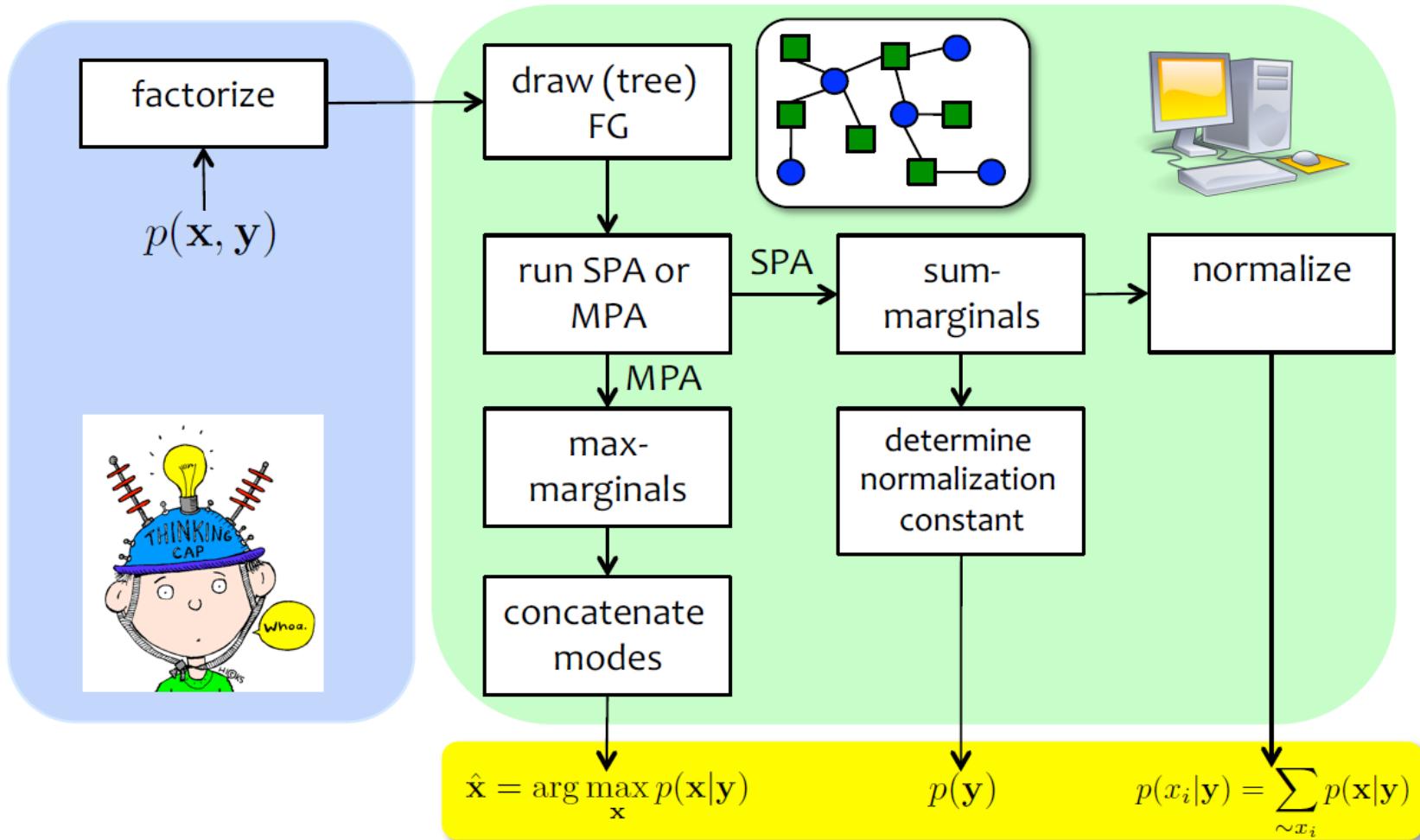
$$p_{B|Y}(1|[0,1]) = q_2(1-q_2) / (q_1(1-q_1) + (1-q_2)q_2)$$

Recall:

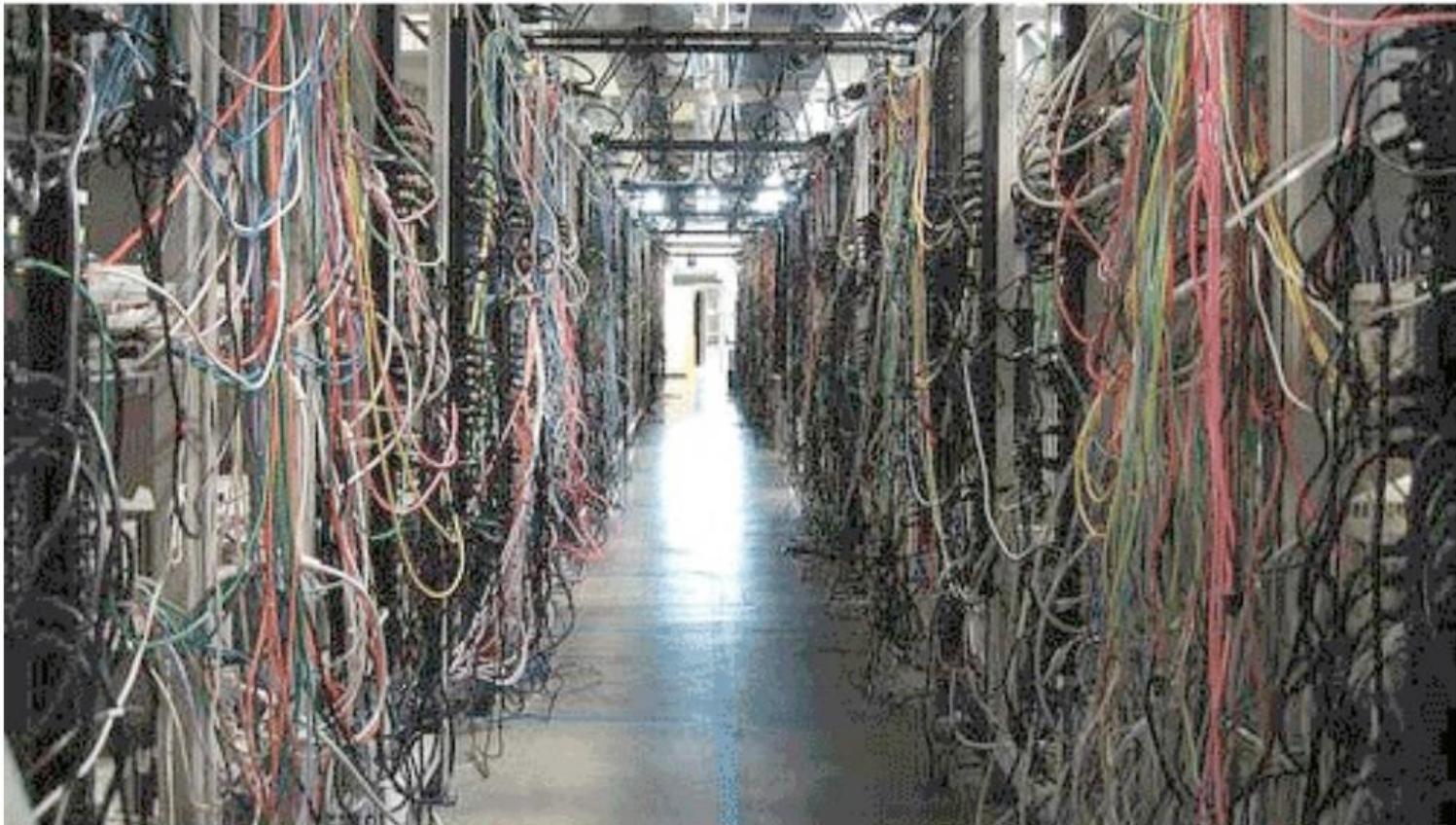
- $p(a|b) \propto p(a,b)$
- $p(a) = \sum_b p(a,b)$



Inference recipe



Practicalities



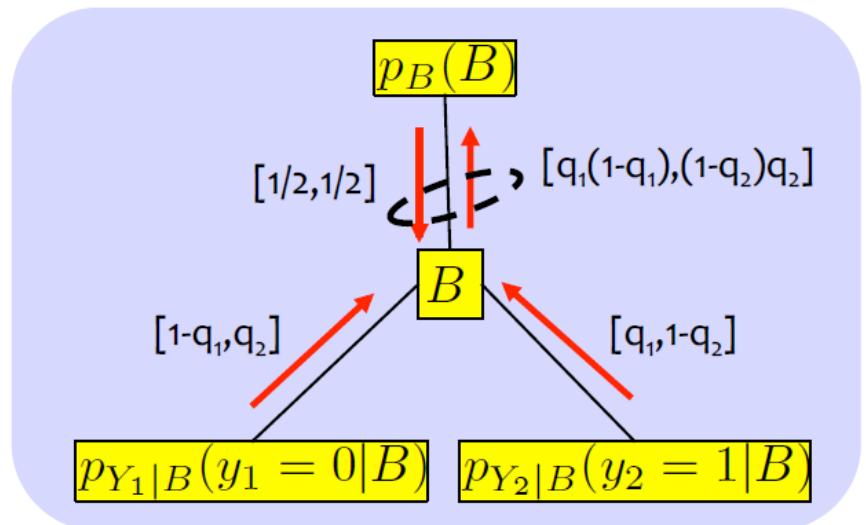


Message representation



Messages and their representations

- In inference problems
 - Initialization: messages are likelihoods or pmfs
 - SPA: multiplication and addition of messages
- Numerical stability issue
 - Messages get smaller and smaller in magnitude
- Several approaches to solve this
 - Normalization
 - Log-domain processing



Very small in magnitude

$$p_{B|Y}(0|[0,1]) = q_1(1-q_1) / (q_1(1-q_1)+(1-q_2)q_2)$$

$$p_{B|Y}(1|[0,1]) = q_2(1-q_2) / (q_1(1-q_1)+(1-q_2)q_2)$$

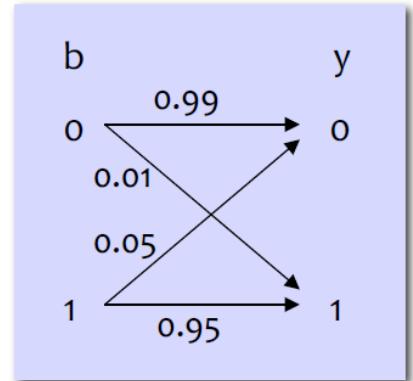


Messages representations

- Normalization
 - $M = [a; b]$ becomes $[a/(a+b); b/(a+b)]$ with normalization constant (NC) $1/(a+b)$
 - Does not affect outcome of SPA (as long as we keep track of NCs)
 - Forgetting NCs does not affect marginal posteriors
 - Forgetting NCs affects likelihood of model
- Log-domain
 - $M = [a; b]$ becomes $[\log(a) \log(b)]$
 - Efficient computation rules (“max-star”)
- Normalization + log-domain
 - $M=[a; b]$ becomes $[\log(a/b)]$
 - Log-likelihood ratio (LLR)
 - Popular for binary variables

Example: normalization

- Dividing all elements of the message with a constant, such that sum=1
 - Messages can be interpreted as distributions
 - Example: $m = [0.001, 0.02] \rightarrow \text{normalized} = [0.0476, 0.9524]$
 - Normalization constant 0.021: $m' = [0.0476, 0.9524; 0.021]$

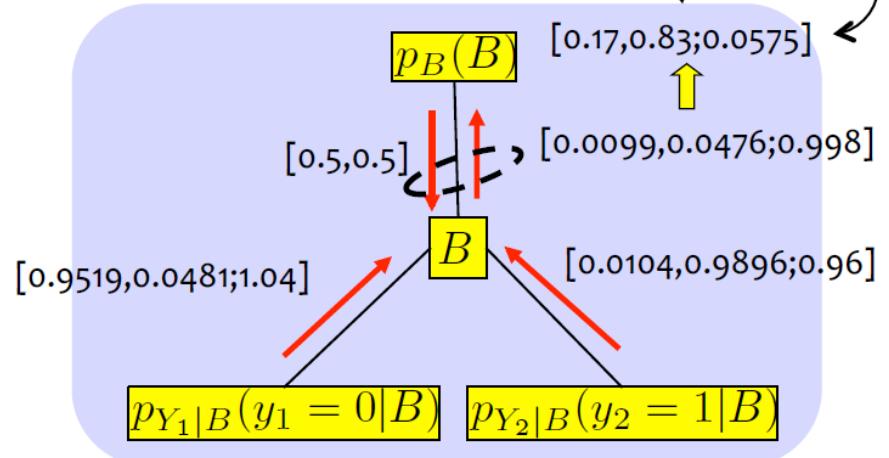
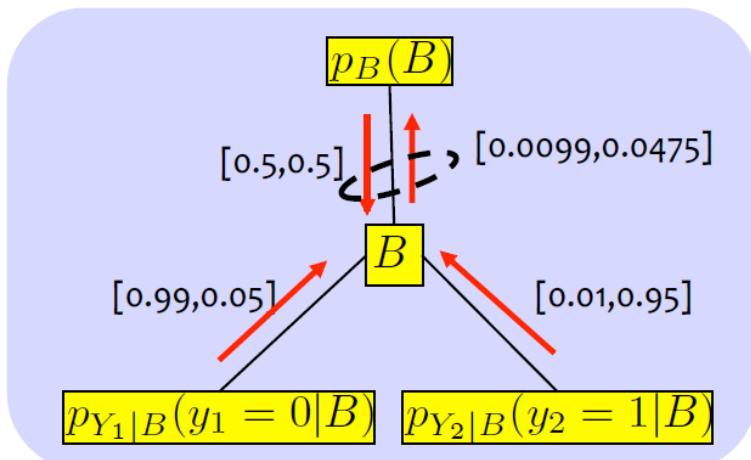


Without normalization:

$$p_{B|Y}(0|[0,1]) = 0.0099 / (0.0099 + 0.0475) = 0.17$$

$$p_{B|Y}(1|[0,1]) = 0.0475 / (0.0099 + 0.0475) = 0.83$$

$$p_Y([0,1]) = 0.0099 + 0.0475 = 0.0575$$



Example: normalization

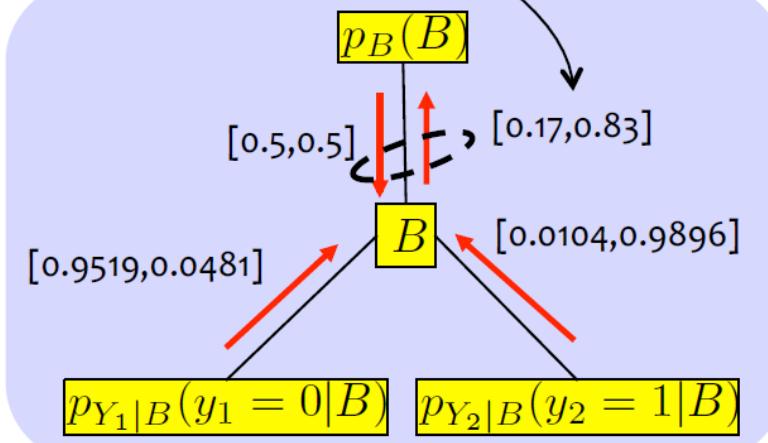
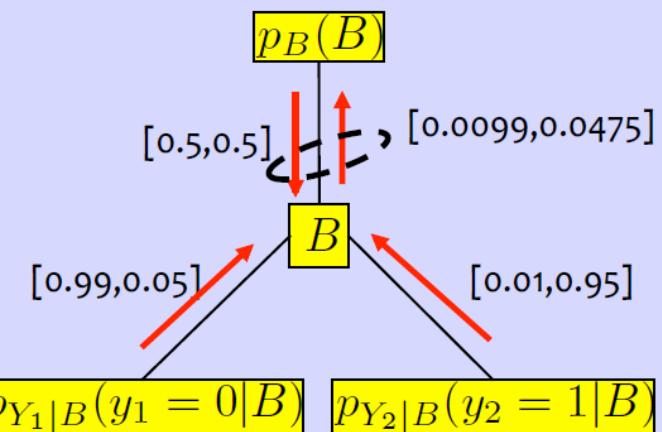
- Forgetting normalization constant is OK

Without normalization:

$$p_{B|Y}(0|[0,1]) = 0.0099 / (0.0099 + 0.0475) = 0.17$$

$$p_{B|Y}(1|[0,1]) = 0.0475 / (0.0099 + 0.0475) = 0.83$$

$$p_Y([0,1]) = 0.0099 + 0.0475 = 0.0575 \quad ???$$



Example: log-domain

- Instead of m , store $\log(m)$
- Increased dynamic range
- Product become sum

Probability domain:

$$p_{B|Y}(0|[0,1]) = 0.0099 / (0.0099 + 0.0475) = 0.17$$

$$p_{B|Y}(1|[0,1]) = 0.0475 / (0.0099 + 0.0475) = 0.83$$

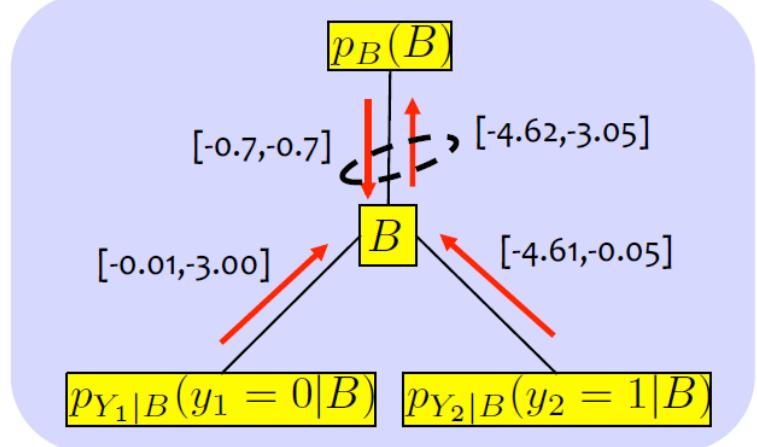
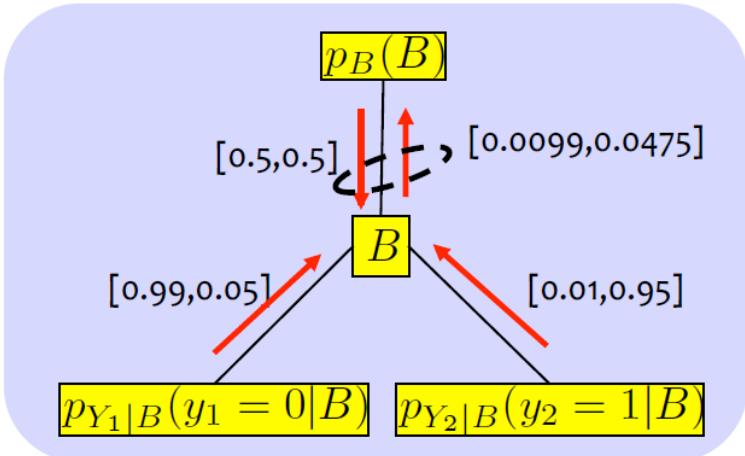
$$p_Y([0,1]) = 0.0099 + 0.0475 = 0.0575$$

Log-domain:

$$p_{B|Y}(0|[0,1]) = \exp(-4.62) / (\exp(-4.62) + \exp(-3.05)) = 0.17$$

$$p_{B|Y}(1|[0,1]) = \exp(-3.05) / (\exp(-4.62) + \exp(-3.05)) = 0.83$$

$$p_Y([0,1]) = \exp(-4.62) + \exp(-3.05) = 0.0575$$





Log-domain messages

- Go back and forth between messages and log-domain

$$\mu_{f_k \rightarrow X_i}(x_i) = \sum_{\sim\{x_i\}} \left(f(x_1, \dots, x_D) \prod_{j \neq i} \mu_{X_j \rightarrow f_k}(x_j) \right)$$

$$L_{f_k \rightarrow X_i}(x_i) = \log \left\{ \sum_{\sim\{x_i\}} \left(f(x_1, \dots, x_D) \prod_{j \neq i} e^{L_{X_j \rightarrow f_k}(x_j)} \right) \right\}$$

- Can be avoided by using the **Jacobian logarithm (aka max-star)**

$$L_{f_k \rightarrow X_i}(x_i) = \mathbb{M}_{\sim\{x_i\}} \left(\log f(x_1, \dots, x_D) + \sum_{j \neq i} L_{X_j \rightarrow f_k}(x_j) \right)$$

- Defined recursively

$$\mathbb{M}(L_1, \dots, L_L) = \mathbb{M}(L_1, \mathbb{M}(L_2, \dots, L_L))$$

$$\mathbb{M}(L_1, L_2) = \max(L_1, L_2) + \log(1 + e^{-|L_1 - L_2|})$$

- With core operation: 1 max + 1 table look-up
- Very efficient

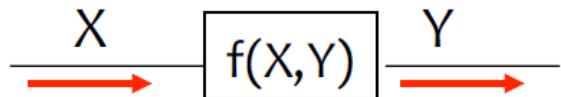
- Example for variables x_1, x_2 defined over $\{-1, 0, 1\}$

$$\mathbb{M}_{\sim\{x_1\}}(f(x_1, x_2)) = \mathbb{M}(f(x_1, -1), f(x_1, 0), f(x_1, +1))$$



Continuous variables

- Interpret messages as probability mass functions
- 3 approaches
 - Make them **discrete**, and use known techniques
 - **Parametric** representation (e.g., Gaussian mixture); message = vector of parameters
 - **Non-parametric** representation; message = vector of (weighted) samples
- We focus on non-parametric representation for simple factor:



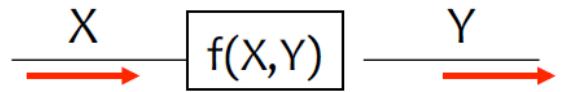
We assume we have a representation of the incoming message on X:
 $\{w_k, x_k\}, k=1..N$



Non-parametric representation: method 1

- Given N samples $\{w_k, x_k\}$ of $p_X(x)$

$$\begin{aligned}
 p_Y(y) &= \gamma \int f(x, y)p_X(x)dx \\
 &= \int p_{Y|X}(y|x)p_X(x)dx \\
 &\approx \int p_{Y|X}(y|x) \sum_{k=1}^N w_k \delta(x - x_k) dx \\
 &= \sum_{k=1}^N w_k p_{Y|X}(y|x_k)
 \end{aligned}$$



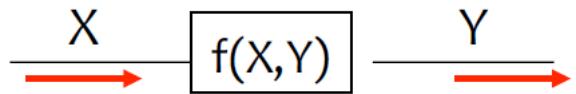
- For every x_k , draw a sample y_k from some nice pdf, weight v_k

$$\begin{aligned}
 y_k &\sim q_{Y|X}(y|x_k) \\
 v_k &\propto w_k \frac{p_{Y|X}(y|x_k)}{q_{Y|X}(y|x_k)} \quad \left. \right\} \text{unknown} \\
 &\propto w_k \frac{f(x_k, y)}{q_{Y|X}(y|x_k)} \quad \left. \right\} \text{known}
 \end{aligned}$$

Non-parametric representation: method 2

- Given N samples $\{w_k, x_k\}$ of $p_X(x)$

$$\begin{aligned}
 p_Y(y) &= \gamma \int f(x, y)p_X(x)dx \\
 &= \int p_{Y|X}(y|x)p_X(x)dx \\
 &\approx \int p_{Y|X}(y|x) \sum_{k=1}^N w_k \delta(x - x_k) dx \\
 &= \sum_{k=1}^N w_k p_{Y|X}(y|x_k)
 \end{aligned}$$



- Draw x_k from $p_X(x)$, draw a sample y_k from some nice pdf, weight v_k

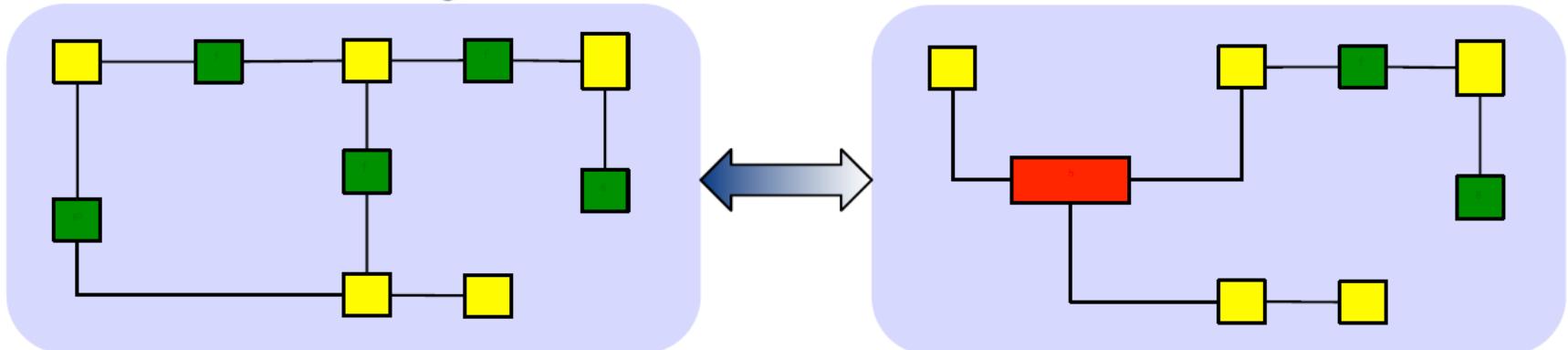
$$x_k \sim p_X(x)$$

$$y_k \sim q_{Y|X}(y|x_k)$$

$$\begin{aligned}
 v_k &\propto \frac{p_{Y|X}(y|x_k)}{q_{Y|X}(y|x_k)} \\
 &\propto \frac{f(x_k, y)}{q_{Y|X}(y|x_k)}
 \end{aligned}
 \left. \begin{array}{l} \text{unknown} \\ \text{known} \end{array} \right\}$$

Graphs with cycles

- Traditional approach
 - Convert to tree using junction tree methods



- Modern approach
 - when we normalize messages, SPA can still give good results
 - MAPD are not exact, but approximations (except means for Gaussian models)
 - Approximate MAPD are called “beliefs”
 - Possible to compute an approximation of the likelihood (using a Bethe free energy approximation)
- Many of the practical applications of SPA involve factor graphs with cycles
 - Turbo codes, LDPC codes, BICM-ID, MIMO detection, Multi-user detection

Message scheduling in Turbo decoder

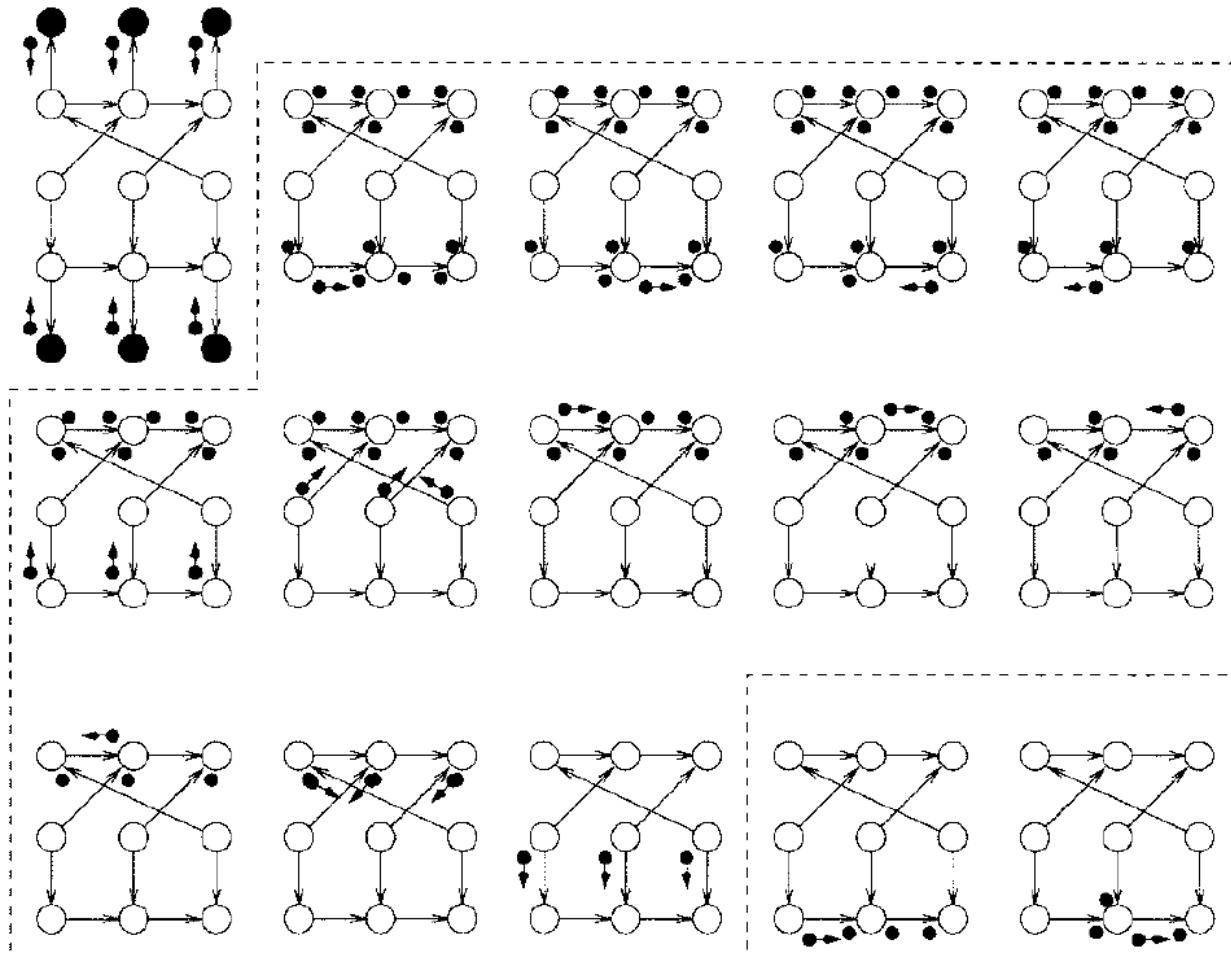
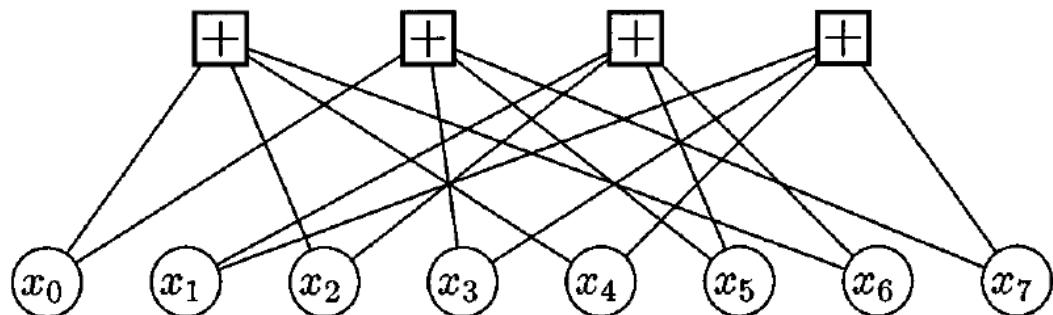
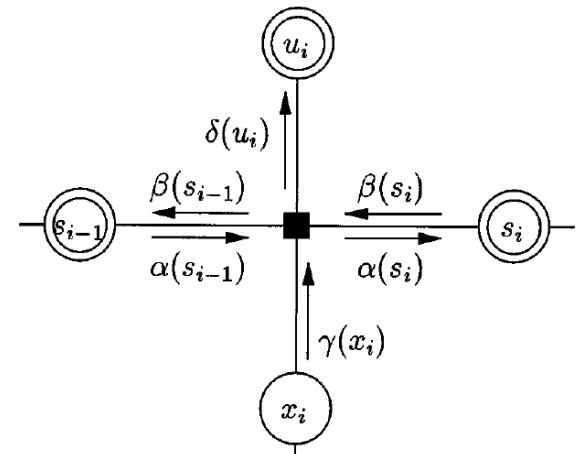
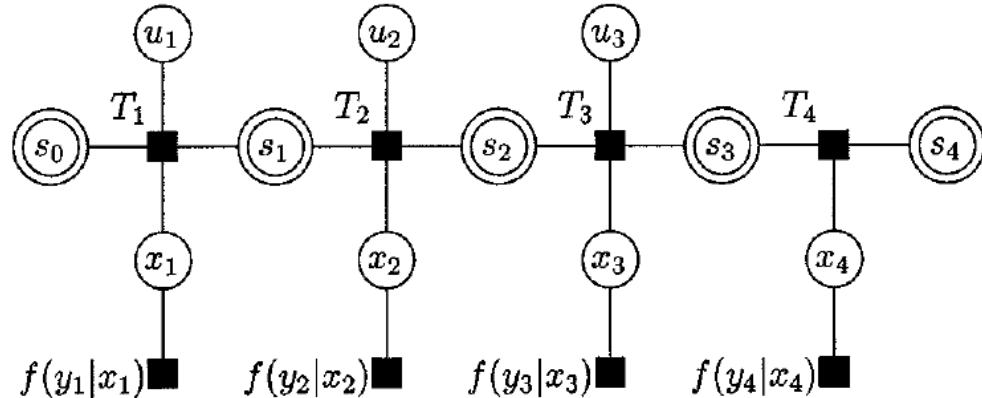


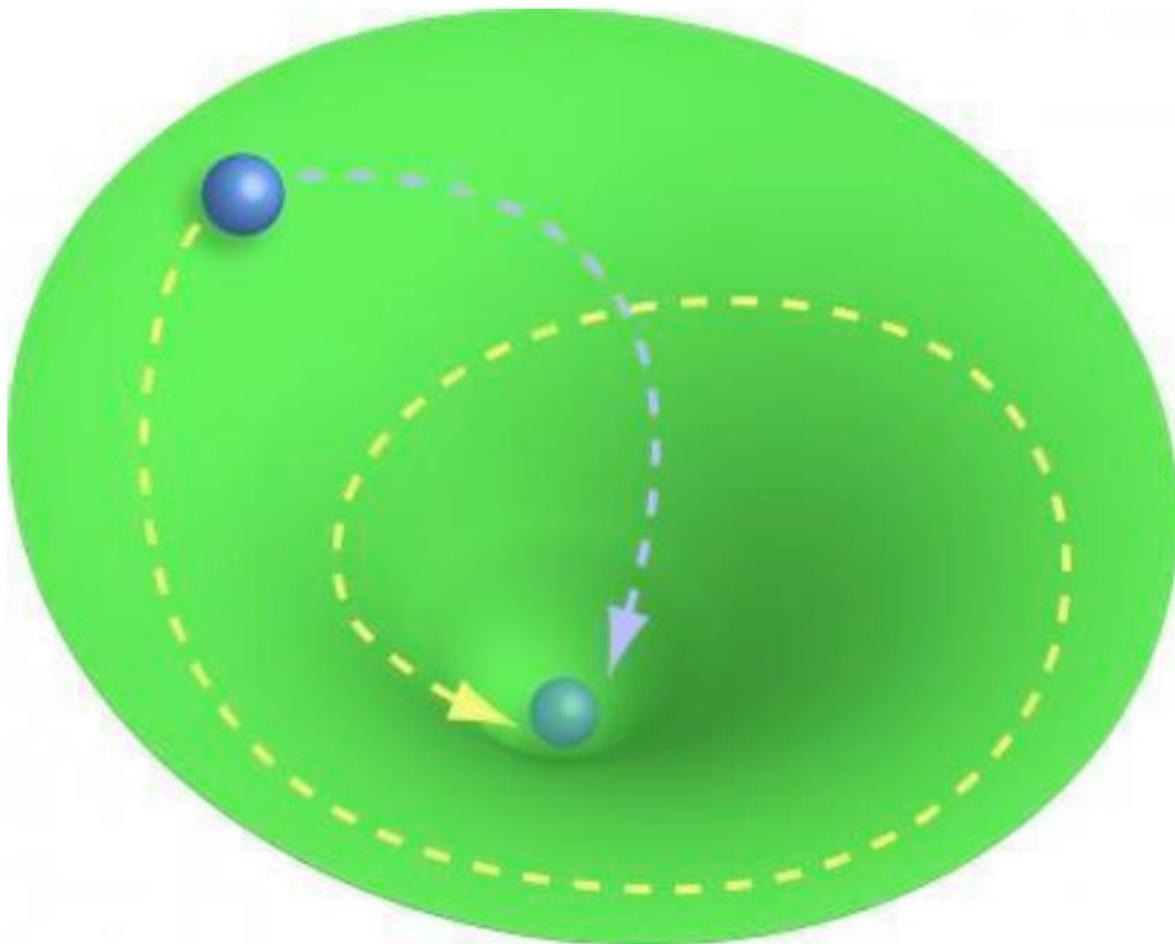
Fig. 10. Probability propagation in the Bayesian network for a turbo code. The dashed polygon encloses the steps in a single iteration. Black dots without arrows represent pending messages.

Message Scheduling in HMM And LDPC Decoder



Flooding
scheduling

Convergence behavior



Turbo decoding example

- Typical behavior

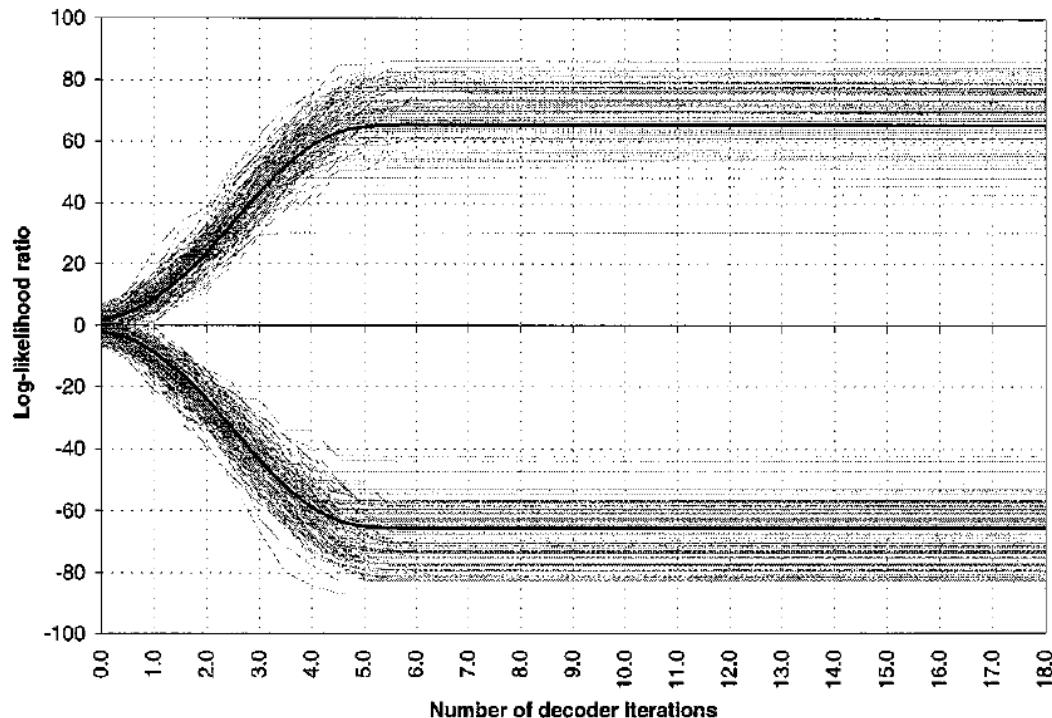


Fig. 1. Bit convergence of a typical frame in which all errors are corrected.
(SNR = 1.77 dB, random interleaver).

Source: Reid, A.C.; Gulliver, T.A.; Taylor, D.P.; , "Convergence and errors in turbo-decoding," *Communications, IEEE Transactions on* , vol.49, no.12, pp.2045-2051, Dec 2001

Turbo decoding example

- Sometimes

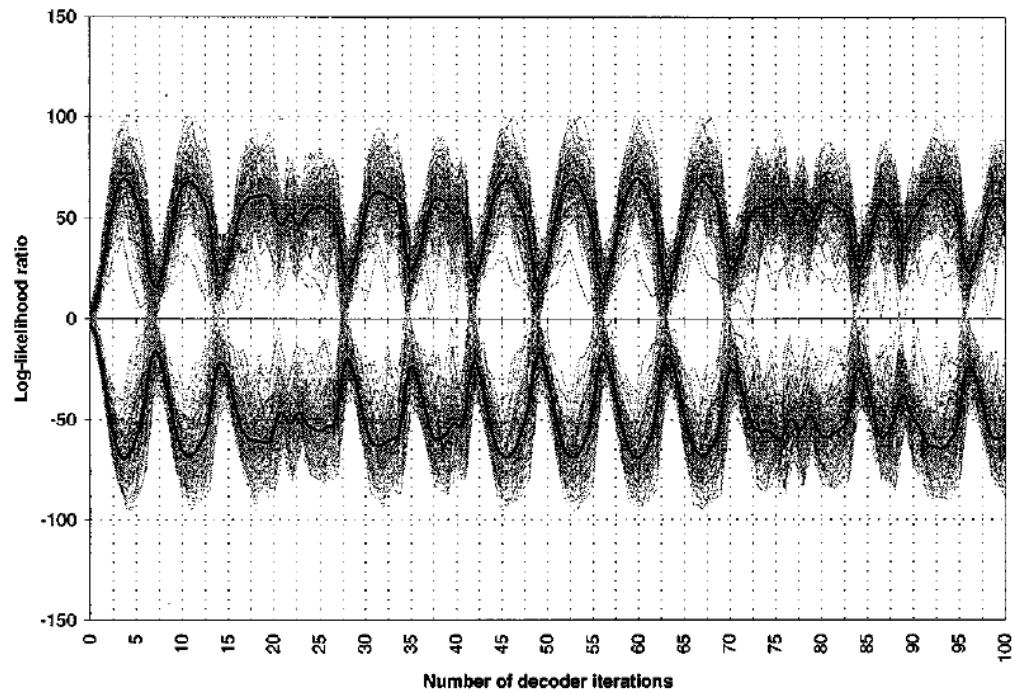


Fig. 11. Bit convergence of a frame which shows large oscillations to 100 iterations. (SNR = 3.27 dB, constrained interleaver).

Source: Reid, A.C.; Gulliver, T.A.; Taylor, D.P.; , "Convergence and errors in turbo-decoding," *Communications, IEEE Transactions on* , vol.49, no.12, pp.2045-2051, Dec 2001



Convergence results for SPA

- General models
 - Convergence is not guaranteed
 - Even when converged, no guarantee on quality of the marginals
- Gaussian models $p(\mathbf{x}) \propto \exp(-\mathbf{x}^T \mathbf{W} \mathbf{x} + \mathbf{b}^T \mathbf{x})$
$$= \prod_{i,j} \exp(-w_{ij} x_i x_j) \times \prod_i \exp(b_i x_i)$$
 1. Convergence is guaranteed when \mathbf{W} is diagonal dominant (Weiss, Freeman, 2001)
 $|w_{ii}| > \sum_{j \neq i} |w_{ij}|, \forall i$
 2. Convergence is guaranteed when $\mathbf{W} = \mathbf{I} - \mathbf{R}$, for which $\rho(\mathbf{R}) < 1$ (Malioutov, Johnsson, Willsky, 2006)
 3. When convergence is achieved, the means of the marginals are correct
 4. When convergence is achieved, the variances of the marginals are typically too small
- Non-Gaussian models (Mooij, Kappen, 2007)
 - Sufficient, not necessary conditions for convergence
 - No guarantee on quality of the solutions

Convergence results for SPA

- Rules of thumb
 - Avoid short cycles
 - Avoid strong interactions among variables

MOOIJ AND KAPPEN: SUFFICIENT CONDITIONS FOR CONVERGENCE OF THE SUM-PRODUCT ALGORITHM

4433

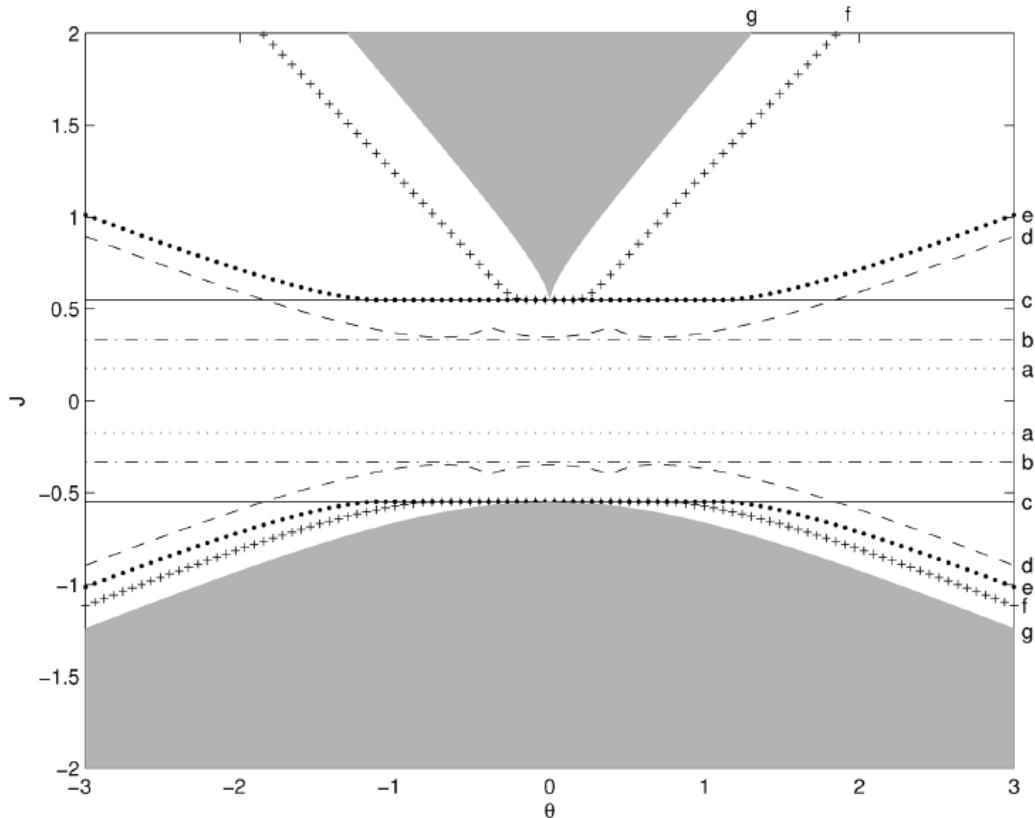


Fig. 4. Comparison of various BP convergence bounds for the fully connected $N = 4$ binary Ising model with uniform coupling J and uniform local field θ .
 a: Heskes' condition, b: Simon's condition, c: spectral radius condition, d: Dobrushin's condition, e: improved spectral radius condition for $m = 1$, f: improved spectral radius condition for $m = 5$, g: uniqueness of Gibbs' measure condition. See the main text (Section VI-A) for more explanation.



MUSIC

- [1] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 33, no. 2, pp. 387–392, 1985.
- [2] M. Wax and I. Ziskind, "Detection of the number of coherent signals by the mdl principle," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 37, no. 8, pp. 1190–1196, 1989.
- [3] B. D. Rao and K. S. Hari, "Performance analysis of root-music," IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 37, no. 12, pp. 1939–1949, 1989.
- [4] Yung-Yi Wang, Jiunn-Tsair Chen and Wen-Hsien Fang, "TST-MUSIC for joint DOA-delay estimation," IEEE Transactions on Signal Processing, vol. 49, no. 4, pp. 721-729, April 200.



MUSIC

System Model

Consider an array of M antennas receives K narrowband signals. The received signal $\mathbf{y}(t) \in \mathbb{C}^{M \times 1}$ is given by

$$\mathbf{y}(t) = \mathbf{A}(\boldsymbol{\theta}) \text{diag}(\mathbf{x}(t)) \mathbf{s}(t) + \mathbf{e}(t)$$

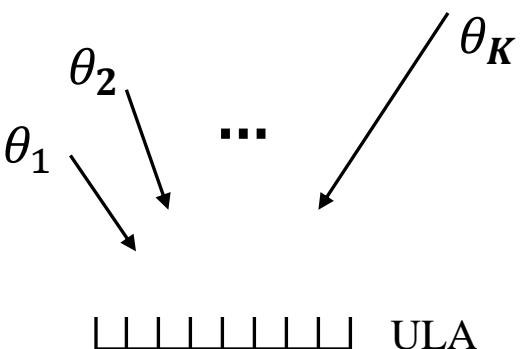
$\mathbf{s}(t) \in \mathbb{C}^{K \times 1}$ is the transmitted signal.

$\mathbf{x}(t) = [x_1(t), \dots, x_K(t)] \in \mathbb{C}^{K \times 1}$ is the complex gain.

$\mathbf{A} = [\mathbf{a}(\theta_1), \dots, \mathbf{a}(\theta_K)] \in \mathbb{C}^{M \times K}$ is the array response matrix, and

$$\mathbf{a}(\theta) = \left[1, e^{-j2\pi \frac{d}{\lambda} \sin(\theta)}, \dots, e^{-j2\pi \frac{(N-1)d}{\lambda} \sin(\theta)} \right]^T.$$

$\mathbf{e}(t) \in \mathbb{C}^{M \times 1}$ is the additive Gaussian noises.





MUSIC

MUSIC

1. Estimate the covariance matrix from T snapshots

$$\mathbf{R} = \frac{1}{T} \sum_{t=1}^T \mathbf{y}(t) \mathbf{y}^H(t)$$

2. Eigenvalue Decomposition

$$\mathbf{R} = \mathbf{U} \boldsymbol{\Sigma} \mathbf{U}^H$$

3. Construct noise subspace

$$\mathbf{R} = \mathbf{U}_S \boldsymbol{\Sigma}_S \mathbf{U}_S^H + \mathbf{U}_N \boldsymbol{\Sigma}_N \mathbf{U}_N^H$$

Number of signals K:
AIC [1] or MDL [2]

Sort by the size of the eigenvalues, and take the eigenvectors corresponding to the first K eigenvalues as the signal subspace. The remaining $M-K$ eigenvalues correspond to eigenvectors as noise subspaces.

4. Calculate spectral function

$$P_{\text{MUSIC}}(\theta) = \frac{1}{\mathbf{a}^H(\theta) \mathbf{U}_N \mathbf{U}_N^H \mathbf{a}(\theta)}$$



MUSIC

Root MUSIC

- Compared with the conventional MUSIC algorithm, the root MUSIC algorithm [3] is to apply the polynomial root-finding method to replace the spectral search of zeros.
- Using the property that the steering vector of the signal is orthogonal to the noise, we wish to find the zeros of the following polynomials:

$$f(z) = \mathbf{p}^H(z) \mathbf{U}_N \mathbf{U}_N^H \mathbf{p}(z)$$

where $\mathbf{p}(z) = [1, z, \dots, z^{M-1}]^T$ and $z = e^{j2\pi \frac{d}{\lambda} \sin(\theta)}$.

- Matlab's roots function can solve polynomial equations with non-negative exponents. For convenience, we rewrite the above equation as

$$f(z) = z^{M-1} \mathbf{p}^T(z^{-1}) \mathbf{U}_N \mathbf{U}_N^H \mathbf{p}(z)$$

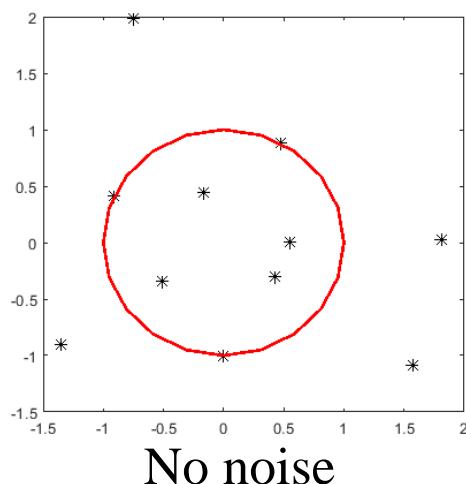
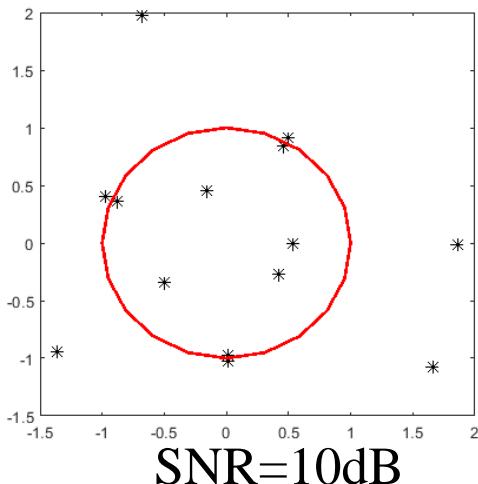
MUSIC

Root MUSIC

- Note that $f(z)$ is a polynomial of degree $2(M-1)$ whose roots are mirror images of the unit circle (z and $\frac{1}{z^*}$). Therefore, we take the phases of the K roots $\hat{z}_1, \hat{z}_2, \dots, \hat{z}_K$ whose moduli are closest to 1 within the unit circle gives the DOA estimate.

$$\hat{\theta}_m = \arcsin \left[\frac{\lambda}{2\pi d} \arg(\hat{z}_m) \right], m = 1, \dots, K$$

DOA(degree):[-30,20,60], $K=3$, $M=8$, $T=20$



For the no noise case, there are $2(M-1)$ solutions to the polynomial, of which K solutions are on the unit circle, $(M-K-1)$ solutions are within the unit circle, and $(M-K-1)$ solutions are in the unit outside the circle [3].



MUSIC

Root MUSIC

Computational complexity: The roots function treats the polynomial coefficients as a vector with $n+1$ elements representing the n -degree characteristic polynomial of the $n \times n$ matrix \mathbf{A} . The roots of the polynomial are found by computing the eigenvalues of the adjoint matrix \mathbf{A} . The computational complexity of this step is $O(2M - 1)^3$. The computational complexity of the spectral search in conventional MUSIC algorithm is $O(IM^2)$ where I is the number of searches. The root algorithm has a greater advantage in complexity when M is less than about 32.



MUSIC

2-D MUSIC

Consider a MIMO-OFDM system where P is the number of OFDM subcarriers. The received signal $\mathbf{Y} \in \mathcal{C}^{M \times P}$ is given by

$$\mathbf{Y} = \mathbf{A}(\boldsymbol{\theta}) \text{diag}(\mathbf{x}) \mathbf{B}(\boldsymbol{\tau}) + \mathbf{Z}$$

where $\mathbf{B} = [\tilde{\mathbf{b}}(\tau_1), \tilde{\mathbf{b}}(\tau_2), \dots, \tilde{\mathbf{b}}(\tau_K)]^T \in \mathcal{C}^{K \times P}$, $\tilde{\mathbf{b}}(\tau_k) = \mathbf{s}_k^T \mathbf{b}(\tau_k)$,

and

$$\mathbf{b}(\tau) = [1, e^{-j2\pi\Delta_f\tau}, \dots, e^{-(P-1)j2\pi\Delta_f\tau}]^T$$

1. Estimate the Spatial covariance matrix and temporal covariance matrix

$$\mathbf{R}^s = \frac{1}{P} \mathbf{Y} \cdot \mathbf{Y}^{II}$$

$$\mathbf{R}^t = \frac{1}{M} \mathbf{Y}^T \cdot \mathbf{Y}^*$$



MUSIC

2. Eigenvalue Decomposition and construct noise subspace

$$\mathbf{R}^s = \mathbf{V}_s^s \Lambda_s^s \mathbf{V}_s^{sII} + \mathbf{V}_n^s \Lambda_n^s \mathbf{V}_n^{sII}$$

$$\mathbf{R}^t = \mathbf{V}_s^t \Lambda_s^t \mathbf{V}_s^{tII} + \mathbf{V}_n^t \Lambda_n^t \mathbf{V}_n^{tII}$$

3. Calculate spectral function

$$\mathcal{P}_{music}^s(\theta) = \frac{1}{\mathbf{a}(\theta)^H (\mathbf{I} - \mathbf{V}_s^s \mathbf{V}_s^{sII}) \mathbf{a}(\theta)}$$

$$\mathcal{P}_{music}^t(\tau) = \frac{1}{\tilde{\mathbf{b}}(\tau)^H (\mathbf{I} - \mathbf{V}_s^t \mathbf{V}_s^{tII}) \tilde{\mathbf{b}}(\tau)}$$

4. Pair (TST-MUSIC [4])

- T-MUSIC : Obtain the group delays $\{\hat{t}_1, \dots, \hat{t}_q\}$
- The temporal filtering matrices $U_n^t, n = 1, \dots, q$ are generated by

$$\mathbf{U}_n^t = \mathbf{I} - \tilde{\mathbf{b}}^*(\hat{t}_n) \tilde{\mathbf{b}}^T(\hat{t}_n)$$

whereas the output of the k th temporal filter is given by

$$\mathbf{X}_k = \mathbf{X}_t \cdot \prod_{n=1; n \neq k}^q \mathbf{U}_n^t$$

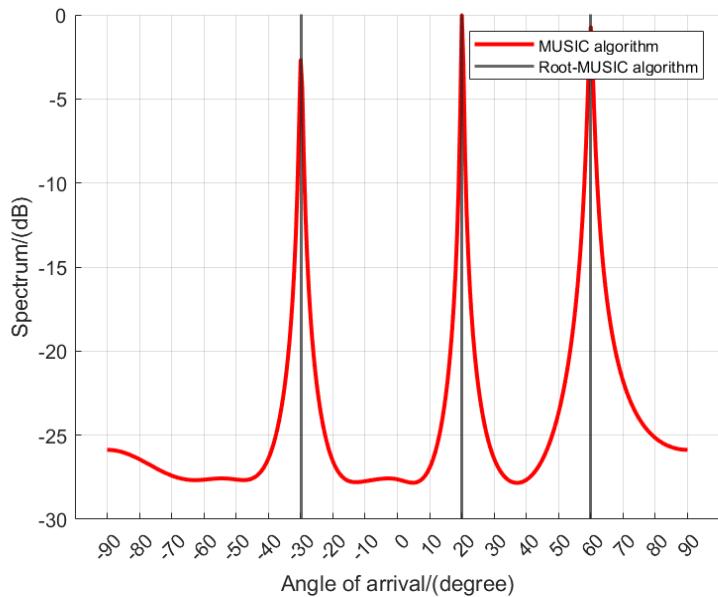
- S-MUSIC

MUSIC

Simulation Results

MUSIC and Root MUSIC

DOA(degree):[-30,20,60], and SNR = 10dB.

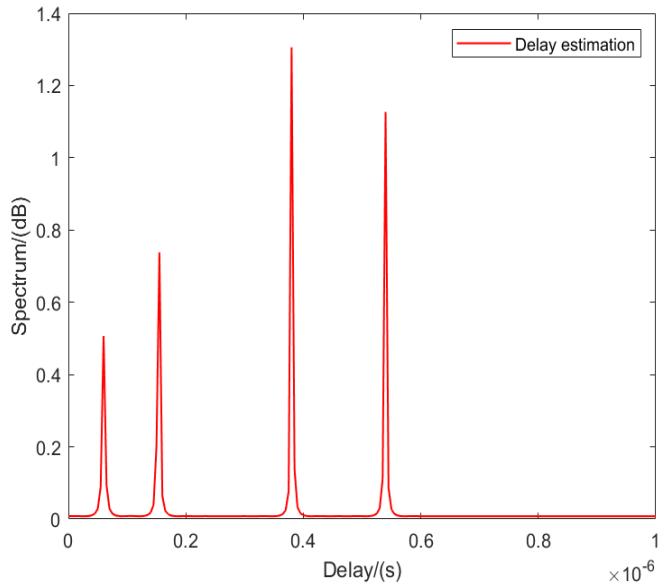


MUSIC and Root MUSIC

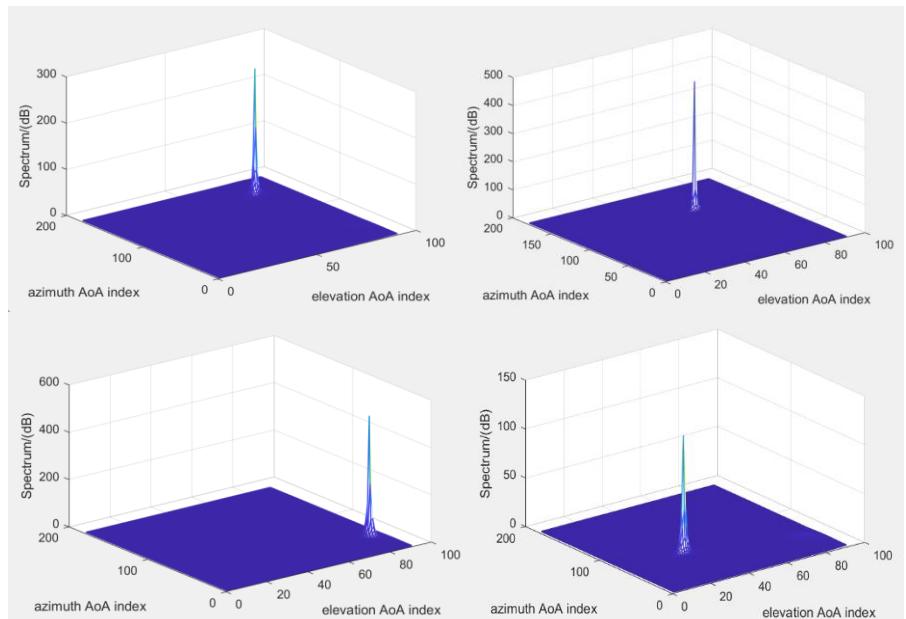
MUSIC

Simulation Results

2-D MUSIC (Estimate angle and delay)



T-MUSIC



S-MUSIC



Performance Bounds

- [1] Guo D , Shamai S , Verdu S . Mutual Information and Minimum Mean-Square Error in Gaussian Channels[J]. IEEE Transactions on Information Theory, 2005, 51(4):1261-1282.
- [2] Prasad S . Certain Relations between Mutual Information and Fidelity of Statistical Estimation[J]. Mathematics, 2010.



Basic Concepts

- Likelihood Function and Entropy

Entropy is the measure of uncertainty of a random variable X

The entropy of X is given by:

1) If X is a discrete random variable with alphabet X and probability mass function p(x)

$$H(X) = - \sum_{x \in X} p(x) \ln p(x) = -E[\ln p(x)]$$

2) If X is a continuous random variable with p.d.f. f(X)

$$h(X) = - \int f(x) \ln f(x) dx$$



Cramer-Rao Bound

- Definition

The Cramer-Rao Bound (CRB) sets a lower bound on the variance of any unbiased estimator, the definition of Cramer-Rao bound is illustrated as follows

$$MSE(\hat{\theta}) \geq I^{-1}(\theta)$$

$$[I(\theta)]_{ij} = E\left\{ \frac{\partial \ln p(x; \theta)}{\partial \theta_i} \frac{\partial \ln p(x; \theta)}{\partial \theta_j} \right\}$$

where $I(\theta)$ is fisher information matrix.

Remarks : It is well known that parameter estimation problem is equivalent to search the peak of the likelihood function of the unknown parameters. The more peaky or spiky the likelihood function, the easier it is to determine the unknown parameters. The peakiness is effectively measured by the negative of the second derivative of the log-likelihood at its peak, in others words, the Cramer-Rao Bound.

Cramer-Rao Bound

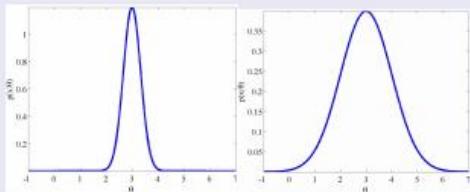
- Typical Example

Example:

Suppose we observe

$$x = A + w$$

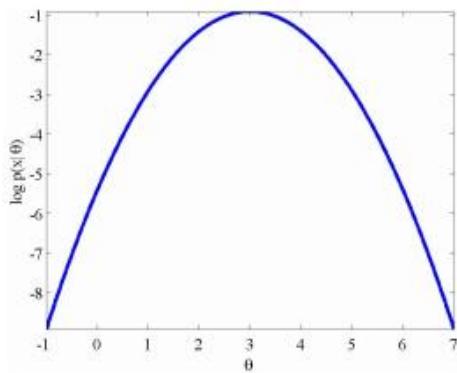
where $w \sim N(0, \sigma^2)$ and A is an unknown parameter. The "smaller" the noise w is, the easier it will be to estimate A from the observation x .



$$A = 3 \text{ and } \sigma = 1/3 \quad A = 3 \text{ and } \sigma = 1$$

Given the left density function we can easily rule out estimates of A greater than 4 or less than 2, since it is very unlikely that such A could give rise to our observation. On the other hand, when $\sigma = 1$ the noise power is larger and it is very difficult to estimate A accurately.

The key thing to notice is that the estimation accuracy of A depends on σ , which in effect determines the peakiness of the likelihood. The more peaky, the better localized the data is about the true parameter.





Mutual Information

- Definition

The definition of mutual information is illustrated as follows

$$I(X;Y) = H(X) - H(X|Y)$$

where X and Y are two random variables. Hence, mutual information is the reduction in uncertainty of X due to knowledge of Y.

- Relation between mutual information and minimum mean-square error

Under the assumption that the conditional pdf of the measurement is Gaussian, the minimum mean-square error (MMSE) is the derivative of mutual information [1].

$$\frac{d}{dSNR} I(SNR) = \frac{1}{2} MMSE(SNR)$$



Mutual Information

- Relation between mutual information and minimum mean-square error

A more important relation between information and estimation error has been obtained in the fully Bayesian context of minimum mean squared error. The conditional differential entropy, $h(X|Y)$, cannot exceed $(1/2)\ln(2\pi eMMSE)$ and hence the mutual information is bounded below by [2]

$$I(X;Y) \geq h(X) - (1/2)\ln(2\pi eMMSE)$$

Remarks: The existence of such a lower bound, while extending previous work relating the mutual information to the fisher information that is valid only in the asymptotic and high-SNR limits, elucidates further the fundamental connection between information and estimation theoretic measures of fidelity.



Part II: Compressive Sensing (CS)

- Compressive Sensing in a Nutshell
- Optimization Based CS recovery Algorithm
 - ℓ_1 -norm minimization, Reweighted ℓ_1 -norm minimization, LASSO, Generalized LASSO
 - RIP and Performance Guarantee
- Greedy CS Recovery Algorithms
 - OMP, CoSaMP, SP
- Message Passing CS Recovery Algorithms
 - AMP, Big-AMP for bilinear models
- Applications to Wireless Communications



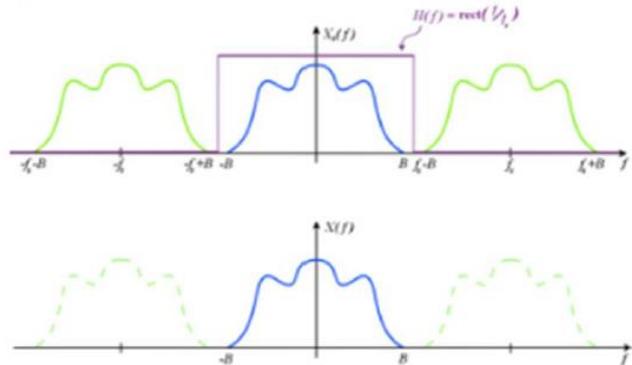
Compressive Sensing in a Nutshell

- [1] E. Candes and M. Wakin, “An introduction to compressive sampling,” IEEE Signal Process. Mag., vol. 25, no. 2, pp. 21–30, Mar. 2008.
- [2] R. Baraniuk, M. A. Davenport, M. F. Duarte, C. Hegde, J. Laska, M. Sheikh, and W. Yin, “An introduction to compressive sensing,” 2011 [Online]. Available: <http://cnx.org/content/col11133/1.5/>

Background of Compressive Sensing

● Compressive Sensing (CS)

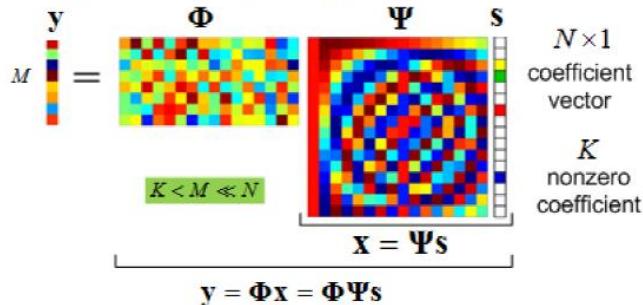
- (1949) Shannon-Nyquist sampling theory: sufficient condition for perfect reconstruction of a bandwidth limited signals



$$f_s \geq 2B$$

Shannon

- (2006) Compressive sensing: Acquire and reconstruct a sparse signal by a sampling rate much lower than the Nyquist rate



$$M \ll N$$

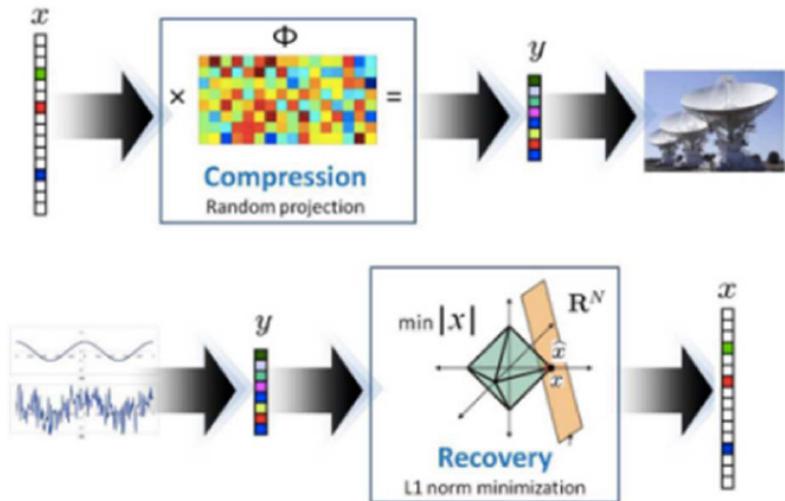
Donoho

D. L. Donoho, "Compressed sensing", *IEEE Trans. Info. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006. (cited by 14106 times)

Background of Compressive Sensing

● Key idea of CS

- Conventional way: sampling the signal at a high rate first, and then compress the signal to remove the redundancy
- CS: directly sampling the inherent information of signals, and then reconstruct the high-dimensional signal from low-dimensional measurement via optimization
- Three steps of CS: 1) Spare representation; 2) Compression; 3) Recovery
- Applications of CS: Image/Vedio processing, MRI, Radar, Wireless communications ...



Compressive sensing is a **breakthrough theory** for signal processing, which has great potential impacts in many applied fields including wireless communications



Compressive Sensing in a Nutshell

- The standard compressive sensing model

$$\mathbf{y} = \Phi \mathbf{x} + \mathbf{n}$$

M < N Compressed measurements ← N x 1 sparse signal → M x N measurement matrix → M x 1 noise vector

- Understand CS in the noiseless case $\mathbf{y} = \Phi \mathbf{x}$

- If there is no sparse constraint on \mathbf{x} , this linear equation is underdetermined has multiple solution -> impossible to recover the signal
- However, if there is a sparse constraint on \mathbf{x} , and the sparsity level K (# of non-zero elements of \mathbf{x}) is sufficiently small, then the solution of this linear equation which also satisfies the sparse constraint is unique -> The signal can be perfectly recovered

$$\mathbf{x} \in \Sigma_{(N,K)} \triangleq \{\mathbf{z} : \mathbf{z} \in \mathbb{C}^N, \|\mathbf{z}\|_0 \leq K\} \quad \text{Sparse constraint}$$

- How about the noisy case?

- If $M > N$, \mathbf{x} can be recovered using the maximum likelihood estimation (MLE):

$$\mathbf{x} = \underset{\mathbf{z}}{\operatorname{argmin}} \|\Phi \mathbf{z} - \mathbf{y}\|_2 = \Phi^\dagger \mathbf{y}.$$

- Similarly, when $M < N$, under the sparse constraint with sufficiently small K , the resulting *sparse MLE problem* has unique solution which is a *stable estimation*

$$\mathbf{x} = \underset{\mathbf{z}}{\operatorname{argmin}} \|\Phi \mathbf{z} - \mathbf{y}\|_2 \quad \text{s.t. sparse constraint}$$



CS Recovery Algorithms

- When should we use CS?
 - The signal to be estimated has large dimension and is sparse ($K \ll N$)
 - The support (locations of the non-zero elements) of the sparse signal is unknown
- Existing CS recovery algorithm can be classified into three classes
 - Optimization based CS recovery algorithm, e.g., ℓ_0 -norm/ ℓ_1 -norm minimization
$$\hat{\mathbf{x}} = \underset{\mathbf{z}}{\operatorname{argmin}} \|\mathbf{z}\|_0, \text{ s.t. } \|\Phi\mathbf{z} - \mathbf{y}\|_2 \leq \varepsilon \quad \text{Non-convex, not useful in practice}$$
$$\hat{\mathbf{x}} = \underset{\mathbf{z}}{\operatorname{argmin}} \|\mathbf{z}\|_1, \text{ s.t. } \|\Phi\mathbf{z} - \mathbf{y}\|_2 \leq \varepsilon \quad \text{Convex relaxation, widely used in practice}$$
 - Greedy CS recovery algorithms
 - OMP, CoSaMP, SP
 - Good tradeoff between performance and complexity
 - Message passing CS recovery algorithms
 - BP, AMP
 - Good tradeoff between performance and complexity
 - In large scale CS problem, accurate performance bound can be obtained using state evolution



Optimization Based CS Recovery Algorithm

The Standard Compressive Sensing Problem



- \mathbf{x} : unknown signal of interest in \mathbb{K}^N
- \mathbf{y} : measurement vector in \mathbb{K}^m with $m \ll N$,
- s : sparsity of $\mathbf{x} = \text{card}\{j \in \{1, \dots, N\} : x_j \neq 0\}$.

Find concrete sensing/recovery protocols, i.e., find

- ▶ measurement matrices $A : \mathbf{x} \in \mathbb{K}^N \mapsto \mathbf{y} \in \mathbb{K}^m$
- ▶ reconstruction maps $\Delta : \mathbf{y} \in \mathbb{K}^m \mapsto \mathbf{x} \in \mathbb{K}^N$

such that

$$\Delta(A\mathbf{x}) = \mathbf{x} \quad \text{for any } s\text{-sparse vector } \mathbf{x} \in \mathbb{K}^N.$$

In realistic situations, two issues to consider:

Stability: \mathbf{x} not sparse but compressible,

Robustness: measurement error in $\mathbf{y} = A\mathbf{x} + \mathbf{e}$.

ℓ_0 -Minimization



Since

$$\|\mathbf{x}\|_p^p := \sum_{j=1}^N |x_j|^p \xrightarrow{p \rightarrow 0} \sum_{j=1}^N \mathbf{1}_{\{x_j \neq 0\}},$$

the notation $\|\mathbf{x}\|_0$ [sic] has become usual for

$$\|\mathbf{x}\|_0 := \text{card}(\text{supp}(\mathbf{x})), \quad \text{where } \text{supp}(\mathbf{x}) := \{j \in [N] : x_j \neq 0\}.$$

For an s -sparse $\mathbf{x} \in \mathbb{K}^N$, observe the equivalence of

- ▶ \mathbf{x} is the unique s -sparse solution of $A\mathbf{z} = \mathbf{y}$ with $\mathbf{y} = A\mathbf{x}$,
- ▶ \mathbf{x} can be reconstructed as the unique solution of

$$(P_0) \quad \underset{\mathbf{z} \in \mathbb{K}^N}{\text{minimize}} \quad \|\mathbf{z}\|_0 \quad \text{subject to } A\mathbf{z} = \mathbf{y}.$$

This is a combinatorial problem, NP-hard in general.



Minimal Number of Measurements

Given $A \in \mathbb{K}^{m \times N}$, the following are equivalent:

1. Every s -sparse \mathbf{x} is the unique s -sparse solution of $A\mathbf{z} = A\mathbf{x}$,
2. $\ker A \cap \{\mathbf{z} \in \mathbb{K}^N : \|\mathbf{z}\|_0 \leq 2s\} = \{\mathbf{0}\}$,
3. For any $S \subset [N]$ with $\text{card}(S) \leq 2s$, the matrix A_S is injective,
4. Every set of $2s$ columns of A is linearly independent.

As a consequence, exact recovery of every s -sparse vector forces

$$m \geq 2s.$$

This can be achieved using partial Vandermonde matrices.



ℓ_1 -Minimization (Basis Pursuit)

Replace (P_0) by

$$(P_1) \quad \underset{\mathbf{z} \in \mathbb{K}^N}{\text{minimize}} \quad \|\mathbf{z}\|_1 \quad \text{subject to } A\mathbf{z} = \mathbf{y}.$$

- ▶ Geometric intuition
- ▶ Unique ℓ_1 -minimizers are at most m -sparse (when $\mathbb{K} = \mathbb{R}$)
- ▶ Convex optimization program, hence solvable in practice
- ▶ In the real setting, recast as the linear optimization program

$$\underset{\mathbf{c}, \mathbf{z} \in \mathbb{R}^N}{\text{minimize}} \quad \sum_{j=1}^N c_j \quad \text{subject to } A\mathbf{z} = \mathbf{y} \text{ and } -c_j \leq z_j \leq c_j.$$

- ▶ In the complex setting, recast as a second order cone program



Restricted Isometry Constants

Restricted Isometry Constant δ_s = the smallest $\delta > 0$ such that

$$(1 - \delta) \|\mathbf{z}\|_2^2 \leq \|A\mathbf{z}\|_2^2 \leq (1 + \delta) \|\mathbf{z}\|_2^2 \quad \text{for all } s\text{-sparse } \mathbf{z} \in \mathbb{C}^N.$$

Alternative form:

$$\delta_s = \max_{\text{card}(S) \leq s} \|A_S^* A_S - \text{Id}\|_{2 \rightarrow 2}.$$

Suitable CS matrices have small restricted isometry constants.

Typically, $\delta_s \leq \delta_*$ holds for a number of random measurements

$$m \approx \frac{c}{\delta_*^2} s \ln(eN/s).$$

In fact, $\delta_s \leq \delta_*$ imposes $m \geq \frac{c}{\delta_*^2} s$.

Implications of RIP

[Candès (+ et al.); see also Cohen et al., Vershynin et al.]



If $\delta_{2K} < 0.41$, ensured:

1. *Tractable recovery:* All K -sparse x are perfectly recovered via ℓ_1 minimization.
2. *Robust recovery:* For any $x \in R^N$,

$$\|x - \hat{x}\|_{\ell_1} \leq C \|x - x_K\|_{\ell_1} \text{ and } \|x - \hat{x}\|_{\ell_2} \leq C \frac{\|x - x_K\|_{\ell_1}}{K^{1/2}}.$$

3. *Stable recovery:* Measure $y = \Phi x + e$, with $\|e\|_2 < \varepsilon$, and recover

$$\hat{x} = \arg \min \|x'\|_1 \text{ s.t. } \|y - \Phi x'\|_2 \leq \epsilon.$$

Then for any $x \in R^N$,

$$\|x - \hat{x}\|_{\ell_2} \leq C_1 \frac{\|x - x_K\|_{\ell_1}}{K^{1/2}} + C_2 \epsilon.$$



Verifying RIP: How Many Measurements?

- Want RIP of order $2K$ (say) to hold for $M \times N \Phi$
 - difficult to verify for a given Φ
 - requires checking eigenvalues of each submatrix
- Prove ***random*** Φ will work
 - ***iid Gaussian entries***
 - ***iid Bernoulli entries (+/- 1)***
 - ***iid subgaussian entries***
 - ***random Fourier ensemble***
 - ***random subset of incoherent dictionary***
- In each case, **$M = O(K \log N)$** suffices
 - with very high probability, usually $1 - O(e^{-CN})$
 - slight variations on log term
 - some proofs complicated, others simple (more soon)

Optimality

[Candès; Donoho]



- Gaussian Φ has RIP order $2K$ (say) with $M = O(K \log(N/M))$
- Hence, for a given M , for $x \in wI_p$ (i.e., $|x|_{(k)} \sim k^{-1/p}$), $0 < p < 1$, (or $x \in I_1$)

$$\begin{aligned}\|x - \hat{x}\|_{\ell_2} &\leq CK^{-1/2} \|x - x_K\|_{\ell_1} \\ &\leq CK^{1/2-1/p} \\ &\leq C(M/\log(N/M))^{1/2-1/p}\end{aligned}$$

- Up to a constant, these bounds are *optimal*: no other linear mapping to R^M followed by *any* decoding method could yield lower reconstruction error over classes of compressible signals
- Proof (geometric): Gelfand n-widths [Kashin; Gluskin, Garnaev]

Reweighted L1-norm Minimization

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{s.t. } \mathbf{y} = \mathbf{A}\mathbf{x}$$

- Apart from the popular ℓ_1 approximation, consider the following concave approximation:

$$\min_{\mathbf{x} \in \mathbb{R}^n} \sum_{i=1}^n \log(1 + |x_i|/\epsilon) \quad \text{s.t. } \mathbf{y} = \mathbf{A}\mathbf{x},$$

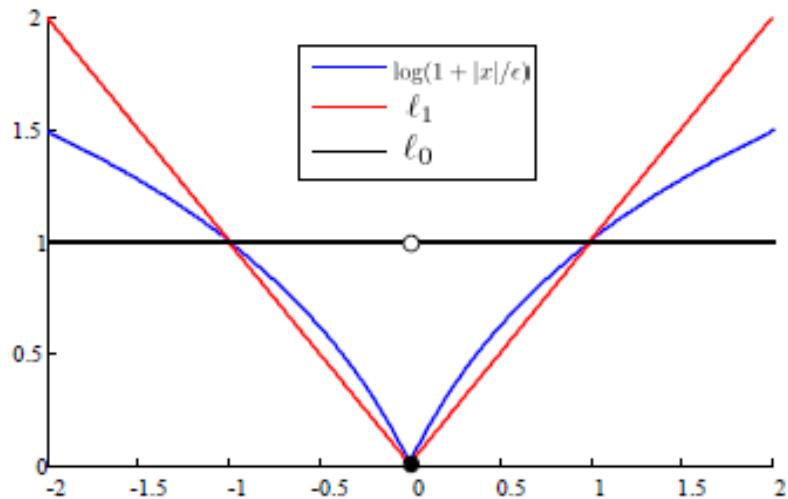


Figure 3: $\log(1 + |x|/\epsilon)$ promotes more sparsity than ℓ_1

Candès, E.J., Wakin, M.B. & Boyd, S.P. Enhancing Sparsity by Reweighted ℓ_1 Minimization. J Fourier Anal Appl 14, 877–905 (2008). <https://doi.org/10.1007/s00041-008-9045-x>



Reweighted L1-norm Minimization

$$\min_{\mathbf{x} \in \mathbb{R}^n} \sum_{i=1}^n \log(1 + |x_i|/\epsilon) \quad \text{s.t. } \mathbf{y} = \mathbf{A}\mathbf{x},$$

which can be equivalently written as

$$\min_{\mathbf{x}, \mathbf{z} \in \mathbb{R}^n} \sum_{i=1}^n \log(z_i + \epsilon) \quad \text{s.t. } \mathbf{y} = \mathbf{A}\mathbf{x}, \quad |x_i| \leq z_i, \quad i = 1, \dots, n \quad (8)$$

- Problem (8) minimizes a concave objective, so it's a special case of DC programming ($g(x) = 0$). Linearizing the concave function at (x^r, z^r) yields

$$(x^{r+1}, z^{r+1}) = \arg \min \sum_{i=1}^n \frac{z_i}{z_i^r + \epsilon} \quad \text{s.t. } \mathbf{y} = \mathbf{A}\mathbf{x}, \quad |x_i| \leq z_i, \quad i = 1, \dots, n$$

- We solve a sequence of reweighted ℓ_1 problems.



Sparse MAP & LASSO

- Choose a sparse prior

$$p(\mathbf{x}) = c \exp(-\lambda \|\mathbf{x}\|_1)$$

- The corresponding MAP estimator is

$$\mathbf{x}_{MAP} = \operatorname{argmin}_{\mathbf{x}} \frac{\|\mathbf{y} - \Phi \mathbf{x}\|^2}{2\sigma^2} + \lambda \|\mathbf{x}\|_1 \quad \text{LASSO}$$

- Improve Robustness w.r.t. the uncertainty of λ
 - Use gamma distribution to model λ and use EM to learn gamma parameters (reweighted 11norm)
 - Consider an alternative formulation

$$\hat{\mathbf{x}} = \operatorname{argmin}_{\mathbf{x}} \|\mathbf{x}\|_1, \text{ s.t. } \|\mathbf{y} - \Phi \mathbf{x}\|^2 \leq \varepsilon$$



Generalized LASSO

- In the generalized LASSO, the recovery of the sparse signal can be formulated as the following convex optimization problem:

$$\min_{\mathbf{x}} \|\mathbf{y} - \Phi \mathbf{x}\|^2 + \sigma \lambda f(\mathbf{x})$$

- $f(\mathbf{x})$ is a convex penalty function that encodes the specific sparse structure
 - For a general sparse signal, we may choose $f(\mathbf{x}) = \|\mathbf{x}\|_1$
 - For block-sparse signal, we may choose

$$f(\mathbf{x}) = \sum_{i=1}^B \|\mathbf{x}_i\|$$

Theorem 1 (aNSE for Generalized ℓ_2 -LASSO). *Suppose the measurement matrix Φ has i.i.d. complex Gaussian entries with zero mean and unit variance and the penalty function $f(\mathbf{x})$ is convex. Let $\tau^* = \operatorname{argmin}_{\tau} \bar{d}(\tau)$. Let $\frac{M}{N} \rightarrow \delta \in (0, \infty)$ and $\frac{d(\tau^*)}{M} \rightarrow d^* \in (0, 1)$. The optimal λ that minimizes the aNSE is given by*

$$\lambda^* \triangleq \operatorname{argmin}_{\lambda} \text{aNSE}(\lambda) = \tau^* \sqrt{M - \bar{d}(\tau^*)},$$

and the corresponding minimum aNSE is given by

$$\text{aNSE}(\lambda^*) = \frac{\bar{d}(\tau^*)}{M - \bar{d}(\tau^*)}.$$

$$\begin{aligned} \text{aNSE}(\lambda) &\triangleq \lim_{\sigma \rightarrow 0} \|\hat{\mathbf{x}} - \mathbf{x}^*\|^2 / \sigma^2 \\ \bar{d}(\tau) &\triangleq \mathbb{E}_{\mathbf{w}} [\|\mathbf{w} - \tau \partial f(\mathbf{x}^*)\|^2] \end{aligned}$$

$f(\mathbf{x}) = \|\mathbf{x}\|_1$ and \mathbf{x} is a complex vector

Theorem 2 (NMSE for Complex LASSO). *Suppose the measurement matrix Φ' has i.i.d. complex Gaussian entries with zero mean and unit variance. Let $\tau^* = \operatorname{argmin}_{\tau} d(\tau)$, where*

$$\begin{aligned} d(\tau) &\triangleq \mathbb{E} [| \varphi(X + Z; \tau) - X |^2] \\ &= \epsilon(1 + \tau^2) + \left[e^{-\tau^2} - 2\tau\sqrt{\pi}Q(\sqrt{2}\tau) \right](1 - \epsilon), \end{aligned}$$

$X \sim (1 - \epsilon)\mathcal{D}(|X|) + \epsilon\mathcal{G}^*(X)$, $Z \sim \mathcal{CN}(0, 1)$, and $Q(\cdot)$ is the Q function (tail distribution function of standard Gaussian). Let $M', N' \rightarrow \infty$ such that $\frac{N'}{M'} \rightarrow \delta' \in (0, \infty)$. Then the optimal LASSO parameter λ that minimizes the NSE is given by

$$\lambda^* \triangleq \operatorname{argmin}_{\lambda} NSE(\lambda) = \tau^* \sqrt{\delta' - d(\tau^*)},$$

and the corresponding NSE is given by

$$NSE(\lambda^*) = \frac{\delta' d(\tau^*)}{(\delta' - d(\tau^*))^+}.$$

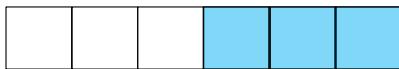
Note that $d(\tau^*)$ is usually called the *minimax risk of the soft thresholding*

Burst-sparse versus Block-sparse

Possible realizations of the support of the example block sparse signal

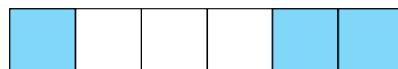
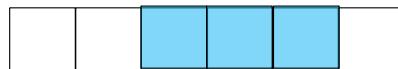
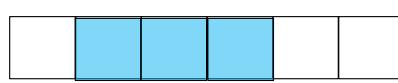
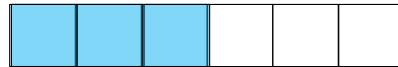


 candidate support



(a) In the example block sparse signal, the support is chosen from 2 non-overlapping candidate supports.

Possible realizations of the support of the example burst sparse signal



(b) In the example burst sparse signal, the support is chosen from 6 overlapping candidate supports.



Greedy CS Recovery Algorithms

Orthogonal Matching Pursuit

Starting with $S^0 = \emptyset$ and $\mathbf{x}^0 = \mathbf{0}$, iterate

$$(\text{OMP}_1) \quad S^{n+1} = S^n \cup \left\{ j^{n+1} := \operatorname{argmax}_{j \in [N]} \left\{ |(A^*(\mathbf{y} - A\mathbf{x}^n))_j| \right\} \right\},$$

$$(\text{OMP}_2) \quad \mathbf{x}^{n+1} = \operatorname{argmin}_{\mathbf{z} \in \mathbb{C}^N} \left\{ \|\mathbf{y} - A\mathbf{z}\|_2, \operatorname{supp}(\mathbf{z}) \subseteq S^{n+1} \right\}.$$

- ▶ The norm of the residual decreases according to

$$\|\mathbf{y} - A\mathbf{x}^{n+1}\|_2^2 \leq \|\mathbf{y} - A\mathbf{x}^n\|_2^2 - |(A^*(\mathbf{y} - A\mathbf{x}^n))_{j^{n+1}}|^2.$$

- ▶ Every vector $\mathbf{x} \neq \mathbf{0}$ supported on S , $\operatorname{card}(S) = s$, is recovered from $\mathbf{y} = A\mathbf{x}$ after at most s iterations of OMP if and only if A_S is injective and

$$(\text{ERC}) \quad \max_{j \in S} |(A^*\mathbf{r})_j| > \max_{\ell \in \overline{S}} |(A^*\mathbf{r})_\ell|$$

for all $\mathbf{r} \neq \mathbf{0} \in \{A\mathbf{z}, \operatorname{supp}(\mathbf{z}) \subseteq S\}$.

Recovery by Orthogonal Matching Pursuit

OMP: Start with $S^0 = \emptyset$ and $\mathbf{x}^0 = \mathbf{0}$, and iterate:

$$S^n = S^{n-1} \cup \{j^n := \operatorname{argmax}_j |(A^*(\mathbf{y} - A\mathbf{x}^{n-1}))_j|\},$$

$$\mathbf{x}^n = \operatorname{argmin}_{\mathbf{z}} \{\|\mathbf{y} - A\mathbf{z}\|_2, \operatorname{supp}(\mathbf{z}) \subseteq S^n\}.$$

If A has ℓ_2 -normalized columns $\mathbf{a}_1, \dots, \mathbf{a}_N$, then

$$\begin{aligned} \|\mathbf{y} - A\mathbf{x}^n\|_2^2 &= \|\mathbf{y} - A\mathbf{x}^{n-1}\|_2^2 - \frac{|(A^*(\mathbf{y} - A\mathbf{x}^{n-1}))_{j^n}|^2}{d(\mathbf{a}_{j^n}, \operatorname{span}[\mathbf{a}_j, j \in S^{n-1}])^2} \\ &\leq \|\mathbf{y} - A\mathbf{x}^{n-1}\|_2^2 - |(A^*(\mathbf{y} - A\mathbf{x}^{n-1}))_{j^n}|^2. \end{aligned}$$

- ▶ Suppose that $\delta_{13s} < 1/6$. Then s -sparse recovery via $12s$ iterations of OMP is stable and robust.



CoSaMP

1) Initialization:

$$\mathbf{x}_{-1} = 0 \quad (\mathbf{x}_J \text{ is the estimate of } \mathbf{x} \text{ at the } J^{\text{th}} \text{ iteration})$$
$$\mathbf{r} = \mathbf{y} \quad (\text{the current residual})$$

2) Loop until convergence

i) Compute the current error: (Note that for Gaussian Φ , $\Phi^T \Phi$ is \sim diagonal)

$$\mathbf{e} = \Phi^* \mathbf{r}.$$

ii) Compute the best $2K$ support set of the error (index set):

$$\Omega = \mathbf{e}_{2K}.$$

iii) Merge the the strongest support sets:

$$T = \Omega \bigcup \text{supp}(\mathbf{x}_{J-1}).$$

iv) Perform a Least-Squares Signal Estimation:

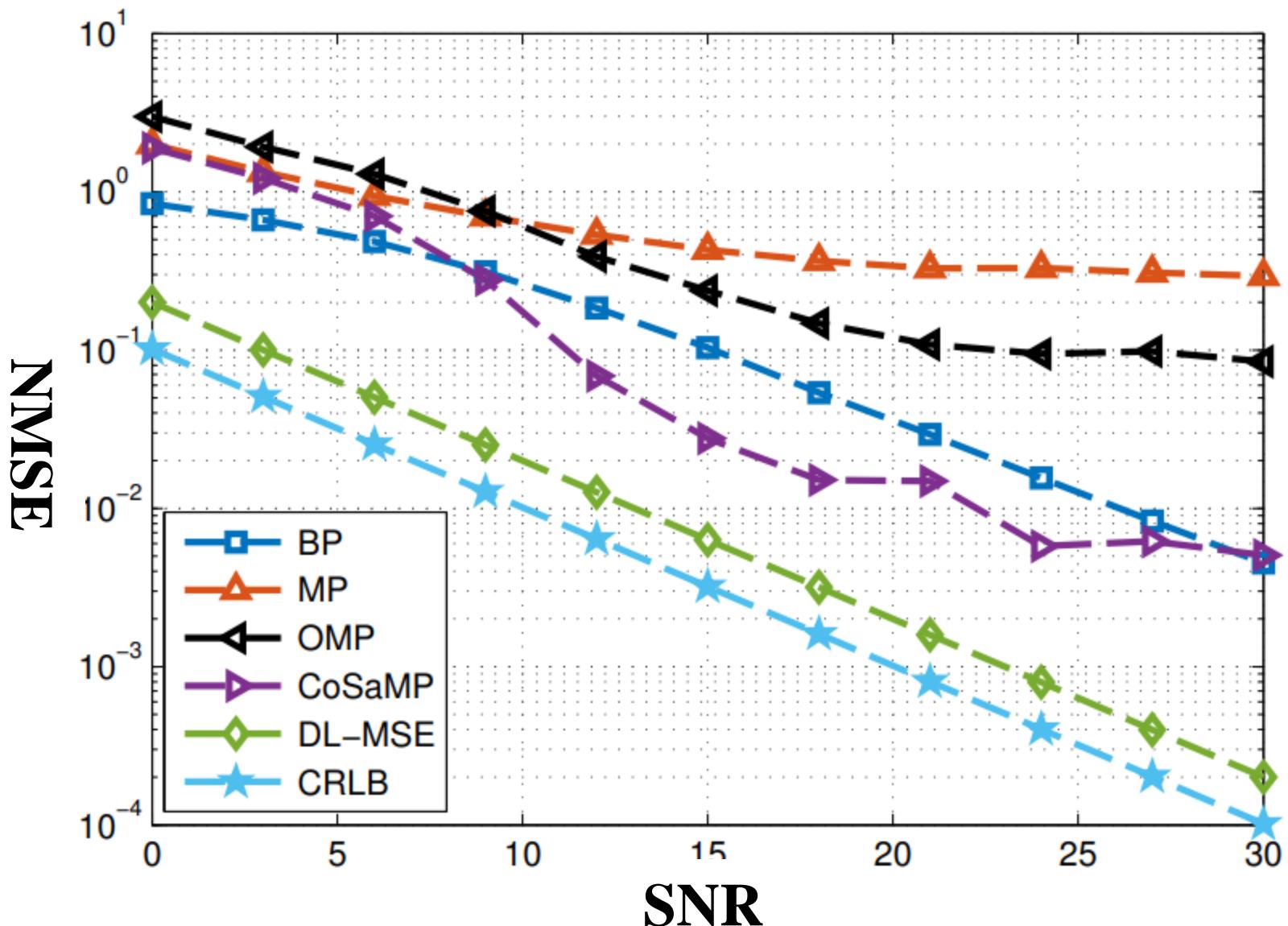
$$\mathbf{b}_{|T} = \Phi_{|T}^\dagger \mathbf{y}, \quad \mathbf{b}_{|T^c} = 0.$$

v) Prune \mathbf{x}_J and compute \mathbf{r} for next round:

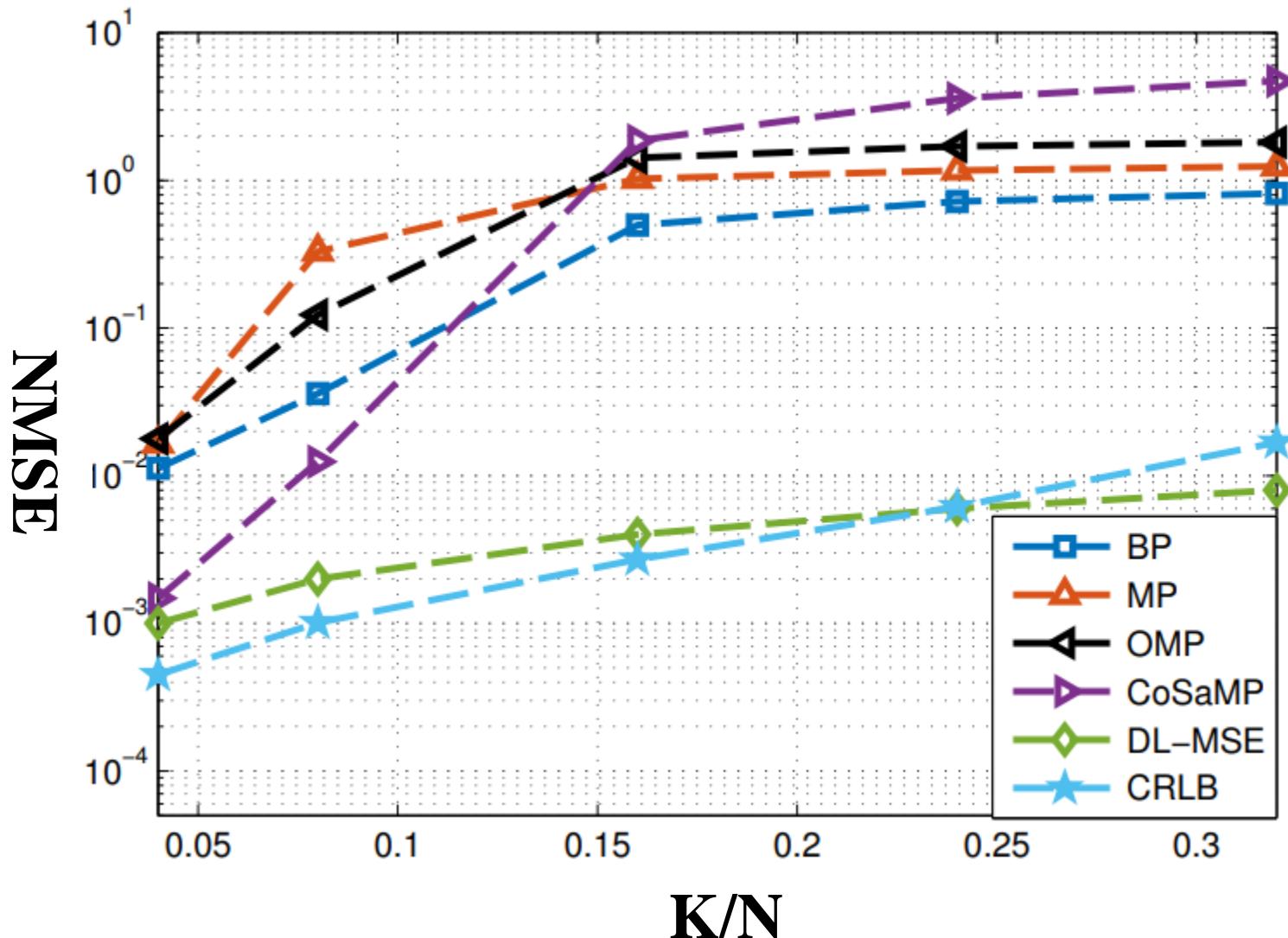
$$\mathbf{x}_J = \mathbf{b}_k,$$

$$\mathbf{r} = \mathbf{y} - \Phi \mathbf{x}_J.$$

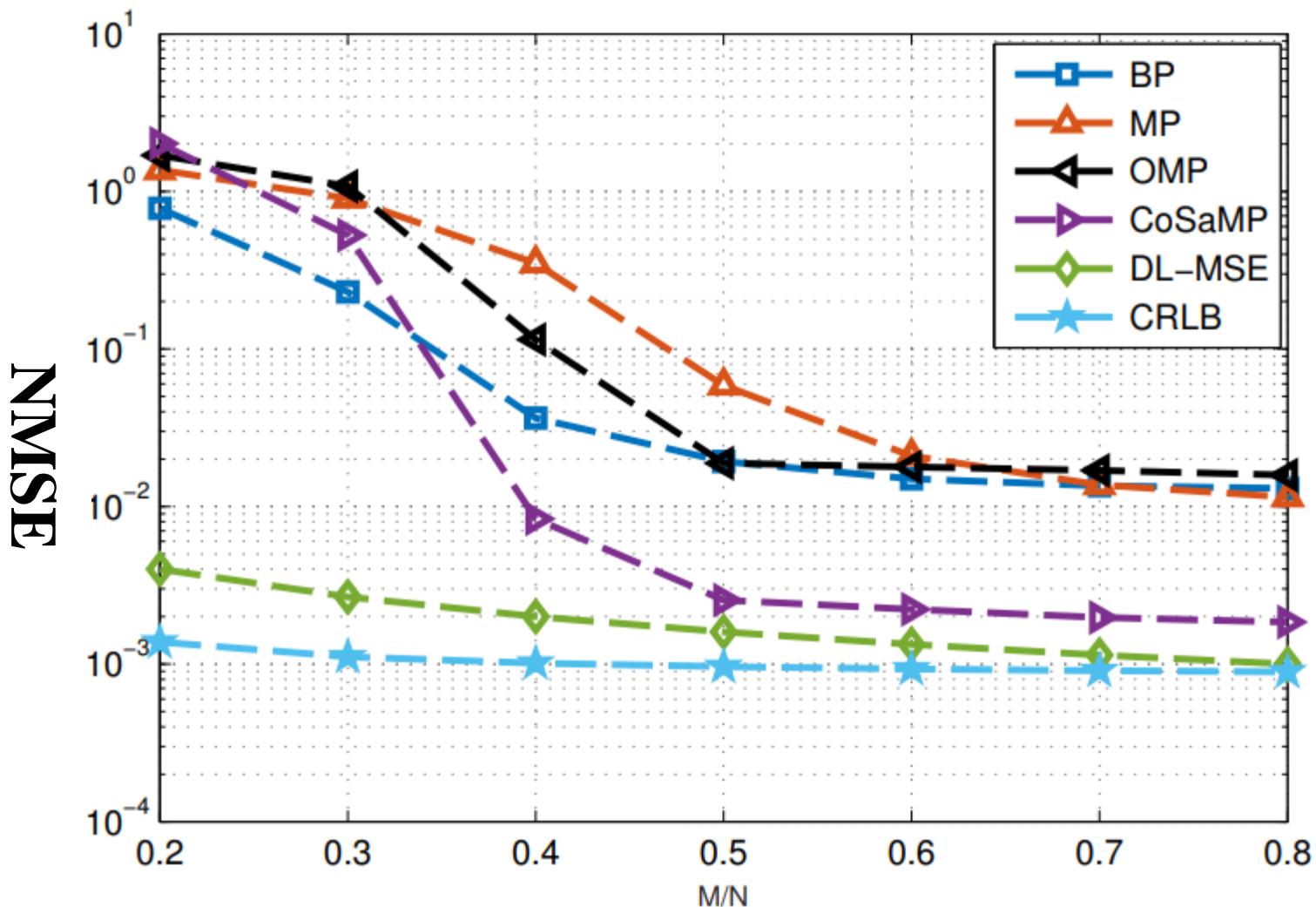
Performance Comparison



Performance Comparison



Performance Comparison





Performance Bounds on CoSaMP

Theorem 2 (Conditions for the Optimality of CoSaMP): Suppose that $\bar{\Phi}$ satisfies the $4K$ -RIP with constant $\delta_{4K} < 0.5$. Apply CoSaMP with input $\mathbf{y}, \Phi, K, \eta$ to obtain $\hat{\mathbf{x}}$ as an approximation of \mathbf{x} . Then $\hat{\mathbf{x}}$ is the unique global optimal solution of Problem \mathcal{P}_A if the following conditions are satisfied:

$$\min_{j \in T_{\mathbf{x}}} |d_j \mathbf{x}(j)| > \eta + \tau(\delta_{4K}) \|\mathbf{n}\|_2, \quad (9)$$

where

$$\begin{aligned} \tau(\delta_{4K}) &= \frac{(\sqrt{2} + 1)\delta_{4K}(\sqrt{2 - 2\delta_{4K}} + \sqrt{1 + \delta_{4K}})}{(1 - \rho)(1 - \delta_{4K})} \\ &\quad + \frac{(2\sqrt{2} + 1)\sqrt{1 + \delta_{4K}}}{(1 - \rho)}, \\ \rho &= \sqrt{\frac{2\delta_{4K}^2(1 + 2\delta_{4K}^2)}{1 - \delta_{4K}^2}} < 1. \end{aligned} \quad (10)$$

$$\mathcal{P}_A : \min_{\mathbf{z}} \|\Phi \mathbf{z} - \mathbf{y}\|_2, \text{ s.t. } \|\mathbf{z}\|_0 = K. \quad (6)$$

The following theorem shows that the solution \mathbf{x}^* of \mathcal{P}_A is a stable estimation of the sparse signal \mathbf{x} if $\bar{\Phi}$ satisfies RIP.

Theorem 1 (Error Bound for Sparse MLE \mathcal{P}_A): Suppose that $\bar{\Phi}$ satisfies the $2K$ -RIP with constant $\delta_{2K} \in (0, 1)$. Let \mathbf{x}^* denote any optimal solution of Problem \mathcal{P}_A . Then we have

$$\|\mathbf{D}(\mathbf{x}^* - \mathbf{x})\|_2 \leq \frac{2}{\sqrt{1 - \delta_{2K}}} \|\mathbf{n}\|_2, \quad (7)$$

where $\mathbf{D} = \text{diag}(d_1, \dots, d_N)$ with $d_j = \|\phi_j\|$. Moreover, the optimal solution of \mathcal{P}_A is uniquely given by $\mathbf{x}^*(T_{\mathbf{x}}) = \Phi^\dagger(T_{\mathbf{x}})\mathbf{y}$ and $\mathbf{x}^*(T_{\mathbf{x}}^c) = \mathbf{0}$ if

$$\min_{j \in T_{\mathbf{x}}} |d_j \mathbf{x}(j)| > \frac{2}{\sqrt{1 - \delta_{2K}}} \|\mathbf{n}\|_2. \quad (8)$$

Extension to Individual Sparsity Levels:

An Liu and Vincent K.N. Lau, 'Joint Interference Mitigation and Data Recovery in Compressive Domain: A Sparse MLE Approach', IEEE Trans. Signal Processing, vol.62, no.19, pp.5184-5195, Oct.1, 2014.



Message Passing CS Recovery Algorithms

- [1] D. L. Donoho, A. Maleki, and A. Montanari, “Message passing algorithms for compressed sensing,” Proc. Nat. Acad. Sci., vol. 106, pp. 18914–18919, 2009.
- [2] M. Bayati and A. Montanari, “The dynamics of message passing on dense graphs, with applications to compressed sensing,” IEEE Trans. Inf. Theory, vol. 57, no. 2, pp. 764–785, Feb. 2011.
- [3] A. Montanari, “Graphical models concepts in compressed sensing,” in Compressed Sensing: Theory and Applications. Cambridge, U.K.: Cambridge Univ. Press, 2012, pp. 394–438.

Compressed Sensing and its Challenges

Significant work beginning ~2006:

- Many successful applications, e.g. MRI
- Fast algorithms
- Scaling laws for measurements required to recover ‘true’ sparse vector

Challenges:

- Most analyses only provide bounds; can be conservative
- Methods specific to LASSO and don’t generalize
- Lacking results related to theoretically-optimal estimate that minimizes MSE $\mathbb{E}\|\hat{\beta} - \beta_0\|^2$ (assuming a known prior on β_0)
 - When can we achieve this and how?
- Limited insight on the distribution of $\hat{\beta}$; needed for inference

Approximate Message Passing (AMP)

Low complexity, scalable algorithm studied extensively for solving high-dimensional linear regression in **compressed sensing**

Benefits of AMP

For certain random matrices,

- Fast convergence
- Asymptotically exact characterization
- Testable conditions for optimality

Though studied extensively for compressed sensing, theory provides insights into many more complex models

(GLMS, logistic regression, phase retrieval, multilayer models, PCA, optimization, ...)

Assuming:

- entries of A are iid $\mathcal{N}(0, \frac{1}{m})$
- dimensions of A are large, $\frac{m}{N} \rightarrow \delta$ (δ is $\Theta(1)$)

AMP ‘derived’ as approximation of loopy belief propagation (BP)
for dense graphs

[Mézard '89, Opper-Winther '96, Kabashima '03, '08, Donoho-Maleki-Montanari '09,
Rangan '11, Krzakala et al '12, Schniter '11, ...]

While BP used to derive, need other techniques to analyze

AMP iteratively produces estimates $\beta^0 = 0, \beta^1, \dots, \beta^t, \dots$

$$r^t = y - A\beta^t + \frac{r^{t-1}}{m} \|\beta^t\|_0$$
$$\beta^{t+1} = \text{soft}(\beta^t + A^T r^t; \theta_t)$$

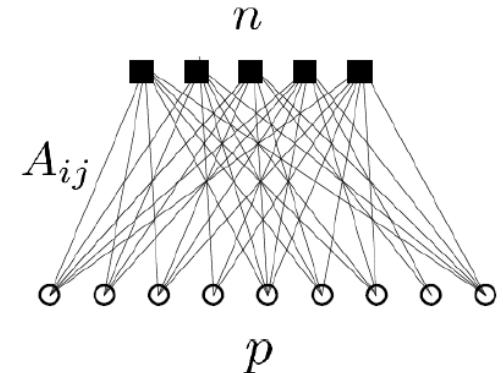
- r^t is the ‘modified residual’ after step t
- soft denoises the *effective observation* to produce β^{t+1}
- θ_t chosen wrt LASSO penalty λ , changing with t

AMP can be thought of as the *limit* of message passing algorithms
when the underlying graph is completely dense.

see A. Montanari, “Graphical models concepts in compressed sensing,” for more details on the connections between AMP and LASSO and the choices of algorithm parameters

1. Gibbs measure

$$\mathbb{P}(dx|y) = \frac{1}{Z(y, \beta)} \left\{ e^{-\beta \frac{\|Ax-y\|^2}{2}} \prod_{i=1}^p e^{-\beta \lambda |x_i|} \right\}$$



2. Message-passing (MP) algorithm at $\beta = \infty$

$$z_{j \rightarrow i}^t = y_j - \sum_{\ell \in [p] \setminus i} A_{i\ell} x_{\ell \rightarrow j}^t$$

$$x_{i \rightarrow j}^{t+1} = \eta \left(\sum_{k \in [n] \setminus j} A_{ki} z_{k \rightarrow i}^t; \lambda \right)$$

3. Look at first order approximation of the messages (AMP)

$$x^0 = 0, \quad x^{t+1} = \eta(x^t + A^T z^t; \frac{\lambda}{1 - \|x^t\|_0/n})$$

$$z^t = y - Ax^t + \frac{\|x^t\|_0}{n} z^{t-1}$$

Onsager reaction term.

In the approximation, only $\mathcal{O}(1)$ and $\mathcal{O}(n^{-1/2})$ terms are kept and all smaller terms are omitted.

Solving the LASSO:

$$\hat{\beta} = \arg \min_{\beta} \|y - A\beta\|^2 + \lambda \|\beta\|_1$$

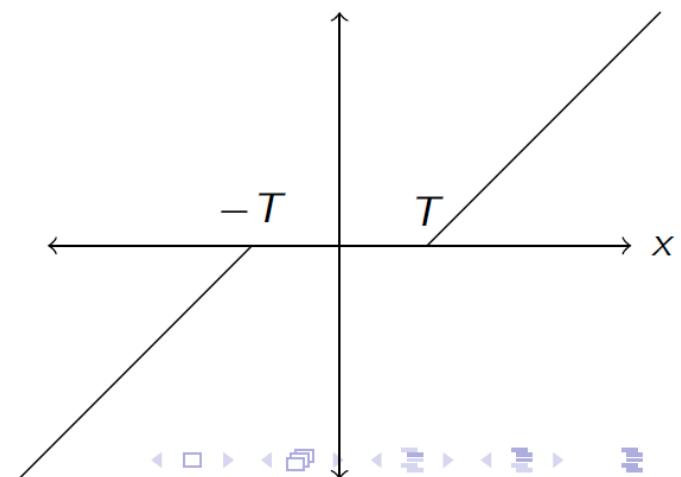
First-order methods: Iteratively generate estimates of β_0 as
 β^1, β^2, \dots

1. Proximal Gradient (aka Iterative Soft-Thresholding)

$$r^t = y - A\beta^t$$

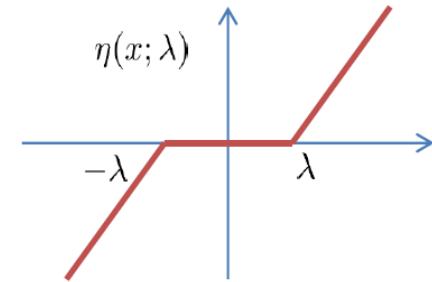
$$\beta^{t+1} = \text{soft}(\beta^t + sA^T r^t; s\lambda)$$

$$\text{soft}(x; T) = \begin{cases} x - T, & x \geq T, \\ 0, & -T < x < T, \\ x + T, & x \leq -T. \end{cases}$$



Some intuition: scalar case ($p=n=1$)

- For $y = x_0 + w$, $w \sim N(0, \sigma^2)$
- LASSO estimate is $\hat{x} = \arg \min_x \left\{ \frac{(y-x)^2}{2} + \lambda|x| \right\}$
- Simple calculus: $\hat{x} = \eta(y; \lambda)$
- Then MSE is



$$\text{mse}(\sigma^2; p_X, \lambda) = \mathbb{E} \left\{ \left[\eta(X + \sigma Z; \lambda) - X \right]^2 \right\}$$

- With independent $Z \sim N(0, 1)$, $X \sim p_X$

AMP iteratively produces estimates $\beta^0 = 0, \beta^1, \dots, \beta^t, \dots$

$$r^t = y - A\beta^t + \frac{r^{t-1}}{m} \|\beta^t\|_0$$
$$\beta^{t+1} = \text{soft}(A^T r^t + \beta^t; \theta_t)$$

With the assumptions:

- Entries of A are iid $\mathcal{N}(0, \frac{1}{m})$
- Dimensions of A are large, $\frac{m}{N} \rightarrow \delta$ (δ is constant)

The *momentum* term in r^t ensures that asymptotically

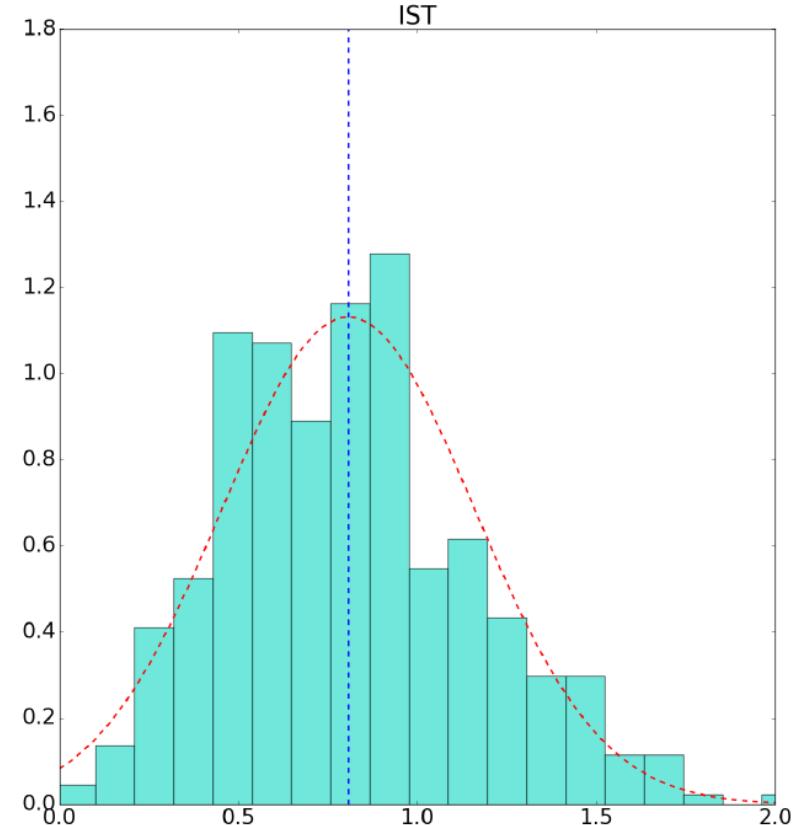
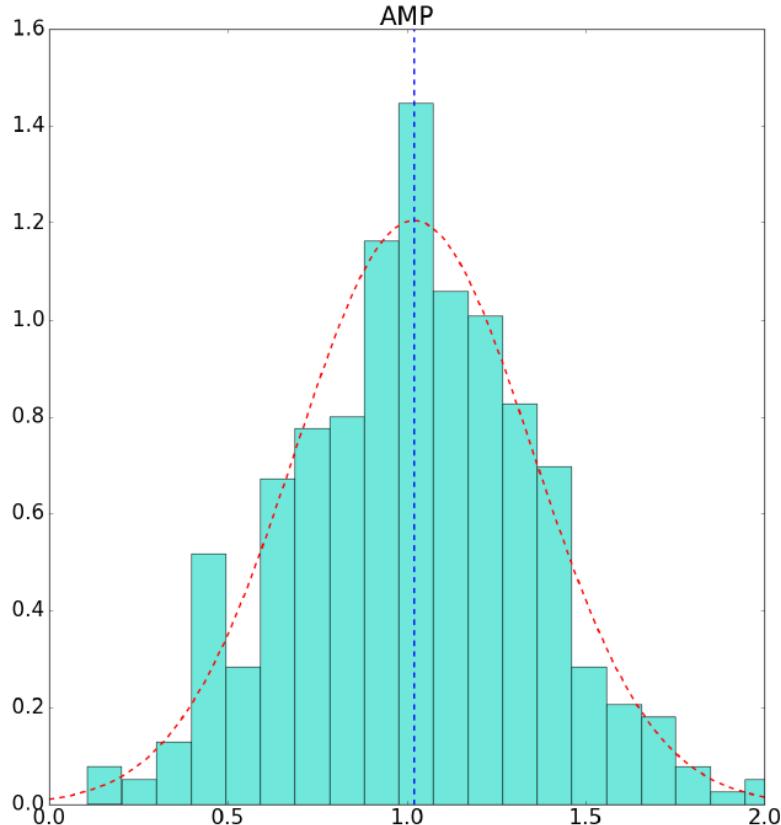
$$A^T r^t + \beta^t \approx \beta_0 + \tau_t Z \quad \text{where } Z \text{ is } \mathcal{N}(0, 1)$$

\Rightarrow Effective observation $A^T r^t + \beta^t$ is true signal observed in independent Gaussian noise with τ_t predicted by **state evolution**.

Example: $y = A\beta_0$

$A : m \times N = 2000 \times 4000$; β_0 has 500 non-zeros \sim iid unif ± 1

Histogram of $A^T r^t + \beta^t$ at indices where $\beta_0 = +1$ at $t = 10$



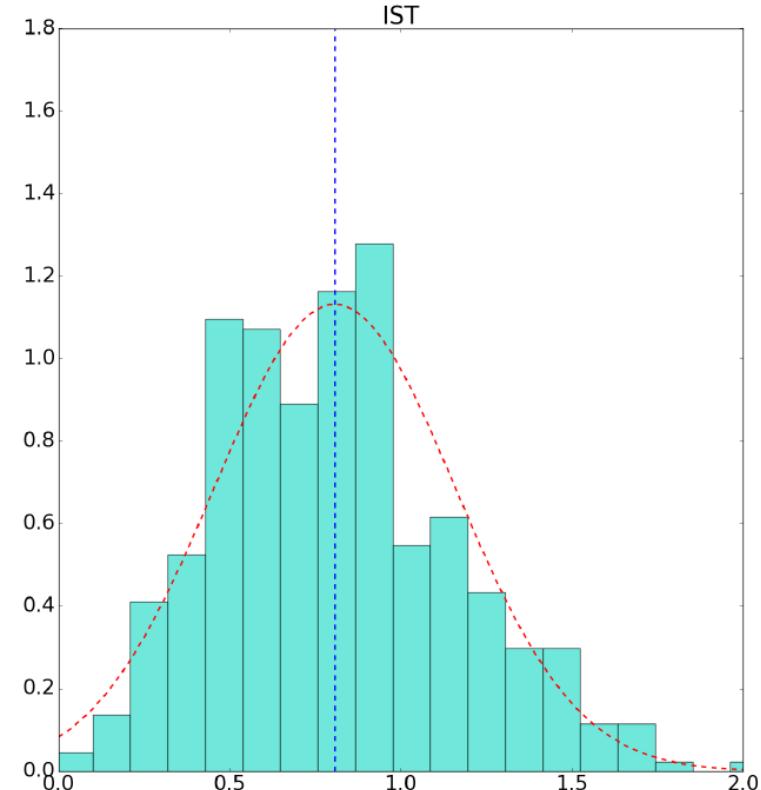
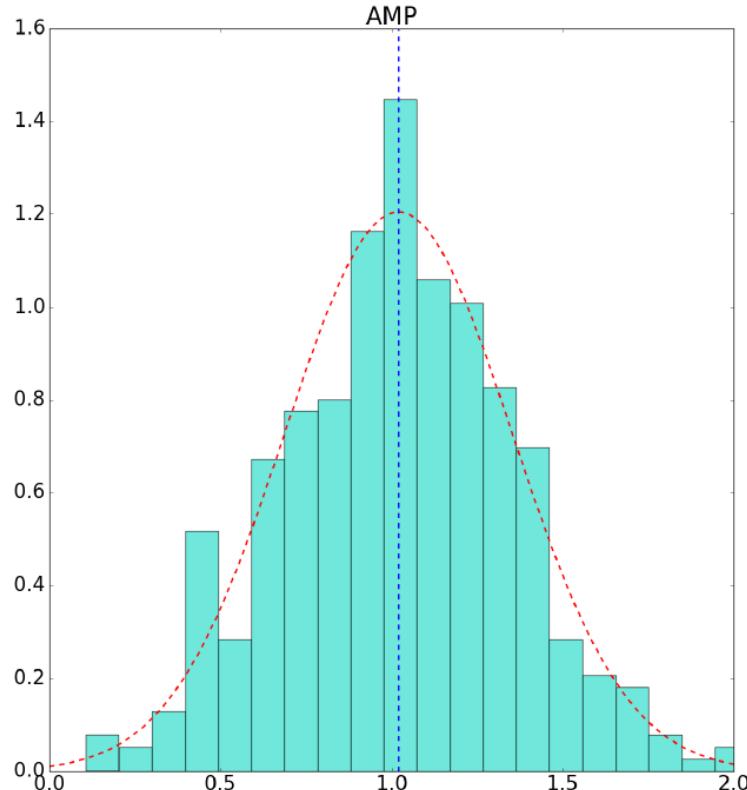
with $r^t = y - A^T \beta^t + r^{t-1} \frac{\|\beta^t\|_0}{m}$

with $r^t = y - A^T \beta^t$

Example: $y = A\beta_0$

$A : m \times N = 2000 \times 4000$; β_0 has 500 non-zeros \sim iid unif ± 1

Histogram of $A^T r^t + \beta^t$ at indices where $\beta_0 = +1$ at $t = 10$



- **Here:** empirical observation at a single t for specific m, N
- **Later:** rigorous proof that statistical properties exact in limit of m, N for all t

General AMP Framework

In the talk so far:

- LASSO motivated by a sparse signal (and unknown signal prior distribution)
- Goal to minimize LASSO cost
- Note: theorem doesn't require sparsity of the assumed prior.
This suggests....

First generalization:

- Known signal prior distribution (sparsity-inducing or not)
- Goal to minimize mean squared error (MSE) of estimate

Let $y = A\beta_0 + w$, $\beta_0 \stackrel{i.i.d.}{\sim} p_\beta$, $w \stackrel{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$

$$r^t = y - A\beta^t + \frac{r^{t-1}}{m} \sum_{i=1}^N \eta'_{t-1}(A^T r^{t-1} + \beta^{t-1})_i$$

$$\beta^{t+1} = \eta_t(A^T r^t + \beta^t)$$

Function η_t chosen to denoise *effective observation* producing β^{t+1}

KEY: For large m, N , at each time step t

$$A^T r^t + \beta^t \approx \beta_0 + \tau_t Z \quad \text{where } Z \text{ is } \mathcal{N}(0, 1)$$

- p_β known: Bayes-optimal η_t choice minimizes $\mathbb{E}\|\beta_0 - \beta^{t+1}\|^2$. Equals

$$\eta_t(s) = \mathbb{E}[\beta_0 \mid \beta_0 + \tau_t Z = s]$$

- p_β unknown: partial knowledge about β_0 can guide η_t choice.

e.g., see A. Montanari, "Graphical models concepts in compressed sensing," for more details

To summarize:

LASSO for compressed sensing:

- Sparse signal (unknown signal prior distribution)
- Goal to minimize LASSO cost
- Soft-threshold denoiser $\eta(\cdot)$

First generalization:

- Known signal prior distribution (sparsity-inducing or not)
- Goal to minimize mean squared error (MSE)
- Denoiser $\eta_t(s) = \mathbb{E}[\beta_0 | \beta_0 + \mathcal{N}(0, \tau_t) = s]$ when prior known

In both cases, as $N \rightarrow \infty$ with t fixed, AMP admits an *asymptotically exact characterization* via **state evolution**, such that $A^T r^t + \beta^t \approx \beta_0 + \mathcal{N}(0, \tau_t)$.

Under the assumption on the **effective observation**:

$$\beta^t + A^T r^t \approx \beta_0 + \tau_t Z, \quad Z \sim \mathcal{N}(0, \mathbb{I}).$$

If τ_1, τ_2, \dots is decreasing, getting a more ‘pure’ view of β_0 as algorithm iterates. The **state evolution** computes τ_t^2 .

State evolution equations

Set $\tau_0^2 = \sigma^2 + \frac{1}{\delta} \mathbb{E} \|\beta\|^2$,

$$\tau_t^2 = \sigma^2 + \frac{1}{\delta} \mathbb{E} \|\beta - \eta_{t-1}(\beta + \tau_{t-1} Z)\|^2$$

where $Z \sim \mathcal{N}(0, 1)$ independent of $\beta \sim p_\beta$.

State evolution is a scalar recursion that allows for prediction of the performance of AMP at any iteration. Now we make this rigorous.

Intuition behind the State Evolution

Using the assumption: $\mathbf{A}^T \mathbf{r}^t + \boldsymbol{\beta}^t \approx \boldsymbol{\beta}_0 + \tau_t \mathbf{Z}, \mathbf{Z} \sim \mathcal{N}(0, \mathbf{I}_N)$

$$\begin{aligned}\tau_t \mathbf{Z} &= \boldsymbol{\beta}_0 - \boldsymbol{\beta}^t - \mathbf{A}^T \mathbf{r}^t \\&= \boldsymbol{\beta}_0 - \boldsymbol{\beta}^t - \mathbf{A}^T (\mathbf{y} - \mathbf{A} \boldsymbol{\beta}^t) \\&= \boldsymbol{\beta}_0 - \boldsymbol{\beta}^t - \mathbf{A}^T (\mathbf{A} \boldsymbol{\beta}_0 + \mathbf{w}) + \mathbf{A}^T \mathbf{A} \boldsymbol{\beta}^t \\&= (\mathbf{I}_N - \mathbf{A}^T \mathbf{A}) (\boldsymbol{\beta}_0 - \boldsymbol{\beta}^t) - \mathbf{A}^T \mathbf{w}\end{aligned}$$

Expectation for l_2 norm:

$$\mathbb{E} (\|\tau_t \mathbf{Z}\|^2) = \mathbb{E} \left(\left\| (\mathbf{I}_N - \mathbf{A}^T \mathbf{A}) (\boldsymbol{\beta}_0 - \boldsymbol{\beta}^t) \right\|^2 \right) + \mathbb{E} \left(\|\mathbf{A}^T \mathbf{w}\|^2 \right)$$

elements $\sim \mathcal{N}(0, 1/m)$.

$$\begin{array}{ccc} \downarrow & \downarrow & \downarrow \\ N\tau_t^2 & \frac{N^2}{m}\sigma_\beta^2 = \frac{N}{\delta}\sigma_\beta^2 & N\sigma^2 \end{array}$$

Conclusion:

$$\tau_t^2 = \sigma^2 + \frac{1}{\delta}\sigma_\beta^2 \quad \sigma_\beta^2 = \mathbb{E} \left(\|\boldsymbol{\beta} - \eta_{t-1}(\boldsymbol{\beta} + \mathcal{N}(0, \tau_{t-1}))\|^2 \right), \boldsymbol{\beta} \sim p(\boldsymbol{\beta})$$

Assumptions for Performance Guarantees

We make the following assumptions:

- **Measurement matrix**: i.i.d. $\sim \mathcal{N}(0, 1/m)$.
- **Signal**: i.i.d. $\sim p_\beta$, sub-Gaussian.
- **Measurement noise**: i.i.d. $\sim p_W$, sub-Gaussian, $\mathbb{E}[w_i^2] = \sigma^2$.
- **De-noising Functions** η_t : Lipschitz continuous with weak derivative η'_t that's differentiable except possibly at a finite num. of points, with bounded derivative everywhere it exists.

Pseudo-Lipschitz (PL) Loss Functions

A function $\phi : \mathbb{R}^m \rightarrow \mathbb{R}$ is PL if there exists a constant $L > 0$ such that for all $x, y \in \mathbb{R}^m$,

$$|\phi(x) - \phi(y)| \leq L(1 + \|x\| + \|y\|)\|x - y\|.$$

E.g. $\phi(x) = \|x\|_2^2$ (squared-error loss), or $\phi(x) = \|x\|_1$ (ℓ_1 loss).

Performance Guarantees

Theorem (Rush, Venkataraman '18)

Under the assumptions of the previous slide, for any PL function $\phi : \mathbb{R}^2 \rightarrow \mathbb{R}$, $\Delta \in (0, 1)$, and $t \geq 0$,

$$\begin{aligned} P\left(\left|\frac{1}{N} \sum_{i=1}^N \phi(\beta_i^{t+1}, \beta_{0_i}) - \delta \mathbb{E}[\phi(\eta_t(\beta + \tau_t Z), \beta)]\right| \geq \Delta\right) \\ \leq K_t e^{-\kappa_t N \Delta^2}, \end{aligned}$$

for $Z \sim \mathcal{N}(0, 1)$, $\beta \sim p_\beta$ independent with constants K_t, κ_t .

For PL loss functions, can essentially consider AMP estimate β^{t+1} as having i.i.d. entries with each entry $\sim \eta_t(\beta + \tau_t Z)$.

Performance Guarantees

Theorem (Rush, Venkataraman '16)

Under the assumptions of the previous slide, with constants K_t, κ_t , for $\Delta \in (0, 1)$ and $t \geq 0$,

$$P \left(\left| \frac{1}{N} \|\beta^{t+1} - \beta_0\|^2 - \delta(\tau_{t+1}^2 - \sigma^2) \right| \geq \Delta \right) \leq K_t e^{-\kappa_t N \Delta^2}.$$

- Refines the asymptotic result proved by [Bayati-Montanari '11]
- The finite-sample result above implies the asymptotic result (via Borel-Cantelli), i.e. with $\delta = m/N$

$$\lim_{N \rightarrow \infty} \frac{1}{N} \|\beta^{t+1} - \beta_0\|^2 \stackrel{\text{a.s.}}{=} \delta(\tau_{t+1}^2 - \sigma^2).$$

AMP Extensions/Generalizations

- Non-Gaussian noise distributions (**GAMP**) [Rangan '11]
- Different measurement matrices:
 - Sub-Gaussian [Bayati-Lelarge-Montanari '15]
 - Right orthogonally-invariant (**VAMP**) [Schniter-Rangan-Fletcher '16, '17]
 - Spatially-coupled (for improved MSE performance)
[Donoho-Javanmard-Montanari '13, Rush-Hsieh-Venkataraman '18]
- Signals with dependent entries and non-separable denoisers [Ma-Rush-Baron '17, Berthier-Montanari-Nguyen '17]

AMP Extensions/Generalizations

Different measurement models:

- **Bilinear Models** [Parker-Schniter-Cevhar '14]
 - **Multiple Measurement Vectors** [Ziniel-Schniter '13]
 - **Matrix Factorization** [Kabashima-Krzakala-Mézard-Sakata-Zdeborová '16]
 - **Blind Deconvolution**
- **Low-rank Matrix Estimation** [Rangan-Fletcher '12, Lesieur-Krzakala-Zdeborová '15]
 - **Principle Component Analysis** [Deshpandre-Montanari '14, Montanari-Richard '16]
 - **Stochastic Block Model** [Deshpandre-Abbe-Montanari '16]
 - **Replica Method** [Barbier-Dia-Macris-Krzakala-Lesieur-Zdeborová '15]

AMP Summary

$$y = A\beta_0 + w$$

$$r^t = y - A\beta^t + \frac{r^{t-1}}{m} \sum_{i=1}^N \eta'_{t-1}(A^T r^{t-1} + \beta^{t-1})_i$$

$$\beta^{t+1} = \eta_t(A^T r^t + \beta^t)$$

AMP: First-order iterative algorithm

- Theory assumes i.i.d. (sub)Gaussian A
- Sharp theoretical guarantees determined by simple scalar iteration. E.g.,

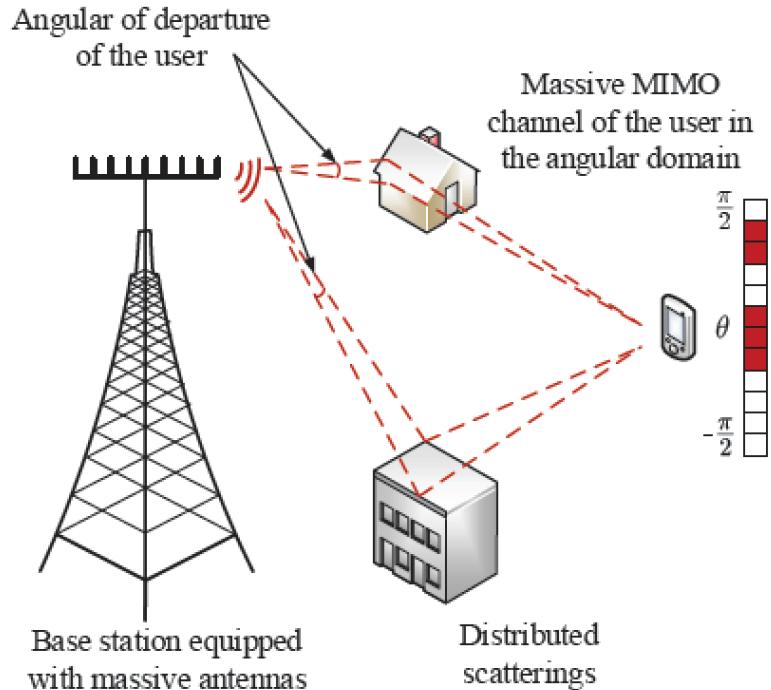
$$\frac{1}{N} \|\beta_0 - \beta^{t+1}\|^2 \approx \delta(\tau_{t+1}^2 - \sigma^2)$$

- AMP can be run even without knowing prior p_β
(our result shows that τ_t^2 concentrates on $\|r^t\|^2/m$)
- Knowing p_β can help choose a good denoiser η_t



Applications to Channel Estimation

System Model for Downlink Massive MIMO



- Transmitter: BS with N antennas.
- Receiver: Users with a single antenna.
- Pilots Transmission: T time slots.
- The received signal is given by

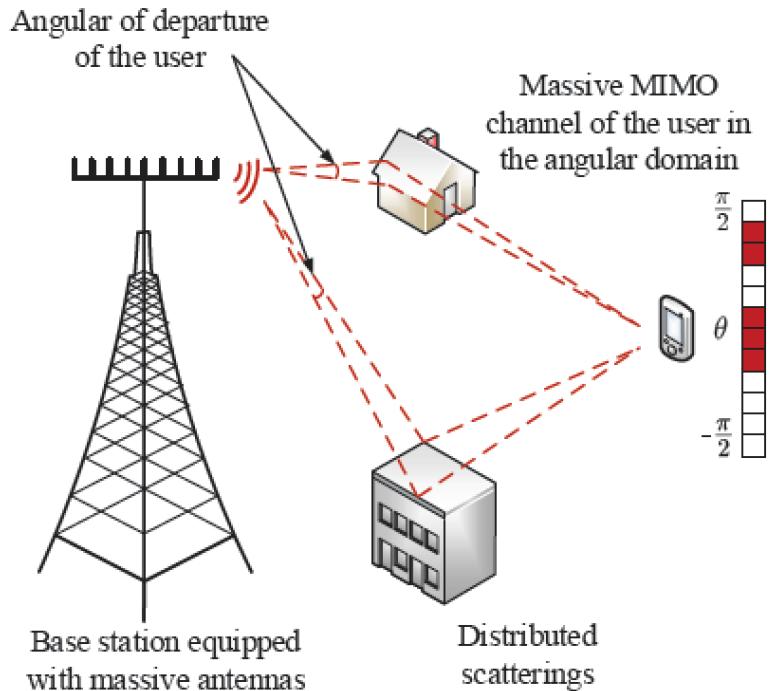
$$\mathbf{y} = \mathbf{X}\tilde{\mathbf{h}} + \mathbf{w}$$

$\mathbf{X} \in \mathbb{C}^{T \times N}$: Transmitted pilots in T time slots.

$\tilde{\mathbf{h}} \in \mathbb{C}^{N \times 1}$: Channel in the **spatial** domain.

$\mathbf{w} \in \mathbb{C}^{N \times 1}$: Gaussian noise.

Clustered Sparsity of Massive MIMO Channel



- Channel can be represented in the angular domain by Fourier transform, then the system model is rewritten as

$$\mathbf{y} = \mathbf{X}\mathbf{F}^H \mathbf{h} + \mathbf{w}$$

$\mathbf{F} \in \mathbb{C}^{N \times N}$: Discrete Fourier transform matrix.

$\mathbf{h} \in \mathbb{C}^{N \times 1}$: Channel in the **angular** domain.

- Due to the limited scatterings, physical channel \mathbf{h} is **sparse and clustered** in the angular domain of the transmit antenna array.

Challenge for Massive MIMO

$$\begin{matrix} \mathbf{y} \\ = \end{matrix} \quad \begin{matrix} \mathbf{A} \\ \left[\begin{array}{c|c} \hline & N \\ \hline T & \\ \hline \end{array} \right] \end{matrix} \quad \begin{matrix} \mathbf{h} \\ + \end{matrix} \quad \begin{matrix} \mathbf{w} \end{matrix}$$

- The linear system model can be further rewritten as

$$\mathbf{y} = \mathbf{A}\mathbf{h} + \mathbf{w}$$

- Traditional schemes: LS, LMMSE, $T > N$.
- Challenge: For a massive MIMO system with N transmit antennas, T is required to be no less than N .

New Scheme: Compressed Sensing

$$\mathbf{y} = \mathbf{A} \mathbf{h} + \mathbf{w}$$

The diagram illustrates the mathematical model of compressed sensing. On the left, a vertical vector \mathbf{y} is shown. To its right is an equals sign followed by a large gray rectangular box labeled \mathbf{A} . Inside this box, there is a coordinate system with a horizontal axis labeled N and a vertical axis labeled T . To the right of the box is a vertical vector \mathbf{h} , followed by a plus sign, and then another vertical vector \mathbf{w} .

- By exploiting the channel sparsity in the angular domain, compressed sensing can reduce the pilot overhead greatly.
 - D. L. Donoho, "Compressed sensing," IEEE Transactions on Information Theory, vol. 52, no. 4, pp. 1289-1306, April 2006, doi: 10.1109/TIT.2006.871582.
- Many algorithms are developed:
 - Least absolute shrinkage and selection operator (LASSO)
 - Orthogonal matching pursuit (OMP)
 - Compressive sampling matching pursuit (CoSaMP)
 - Iterative soft thresholding (ISTA)
 - Approximate message passing (AMP)
 - Existing issues:
high complexity, slow convergence, choice of sensing matrix, limited storage and computational resources at user devices...



Compressed Sensing Algorithms

■ l_0 -minimization

- Non-convex problem:

$$\hat{\mathbf{h}} = \arg \min_{\mathbf{h}} \left\{ \frac{1}{\lambda} \|\mathbf{y} - \mathbf{A}\mathbf{h}\|_2^2 + \|\mathbf{h}\|_0 \right\}. \quad (1)$$

■ l_1 -minimization

- Least absolute shrinkage and selection operator (LASSO)
- Convex programming with polynomial time
- Scalability is still an issue.

$$\hat{\mathbf{h}} = \arg \min_{\mathbf{h}} \left\{ \frac{1}{\lambda} \|\mathbf{y} - \mathbf{A}\mathbf{h}\|_2^2 + \|\mathbf{h}\|_1 \right\}. \quad (2)$$

- J. A. Tropp and A. C. Gilbert, "Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit," IEEE Transactions on Information Theory, vol. 53, no. 12, pp. 4655-4666, Dec. 2007, doi: 10.1109/TIT.2007.909108.
- Tibshirani, Robert. "Regression shrinkage and selection via the LASSO." Journal of the Royal Statistical Society: Series B (Methodological) 58.1 (1996): 267-288.

Approximate Message Passing (AMP)

$$\hat{\mathbf{h}} = \arg \min_{\mathbf{h}} \left\{ \frac{1}{\lambda} \|\mathbf{y} - \mathbf{A}\mathbf{h}\|_2^2 + \|\mathbf{h}\|_1 \right\}$$

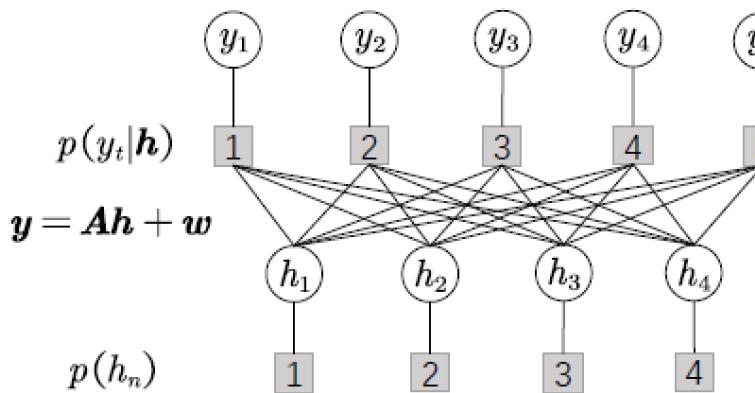


Assign a probability model

$$\arg \max_{\mathbf{h}} \exp \left\{ -\frac{1}{\lambda} \|\mathbf{y} - \mathbf{A}\mathbf{h}\|_2^2 - \|\mathbf{h}\|_1 \right\} = \arg \max_{\mathbf{h}} p(\mathbf{h}|\mathbf{y})$$



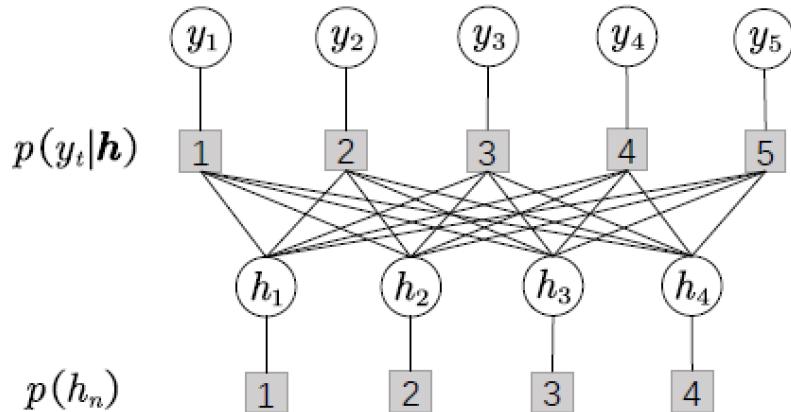
Factor Graph + Belief Propagation



- F. R. Kschischang, B. J. Frey and H. A. Loeliger, "Factor graphs and the sum-product algorithm," IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 498-519, Feb 2001, doi: 10.1109/18.910572.
- Donoho, David L. et al. "Message-passing algorithms for compressed sensing." Proceedings of the National Academy of Sciences 106 (2009): 18914 - 18919.

AMP Algorithm

- In general, the sensing matrix \mathbf{A} is a full matrix.
- The complexity of message passing is proportional to the total number of edges in the graph.



AMP takes two approximations to reduce complexity:

- Gaussian approximation → only need to track mean and variance
- First-order Taylor approximation

- The AMP algorithm:

$$\mathbf{u}^t = \mathbf{y} - \mathbf{A}\mathbf{h}^t + \boxed{\frac{1}{N} \sum_{i=1}^N \eta'(\mathbf{A}^T \mathbf{u}^{t-1} + \mathbf{h}^{t-1}) \cdot \mathbf{u}^{t-1}} \quad (5)$$

$$\mathbf{h}^{t+1} = \eta(\mathbf{A}^T \mathbf{u}^t + \mathbf{h}^t) \quad \text{Onsager reaction term} \quad (6)$$



Part III: Bayesian Inference

- Variational Bayesian Inference
- Sparse Bayesian Learning
- Turbo CS
- Turbo VBI
- Dynamic Bayesian Learning
- Applications to Wireless Communications



Variational Bayesian Inference

- [1] C. W. Fox and S. J. Roberts. "A tutorial on variational Bayesian inference." *Artificial Intelligence Review* 38.2 (2012): 85-95.
- [2] D. G. Tzikas, A. C. Likas, and N. P. Galatsanos, "The Variational Approximation for Bayesian Inference," *Signal Processing Magazine IEEE*, vol. 25, no. 6, pp. 131–146, 2008.



The Mean-Field VBI Algorithm

2.2 The Mean-Field VBI Algorithm

- Define a complete variable $\alpha = \begin{bmatrix} \mathbf{x} \\ \mathbf{s} \end{bmatrix}$ that incorporates all unknown variables.
- A set of virtual densities $\{q(\alpha_j) | \forall \alpha_j \in \alpha\}$ are used to approximate $p(\alpha_j | \mathbf{z})$.
- The optimal variational approximations $\hat{q}(\alpha_j)$, $\forall \alpha_j \in \alpha$, is defined as [1]

$$\hat{q}(\alpha) = \arg \min_{q(\alpha)} \int q(\alpha) \ln \frac{q(\alpha)}{p(\alpha | \mathbf{z})} d\alpha \quad (7)$$

$$\text{s.t. } q(\alpha) = \prod_{j | \forall \alpha_j \in \{\alpha\}} q(\alpha_j), \quad (8)$$

$$\int q(\alpha_i) d\alpha_i = 1, \forall \alpha_i \in \{\alpha\}. \quad (9)$$

Kullback-Leibler divergence, $D_{\text{KL}}[q || p] := \int q(\alpha) \ln \frac{q(\alpha)}{p(\alpha | \mathbf{z})} d\alpha$

$$D_{\text{KL}}[q || p] \geq 0, \quad (10)$$

$$D_{\text{KL}}[q || p] = 0, \text{ iif } q(\alpha) = p(\alpha | \mathbf{z}). \quad (11)$$



The Mean-Field VBI Algorithm

2.2 The Mean-Field VBI Algorithm

- Isolate the term w.r.t. α_j against $\tilde{\alpha}_j$ [2],

$$D_{\text{KL}}[q||p] = \underbrace{\int q(\alpha_j) \ln \frac{\pi(\alpha_j)}{q(\alpha_j)} d\alpha_j}_{D_{\text{KL}}[q(\alpha_j)||\pi(\alpha_j)]} + \underbrace{\ln C - H[\tilde{\alpha}_j]}_{\perp \alpha_j}, \quad (12)$$

$$\pi(\alpha_j) := C^{-1} \exp \left(\langle \ln p(\alpha, z) \rangle_{\tilde{\alpha}_j} \right), \quad (13)$$

$$H[\tilde{\alpha}_j] = \int q(\tilde{\alpha}_j) \log q(\tilde{\alpha}_j) d\tilde{\alpha}_j \quad (14)$$

where $H[\tilde{\alpha}_j]$ is the Shannon entropy that is independent of α_j ,



The Mean-Field VBI Algorithm

2.2 The Mean-Field VBI Algorithm

- Using the variational calculus, the optimal variational approximation $q(\alpha_j)$ of each individual variable α_j , $\forall \alpha_j \in \{\alpha\}$, is derived as

$$q(\alpha_j) \propto \exp \left(\langle \ln p(z, \alpha) \rangle_{\tilde{\alpha}_j} \right). \quad (15)$$

where the symbol $\langle u(\alpha_j) \rangle_{\alpha_j}$ w.r.t. a generic function $u(\alpha_j)$ is expressed as

$$\langle u(\alpha_j) \rangle_{\alpha_j} = \int u(\alpha_j) q(\alpha_j) d\alpha_j, \quad (16)$$

and $\tilde{\alpha}_j$ is defined as the complementary variable of α_j , i.e., $\tilde{\alpha}_j = \alpha \setminus \alpha_j$, by introducing set partition $\{\alpha\} = \{\alpha_j, \tilde{\alpha}_j\}$.

- Specifically,

$$\hat{q}(x) = \mathcal{N}(z | \langle h(x, s) \rangle_s, W). \quad (17)$$

$$\hat{q}(s) = p(s) \mathcal{N}(z | \langle h(x, s) \rangle_x, W). \quad (18)$$



The Mean-Field VBI Algorithm

$$\mathbf{z} = \mathbf{h}(\mathbf{x}, \mathbf{s}) + \mathbf{n}$$

2.2 The Mean-Field VBI Algorithm

- In such a case, the approximate MAP estimate of \mathbf{x} becomes

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} \exp \left(\langle \ln p(\mathbf{z}, \mathbf{x}, \mathbf{s}) \rangle_s \right) \quad (19)$$

$$= \arg \max_{\mathbf{x}} \mathcal{N}(\mathbf{z} | \langle \mathbf{h}(\mathbf{x}, \mathbf{s}) \rangle_s, \mathbf{W}) \quad (20)$$

- The approximate MAP estimate of \mathbf{s} is given by

$$\hat{\mathbf{s}} = \arg \max_{\mathbf{s}} \exp \left(\langle \ln p(\mathbf{z}, \mathbf{x}, \mathbf{s}) \rangle_x \right) \quad (21)$$

$$= \arg \max_{\mathbf{s}} p(\mathbf{s}) \mathcal{N}(\mathbf{z} | \langle \mathbf{h}(\mathbf{x}, \mathbf{s}) \rangle_x, \mathbf{W}) \quad (22)$$

- Approximate MMSE estimation (posteriori expectation) is the other popular choice.
- At each iteration, when updating $q(\alpha_j)$ we assume the other approximation $q(\alpha_k)$, $\forall k \neq j$, are determined and keep fixed.



The Mean-Field VBI Algorithm

Algorithm 1: A generic mean-field VBI algorithm

Input : $\{z\}$.

- 1 Collect the measurement signal z .
- 2 Initialize variational states $q(x)$ and $q(s)$.
- 3 **While** not converge **do**
- 4 Update $q(x) = \mathcal{N}(z | \langle h(x, s) \rangle_s, W)$.
- 5 Update $q(s) = p(s) \mathcal{N}(z | \langle h(x, s) \rangle_x, W)$.
- 6 **End**
- 7 Return the estimation $\hat{x}_{\text{MAP}} = \arg \max_x q(x)$.
- 8 Return the estimation

$$\hat{s}_{\text{MMSE}} = \int s q(s) ds. \quad (23)$$

Output: \hat{x} and \hat{s} .



The Mean-Field VBI Algorithm

2.3 Merits, Challenges and Strategies

- Merits:
 - Distributed calculation (reduced overhead).
 - Local exchange of messages (reduced computational complexity).
- Challenge: the variational update is analytically intractable due to
 - the nonlinear function $h(x, s)$
 - complicated priori or other probability density function.
- Strategies:
 - Importance Sampling → Variational Message Passing (high accuracy)
 - Local linear approximation → Parameterized Update (low complexity)



The Mean-Field VBI Algorithm

2.4 Importance Sampling → Variational Message Passing [3]

- Given $\{\mathbf{x}_k, \omega_k | \forall k = 1 : N_S\} \sim q(\mathbf{x})$, the complicated term becomes

$$\langle \mathbf{h}(\mathbf{x}, \mathbf{s}) \rangle_{\mathbf{x}} \approx \sum_{k=1:N_S} \omega_k \mathbf{h}(\mathbf{x}_k, \mathbf{s}) \quad (24)$$

$$\hat{q}(\mathbf{s}) = p(\mathbf{s}) \mathcal{N}(\mathbf{z} | \langle \mathbf{h}(\mathbf{x}, \mathbf{s}) \rangle_{\mathbf{x}}, \mathbf{W}), \quad (25)$$

$$\approx p(\mathbf{s}) \prod_{k=1:N_S} \exp \left(-\frac{1}{2} \omega_k (\mathbf{z} - \mathbf{h}(\mathbf{x}_k, \mathbf{s}))^\top \mathbf{W} (\mathbf{z} - \mathbf{h}(\mathbf{x}_k, \mathbf{s})) \right), \quad (26)$$

- Generate $\{\mathbf{s}_i | \forall i = 1 : N_S\} \sim \mathcal{N}(\mathbf{s} | \boldsymbol{\mu}_n, \mathbf{U}_n)$, then $q(\mathbf{s})$ can be expressed as

$$q(\mathbf{s}) \approx \sum_{i=1:N_S} \wp_i \delta(\mathbf{s} - \mathbf{s}_i), \quad (27)$$

$$\wp_i = \frac{p(\mathbf{s}_i) \prod_{k=1:N_S} \exp \left(-\frac{1}{2} \omega_k (\mathbf{z} - \mathbf{h}(\mathbf{x}_k, \mathbf{s}_i))^\top \mathbf{W} (\mathbf{z} - \mathbf{h}(\mathbf{x}_k, \mathbf{s}_i)) \right)}{\mathcal{N}(\mathbf{s}_i | \boldsymbol{\mu}_n, \mathbf{U}_n)}. \quad (28)$$



The Mean-Field VBI Algorithm

2.4 Variational Message Passing [3]

- VMP-Associated Estimator
 - Approximate MAP

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}_k | \forall k=1:N_S} \omega(\mathbf{x}_k), \quad (29)$$

where the particle belief ω_k is viewed as a function of the particle support \mathbf{x}_k , which is equivalently denoted by $\omega(\mathbf{x}_k)$.

- Approximate MMSE

$$\hat{\mathbf{x}} = \sum_{k=1:N_S} \omega_k \mathbf{x}_k. \quad (30)$$

Stochastic particle-based variational Bayesian inference can significantly reduce the complexity and accelerate the convergence speed, see:

Z. Hu, A. Liu, Y. Wan, T. X. Han, and M. Zhao, “A two-stage multiband radar sensing scheme via stochastic particle-based variational Bayesian inference,” arXiv preprint arXiv:2207.10427, 2023.



The Mean-Field VBI Algorithm

Algorithm 2: The mean-field VMP algorithm

Input : $\{z, \mu, U\}$.

- 1 Collect the measurement signal z .
 - 2 Initialize x_n , χ_n , s_n and U_n .
 - 3 **While** not converge **do**
 - 4 Draw $\{x_k | \forall k = 1 : N_S\} \sim \mathcal{N}(x | x_n, \chi_n)$.
 - 5 Update ω_k , $\forall k = 1 : N_S$.
 - 6 Estimate $\hat{x} = \sum_{k=1:N_S} \omega_k x_k$, and let $x_n = \hat{x}$.
 - 7 Calculate $\chi_n = \sum_{k=1:N_S} \omega_k (x_k - \hat{x})(x_k - \hat{x})^\top$.
 - 8 Generate $\{s_i | \forall i = 1 : N_S\} \sim \mathcal{N}(s | \mu_n, U_n)$.
 - 9 Update φ_i , $\forall i = 1 : N_S$, based on Eq. (28).
 - 10 Estimate $\hat{s} = \sum_{i=1:N_S} \varphi_i s_i$, and let $s_n = \hat{s}$.
 - 11 Calculate $U_n = \sum_{i=1:N_S} \varphi_i (s_i - \hat{s})(s_i - \hat{s})^\top$.
 - 12 **End**
 - 13 Return the estimation \hat{x} and \hat{s} .
-

Output: \hat{x} and \hat{s} .



The Mean-Field VBI Algorithm

2.5 Local Linear Approximation → Parameterized Update

■ Local approximation

$$h(x, s) = h(x_n, s_n) + \underbrace{\nabla_{x_n} h^\top(x, s_n)(x - x_n)}_{H(x_n; s_n)} + \underbrace{\nabla_{s_n} h^\top(x_n, s)(s - s_n)}_{G(s_n; x_n)}, \quad (31)$$

$$\langle h(x, s) \rangle_x = \underbrace{h(x_n, s_n) + H(x_n; s_n)(\langle x \rangle - x_n) - G(s_n; x_n)s_n}_{g(x_n, s_n)} + G(s_n; x_n)s, \quad (32)$$

$$q(s) \propto p(s)\mathcal{N}(z|\langle h(x, s) \rangle_x, W) \quad (33)$$

$$= \mathcal{N}(s|\mu, U)\mathcal{N}(z' | G(s_n; x_n)s, W)|_{z'=z-g(x_n, s_n)} \quad (34)$$

$$= \mathcal{N}(s|\mu^\#, U^\#), \quad (35)$$

$$\mu^\# = (U^\#)^{-1}((G(s_n; x_n))^\top W(z - g(x_n, s_n)) + U\mu), \quad (36)$$

$$U^\# = (G(s_n; x_n))^\top W G(s_n; x_n) + U. \quad (37)$$



The Mean-Field VBI Algorithm

Algorithm 3: The parameterized VBI algorithm

Input : $\{z, \mu, U\}$.

- 1 Collect the measurement signal z .
- 2 Initialize variational states x_n and s_n , $n = 0$.
- 3 Let $\langle x \rangle = x_0$ and $\langle \mu \rangle = s_0$.
- 4 **While** not converge **do**
- 5 Update $\chi^\# = (\mathbf{G}'(x_n; s_n))^\top W \mathbf{G}'(x_n; s_n)$.
- 6 Update $x^\# = (\chi^\#)^{-1} (\mathbf{G}'(x_n; s_n))^\top W (z - g'(x_n, s_n))$.
- 7 Let $\langle x \rangle = x^\#$.
- 8 Update $U^\# = (\mathbf{G}(x_n, s_n))^\top W \mathbf{G}(x_n, s_n) + U$.
- 9 Update $\mu^\# = (U^\#)^{-1} ((\mathbf{G}(x_n, s_n))^\top W (z - g(x_n, s_n)) + U \mu)$.
- 10 Let $\langle \mu \rangle = \mu^\#$.
- 11 Let $x_n = x^\#$, and $s_n = s^\#$.
- 12 **End**
- 13 Return the estimation $\hat{x}_{\text{MMSE}} = x^\#$
- 14 Return the estimation $\hat{s}_{\text{MMSE}} = s^\#$.

Output: \hat{x} and \hat{s} .



Sparse Bayesian Learning

- [1] M. E. Tipping, “Sparse Bayesian learning and the relevance vector machine,” Journal of Machine Learning Research, vol. 1, no. Jun, pp. 211–244, 2001.
- [2] S. Ji, Y. Xue, and L. Carin, “Bayesian compressive sensing,” IEEE Trans. Signal Process., vol. 56, no. 6, pp. 2346–2356, 2008.



Problem Formulation

$$\begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix}_{N \times 1} = \begin{bmatrix} \phi_{11} & \phi_{12} & \cdots & \phi_{1M} \\ \vdots & \vdots & \cdots & \vdots \\ \phi_{N1} & \phi_{N2} & \cdots & \phi_{NM} \end{bmatrix}_{N \times M} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix}_{M \times 1} + \begin{bmatrix} n_1 \\ n_2 \\ \vdots \\ n_N \end{bmatrix}_{N \times 1}$$

$$\mathbf{t} = \Phi \mathbf{w} + \mathbf{n}$$

\mathbf{t}, Φ is known

\mathbf{w} is sparse $\xrightarrow{\text{controlled by}}$ $p(w_i | \delta_i) = CN(w_i | 0, \delta_i)$

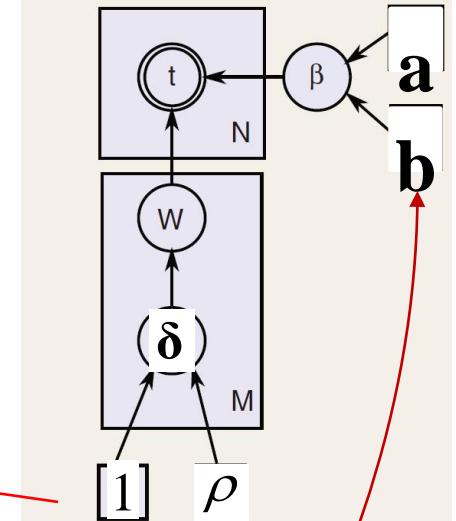
\mathbf{n} is Gaussian

SBL formulation

$$\mathbf{t} = \Phi \mathbf{w} + \mathbf{n}$$

$$p(\mathbf{w}|\boldsymbol{\delta}) = CN\left(\mathbf{w}|\mathbf{0}, diag(\boldsymbol{\delta})\right)$$

$$p(\boldsymbol{\delta}) = \prod_i \Gamma(\delta_i; 1, \rho), \quad \rho = 0.01$$



$$p(\mathbf{n}) = CN\left(\mathbf{n}|\mathbf{0}, \sigma^2 \mathbf{I}\right) = CN\left(\mathbf{n}|\mathbf{0}, \beta^{-1} \mathbf{I}\right)$$

$$p(\mathbf{t}|\mathbf{w}) = CN\left(\mathbf{t}|\Phi \mathbf{w}, \beta^{-1} \mathbf{I}\right)$$

$$p(\beta) = \Gamma(\beta; a, b), \quad a = b = 0.001$$

Bayesian Inference

$$\mathbf{t} = \Phi \mathbf{w} + \mathbf{n}$$



$p(\mathbf{w}, \boldsymbol{\delta}, \beta | \mathbf{t})$ cannot be explicitly calculated

treat \mathbf{w} as a hidden variable

$p(\boldsymbol{\delta}, \beta | \mathbf{t})$ —— evidence function

equivalently

$p(\boldsymbol{\delta}, \beta, \mathbf{t})$ $\xrightarrow{\max_{\boldsymbol{\delta}, \beta}} \ln p(\boldsymbol{\delta}, \beta, \mathbf{t})$



$$\ln p(\delta, \beta, t) = \ln \int p(w, \delta, \beta, t) d\mathbf{w}$$

$$= \ln \int q(w) \frac{p(w, \delta, \beta, t)}{q(w)} d\mathbf{w}$$

Jensen's inequality

hold with equality

$$q(w) = p(w | t, \delta, \beta) \geq \int q(w) \ln \frac{p(w, \delta, \beta, t)}{q(w)} d\mathbf{w}$$

$$\ln p(\delta^{(t)}, \beta^{(t)}, t) = \int q^{(t)}(w) \ln \frac{p(w, \delta^{(t)}, \beta^{(t)}, t)}{q^{(t)}(w)} d\mathbf{w}, \quad \text{with } q^{(t)}(w) \triangleq p(w | t, \delta^{(t)}, \beta^{(t)})$$

$$\{\delta^{(t+1)}, \beta^{(t+1)}\} = \max_{\delta, \beta} \int q^{(t)}(w) \ln \frac{p(w, \delta, \beta, t)}{q^{(t)}(w)} d\mathbf{w}$$

$$\ln p(\delta^{(t+1)}, \beta^{(t+1)}, t) \geq \int q^{(t)}(w) \ln \frac{p(w, \delta^{(t+1)}, \beta^{(t+1)}, t)}{q^{(t)}(w)} d\mathbf{w}$$

$$\geq \int q^{(t)}(w) \ln \frac{p(w, \delta^{(t)}, \beta^{(t)}, t)}{q^{(t)}(w)} d\mathbf{w} = \ln p(\delta^{(t)}, \beta^{(t)}, t)$$



E-step and M-step

$$\{\boldsymbol{\delta}^{(t+1)}, \beta^{(t+1)}\} = \max_{\boldsymbol{\delta}, \beta} \int q^{(t)}(\mathbf{w}) \ln \frac{p(\mathbf{w}, \boldsymbol{\delta}, \beta, \mathbf{t})}{q^{(t)}(\mathbf{w})} d\mathbf{w}$$



$$\max_{\boldsymbol{\delta}, \beta} E \left\langle \ln p(\mathbf{w}, \boldsymbol{\delta}, \beta, \mathbf{t}) \right\rangle_{p(\mathbf{w} | \mathbf{t}, \beta^{(t)}, \boldsymbol{\delta}^{(t)})}$$

E-step:

$$\text{where } \ln p(\mathbf{w}, \boldsymbol{\delta}, \beta, \mathbf{t}) = \ln p(\mathbf{t} | \mathbf{w}, \beta) p(\mathbf{w} | \boldsymbol{\delta}) p(\boldsymbol{\delta}) p(\beta)$$

M-step: Update \delta and \beta



Multiple Measurements with Group Sparsity

$$[\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_L] = \Phi \underbrace{[\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_L]}_{\mathbf{W}} + [\mathbf{n}_1, \mathbf{n}_2, \dots, \mathbf{n}_L]$$

\mathbf{W} is row-sparse $\xrightarrow{\text{controlled by}}$ $p(\mathbf{w}_i | \boldsymbol{\delta}) = CN(\mathbf{w}_i | \mathbf{0}, \boldsymbol{\delta})$

$$\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1L} \\ \hline w_{n1} & w_{n2} & \cdots & w_{nL} \end{bmatrix} \quad \begin{array}{l} \xrightarrow{\hspace{1cm}} \delta_1 \\ \xrightarrow{\hspace{1cm}} \delta_n \end{array}$$



Turbo CS

J. Ma, X. Yuan, and L. Ping, “On the Performance of Turbo Signal Recovery with Partial DFT Sensing Matrices,” IEEE Signal Processing Letters, vol. 22, no. 10, pp. 1580–1584, Oct 2015.

$$y = \underbrace{\begin{matrix} A \\ \vdots \end{matrix}}_N x + n$$

The diagram illustrates the linear system $y = Ax + n$. On the left, a vertical vector y is shown. An equals sign follows. To the right of the equals sign is a large rectangular matrix A , which is circled in red. Below A is a bracket labeled N , indicating it is an $M \times N$ matrix. To the right of the matrix is a plus sign. Following the plus sign are two vertical vectors: x and n .

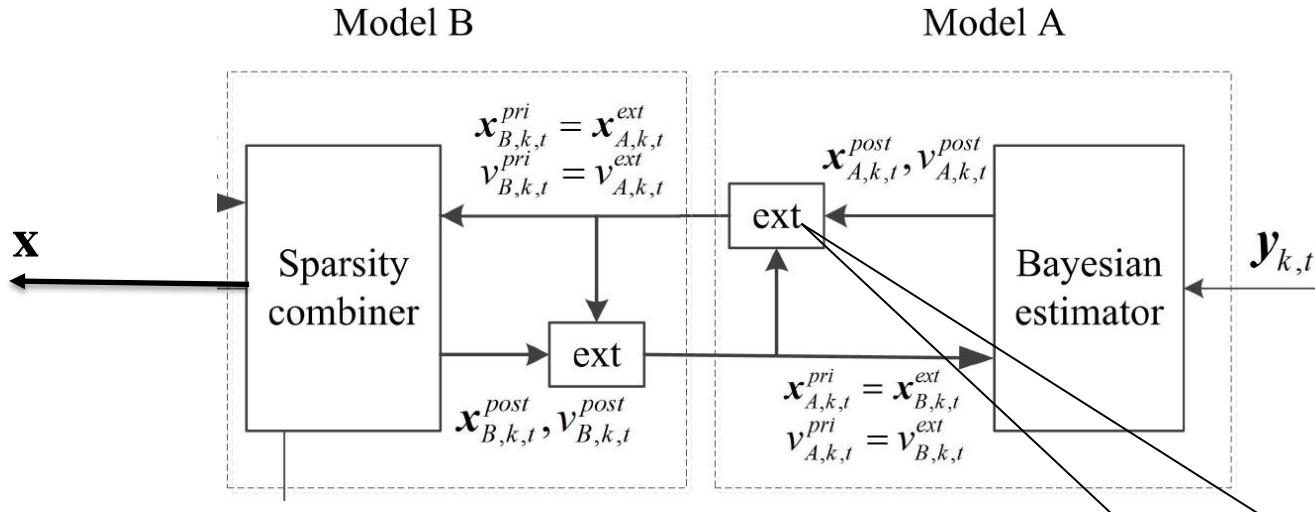
- In many applications, **A is structured rather than iid random.**

- For example, **A** consists of random rows of the DFT matrix in image processing, such as magnetic resonance imaging (MRI).
- AMP doesn't work well when **A** is a partial DFT matrix.
- For structured sensing matrices, how to design a linear-complexity compressed sensing algorithm with near-optimal performance?

$$\begin{matrix} \mathbf{y} \\ M \end{matrix} = \boxed{\mathbf{A}}_{N \times M} \begin{matrix} \mathbf{x} \\ \mathbf{n} \end{matrix} +$$

The diagram illustrates the linear system $\mathbf{y} = \mathbf{Ax} + \mathbf{n}$. On the left, a vertical vector \mathbf{y} is shown with a brace indicating its length M . An equals sign follows. To the right is a large rectangular box labeled \mathbf{A} with a brace at the bottom indicating its width N . To the right of the box is a plus sign. Following the plus sign are two vertical vectors: \mathbf{x} and \mathbf{n} , both with braces indicating their height M .

- Our goal is to estimate \mathbf{x} with partial orthogonal matrix $\mathbf{A} = \mathbf{F}_{partial}$.
- Stakes at hand:
 - The measurement vector \mathbf{y}
 - \mathbf{x} is a sparse signal



✓ Module A: $\mathbf{y} = \mathbf{F}_{\text{partial}} \mathbf{x} + \mathbf{n}$

Assumes that \mathbf{x} follows a Gaussian distribution $p(\mathbf{x}; \mathbf{x}_A^{\text{pri}}, v_A^{\text{pri}}) = \text{CN}(\mathbf{x}; \mathbf{x}_A^{\text{pri}}, v_A^{\text{pri}} \mathbf{I})$, calculate the posterior distribution of \mathbf{x}

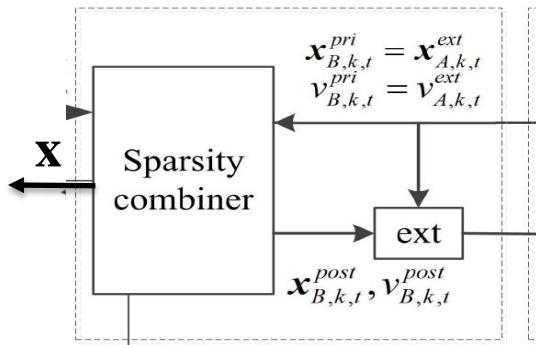
$$p(\mathbf{x} | \mathbf{y}; \mathbf{x}_A^{\text{pri}}, v_A^{\text{pri}}, \sigma_n) = \frac{p(\mathbf{y} | \mathbf{x}; \sigma_n) p(\mathbf{x}; \mathbf{x}_A^{\text{pri}}, v_A^{\text{pri}})}{p(\mathbf{y}; \mathbf{x}_A^{\text{pri}}, v_A^{\text{pri}}, \sigma_n)}$$

The extrinsic messages are constants in iteration

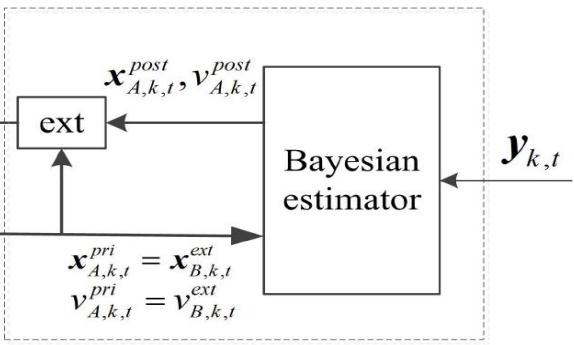
✓ Module B: \mathbf{x} is sparse

In practical application, different elements of \mathbf{x} can be characterized by different variances,

Model B



Model A



✓ Module A:

$$p(\mathbf{x}_k | \mathbf{y}_k; \mathbf{x}_A^{pri}, v_A^{pri}, \sigma_n) = \frac{p(\mathbf{y} | \mathbf{x}; \sigma_n) p(\mathbf{x}; \mathbf{x}_A^{pri}, v_A^{pri})}{p(\mathbf{y}; \mathbf{x}_A^{pri}, v_A^{pri}, \sigma_n)}$$

where

$$p(\mathbf{y} | \mathbf{x}; \sigma_n) = \text{CN} (\mathbf{y}; \mathbf{Ax}, \sigma_n^2 \mathbf{I})$$

$$p(\mathbf{y}; \mathbf{x}_A^{pri}, v_A^{pri}, \sigma_n)$$

$$= \int p(\mathbf{y} | \mathbf{x}; \sigma_n) p(\mathbf{x}; \mathbf{x}_A^{pri}, v_A^{pri}) d\mathbf{x}$$

$$= \text{CN} \left(\mathbf{y}; A\mathbf{x}_A^{pri}, \sigma_n^2 \mathbf{I} + \frac{1}{v_A^{pri}} AA^H \right)$$

$$\begin{aligned} & p(\mathbf{x}_{k,t} | \mathbf{y}_{k,t}; \mathbf{x}_A^{pri}, v_A^{pri}, \sigma_n) \\ &= \text{CN} (\mathbf{x}_{k,t}; \mathbf{x}_A^{post}, \mathbf{V}_A^{post}) \end{aligned}$$

$$\mathbf{V}_A^{post} = \left(\frac{\mathbf{A}^H \mathbf{A}}{\sigma_n^2} + \frac{1}{v_A^{pri}} \mathbf{I} \right)^{-1}$$

$$\mathbf{x}_A^{post} = \mathbf{V}_A^{post} \left(\frac{\mathbf{x}_A^{pri}}{v_A^{pri}} + \frac{\mathbf{A}^H \mathbf{y}}{\sigma_n^2} \right).$$

The complexity is too high



- When A is an arbitrary matrix

$$\begin{aligned}\mathbf{V}_A^{\text{post}} &= \left(\frac{\mathbf{A}^H \mathbf{A}}{\sigma_n^2} + \frac{1}{v_A^{\text{pri}}} \mathbf{I} \right)^{-1} \\ &= v_A^{\text{pri}} \mathbf{I} - \frac{(v_A^{\text{pri}})^2}{\sigma_n^2} \mathbf{V} \mathbf{D}^H \left(\mathbf{I} + \frac{v_A^{\text{pri}}}{\sigma_n^2} \mathbf{D} \mathbf{D}^H \right)^{-1} \mathbf{D} \mathbf{V}^H\end{aligned}$$

Where $\mathbf{U} \mathbf{D} \mathbf{V}^H = \mathbf{A}$

- When A is a partial orthogonal matrix

$$\mathbf{D} = \mathbf{I} \quad \mathbf{V} \mathbf{V}^H \approx \frac{M}{N} \mathbf{I}$$

$$v_A^{\text{post}} = v_A^{\text{pri}} - \frac{M}{N} \cdot \frac{(v_A^{\text{pri}})^2}{v_A^{\text{pri}} + \sigma^2}$$

$$\mathbf{x}_A^{\text{post}} = \mathbf{x}_A^{\text{pri}} + \frac{v_A^{\text{pri}}}{v_A^{\text{pri}} + \sigma_n^2} \mathbf{A}^H (\mathbf{y} - \mathbf{A} \mathbf{x}_A^{\text{pri}})$$

✓ The extrinsic messages

$$v_B^{pri} = v_A^{ext} = \left(\frac{1}{v_A^{post}} - \frac{1}{v_A^{pri}} \right)^{-1}$$

$$\mathbf{h}_B^{pri} = \mathbf{h}_A^{ext} = v_B^{pri} \left(\frac{\mathbf{h}_A^{post}}{v_A^{post}} - \frac{\mathbf{h}_A^{pri}}{v_A^{pri}} \right)$$

When \mathbf{x} is characterized by different variances, the extrinsic variance can be approximated by

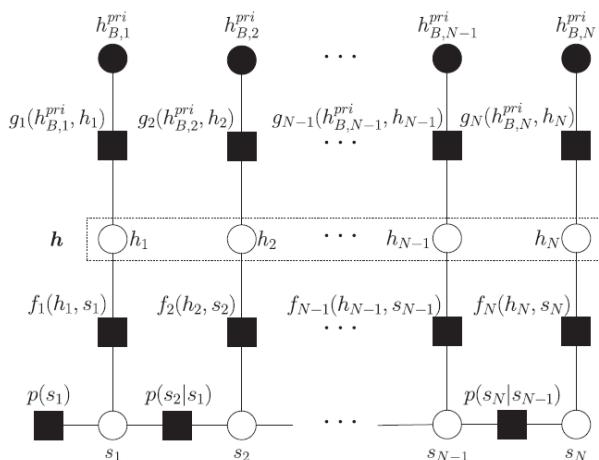
$$v_{B,k,t}^{pri} = \text{mean} \left\{ v_{A,k,t,i}^{ext} \right\}.$$

To start with, a basic assumption is to model \mathbf{h}_B^{pri} as an additive white Gaussian noise (AWGN) observation, i.e.,

$$\mathbf{h}_B^{pri} = \mathbf{h} + \mathbf{w},$$

where $\mathbf{w} \sim \mathcal{CN}(\mathbf{0}, v_B^{pri} \mathbf{I})$ is independent of \mathbf{h} .

✓ Module B: (Markov)



5: Update the messages $\nu_{h_n \rightarrow f_n}(h_n), \nu_{f_n \rightarrow s_n}(s_n)$ using (9) and (10).

$$6: \lambda_1^f = \lambda, \lambda_N^b = \frac{1}{2}$$

for $n = 2, \dots, N$

$$7: \lambda_n^f = \frac{p_{01}(1-\pi_{n-1}^{out})(1-\lambda_{n-1}^f) + p_{11}\pi_{n-1}^{out}\lambda_{n-1}^f}{(1-\pi_{n-1}^{out})(1-\lambda_{n-1}^f) + \pi_{n-1}^{out}\lambda_{n-1}^f}$$

end for

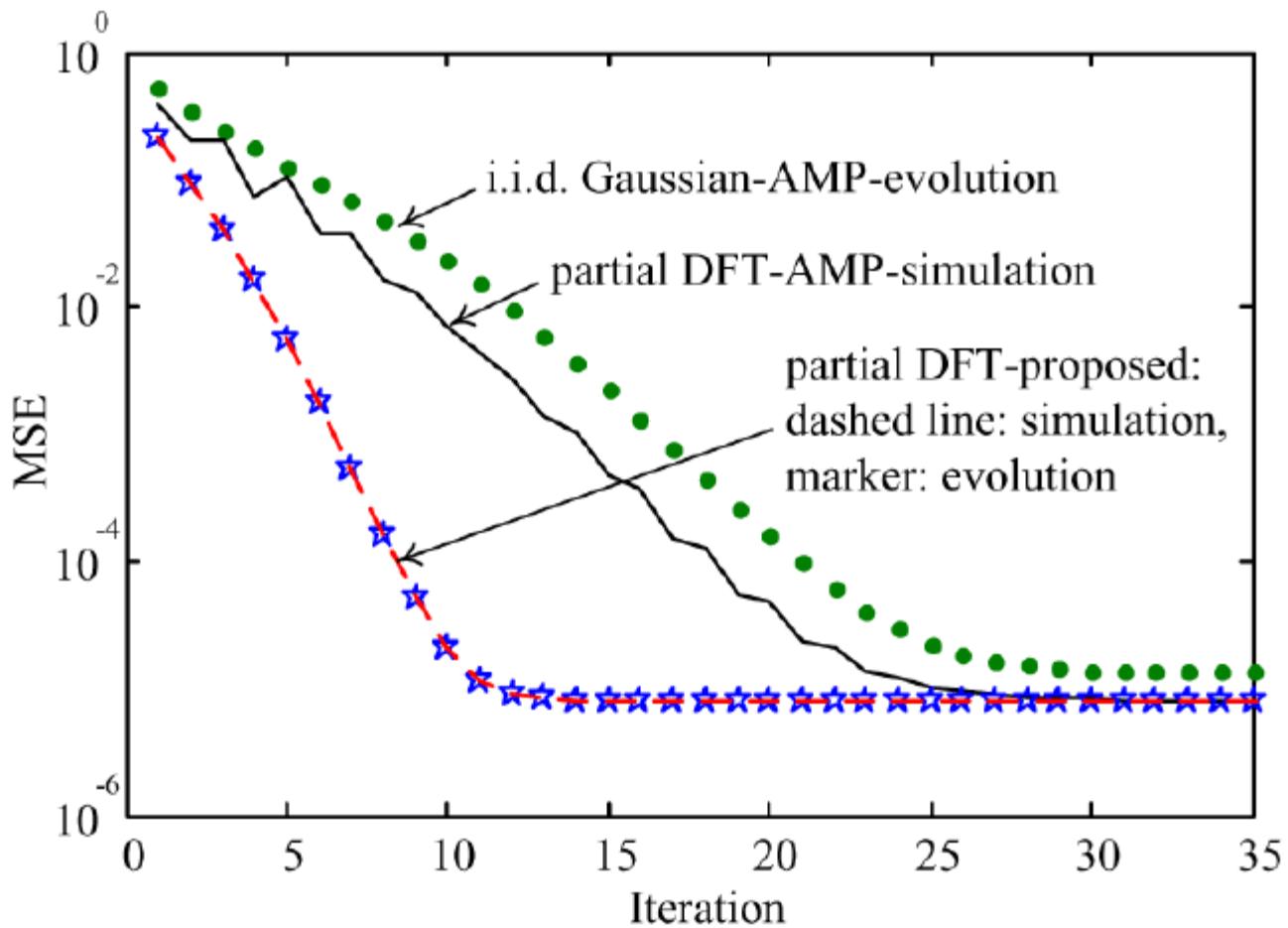
for $n = N-1, \dots, 2, 1$

$$8: \lambda_n^b = \frac{p_{10}(1-\pi_{n+1}^{out})(1-\lambda_{n+1}^b) + (1-p_{10})\pi_{n+1}^{out}\lambda_{n+1}^b}{(p_{00}+p_{10})(1-\pi_{n+1}^{out})(1-\lambda_{n+1}^b) + (p_{11}+p_{01})\pi_{n+1}^{out}\lambda_{n+1}^b}$$

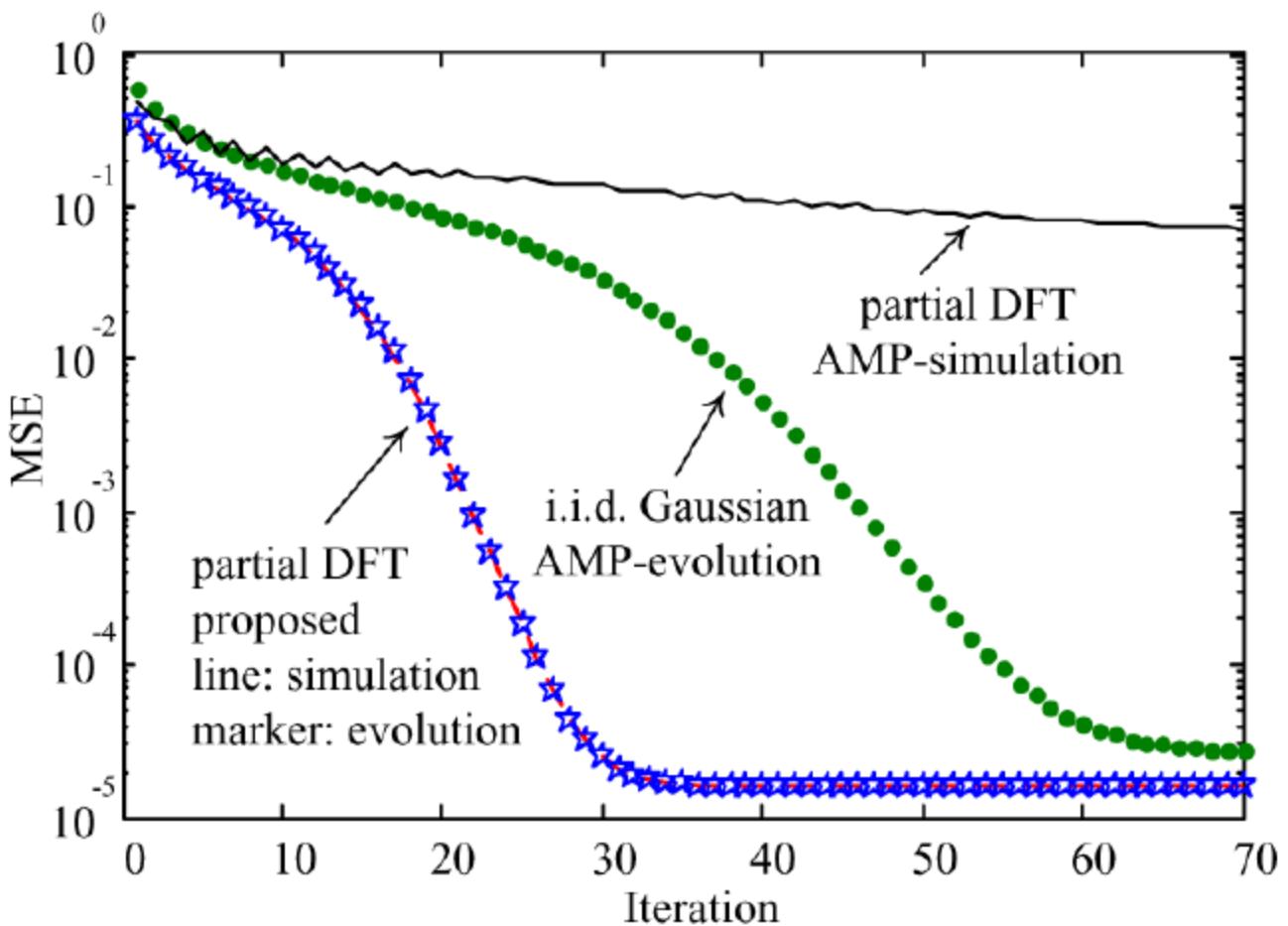
end for

9: Update the messages $\nu_{s_n \rightarrow f_n}(s_n), \nu_{f_n \rightarrow h_n}(h_n)$ using (12)

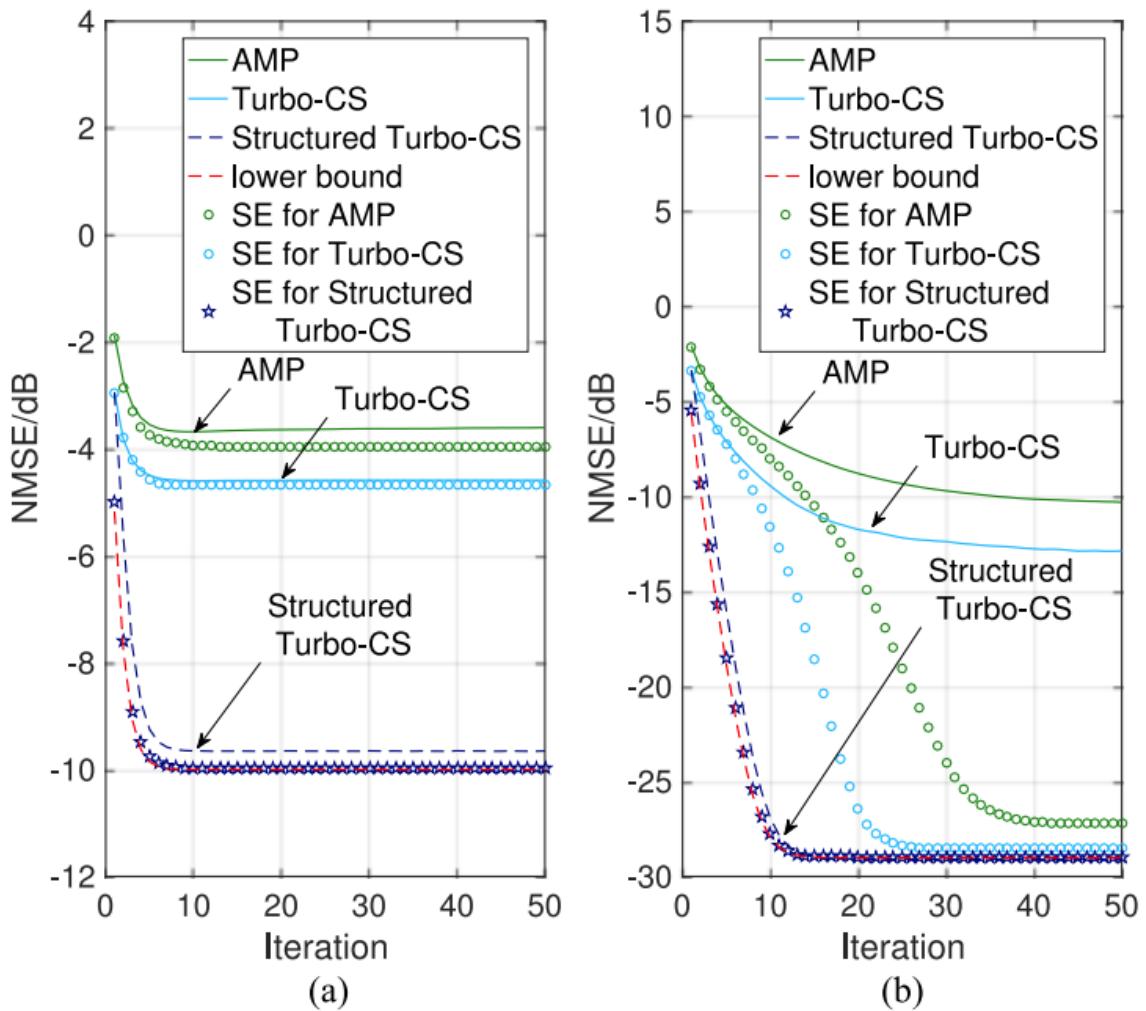




Comparisons of the proposed algorithm and AMP. $N = 8192$, $M = 5734(\approx 0.7N)$, $\lambda = 0.4$, and SNR = 50 dB.



Comparisons of the proposed algorithm and AMP. $N = 32768$, $M = 18022(\approx 0.55N)$, $\lambda = 0.4$, and SNR = 50 dB.



Comparison of state evolution and simulation results under $\text{SNR} = 10 \text{ dB}$ in (a) and 30 dB in (b). Set $N = 128$, $M = 51$, $p_{01} = 1/96$ and $p_{10} = 1/32$ (i.e., $\lambda = 0.25$) for simulation. i.i.d. complex Gaussian matrix is used for simulation of the AMP algorithm.



Turbo VBI

A. Liu, G. Liu, L. Lian, V. K. N. Lau and M. Zhao, “Robust Recovery of Structured Sparse Signals With Uncertain Sensing Matrix: A Turbo-VBI Approach,” in IEEE Transactions on Wireless Communications, vol. 19, no. 5, pp. 3185-3198, May 2020.



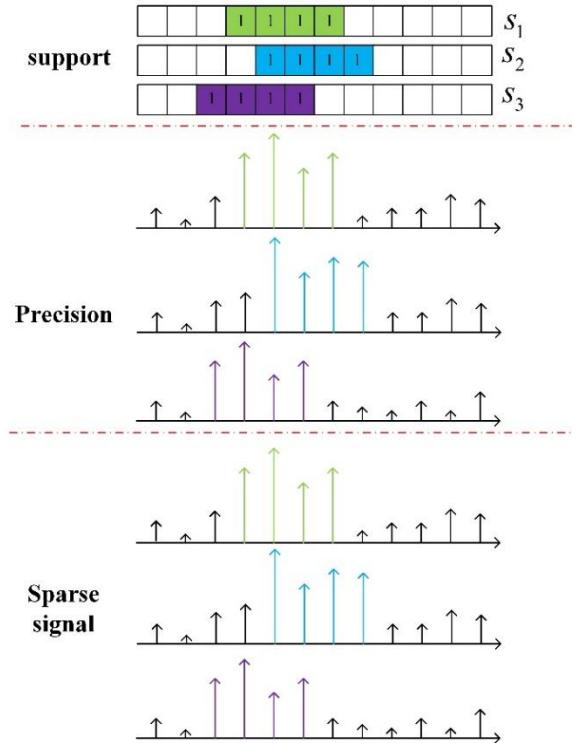
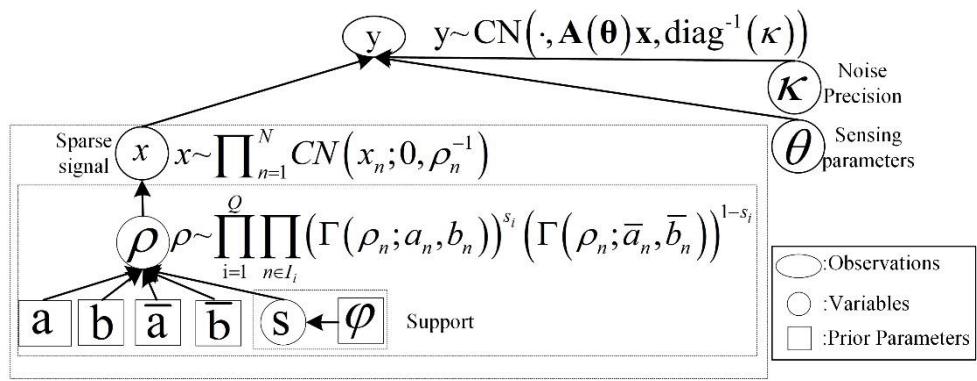
Issues of Existing CS Recovery Algorithms

	Advantages	Disadvantage
AMP-based algorithms	flexible to cover a wide range of structured sparsities	only work well for certain types of sensing matrix
SBL/ VBI	insensitive to the sensing matrix	not flexible enough to model more complicated sparse structures

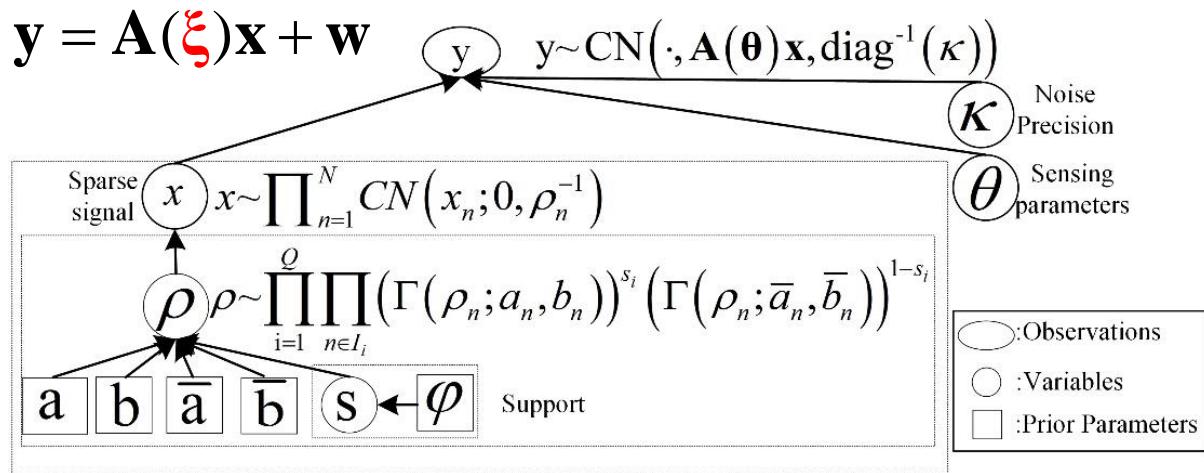
System Model

$$\mathbf{y} = \mathbf{A}(\xi)\mathbf{x} + \mathbf{w}$$

$$p(\mathbf{x}, \boldsymbol{\rho}, \mathbf{s} | \phi) = \underbrace{p(\mathbf{s} | \phi)}_{\text{Support}} \underbrace{p(\boldsymbol{\rho} | \mathbf{s})}_{\text{Precision}} \underbrace{p(\mathbf{x} | \boldsymbol{\rho})}_{\text{Sparse signal}}$$



EM-based Problem Formulation



- ✓ Computing the conditional marginal posteriors

$$p(s_i | \mathbf{y}, \xi) \propto \sum \iint p(\mathbf{x}, \rho, s | \phi) p(\mathbf{y} | \mathbf{x}, \xi) d\rho d\mathbf{x}$$

$$p(\mathbf{x} | \mathbf{y}, \xi) \propto \sum_s \int p(\mathbf{y}, \mathbf{x}, \rho, s | \xi) d\rho = \sum_s \int p(\mathbf{x}, \rho, s | \phi) p(\mathbf{y} | \mathbf{x}, \xi) d\rho$$

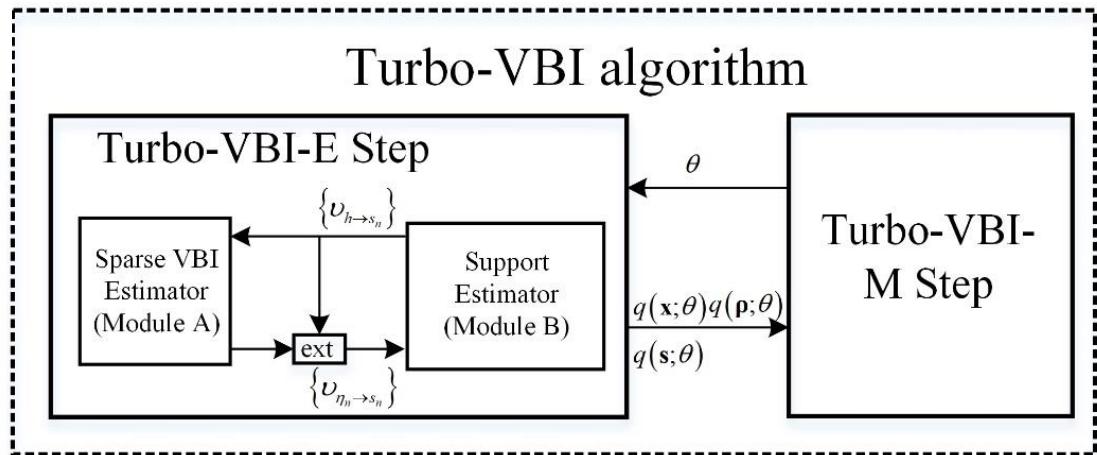
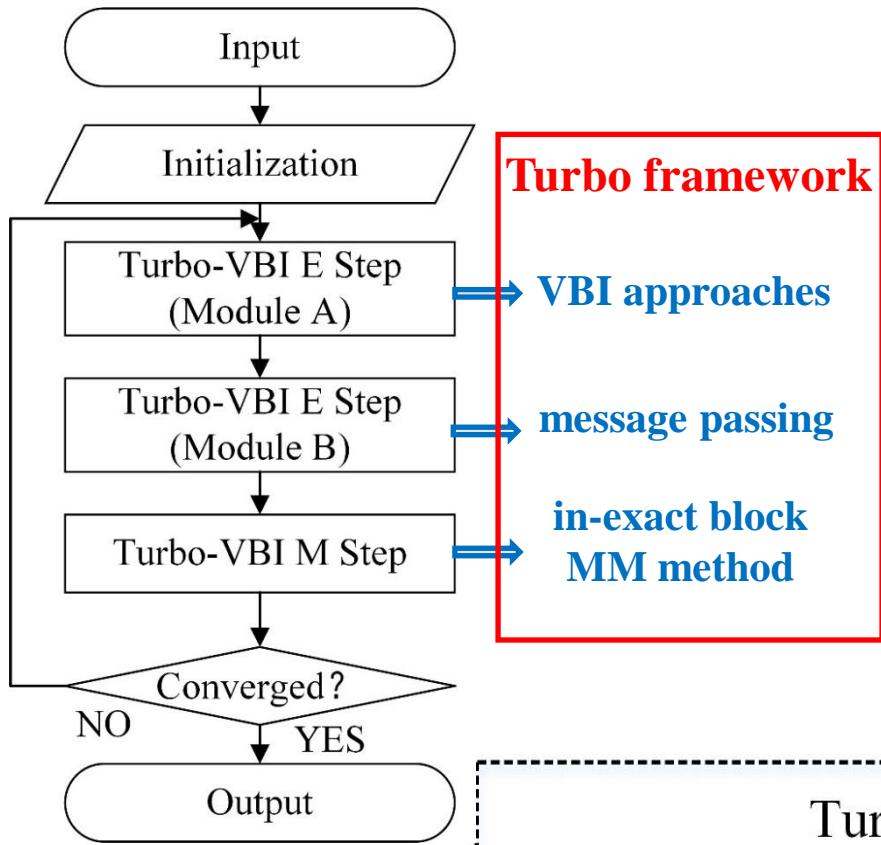
$v = \{x, \rho, s\}$ is the collection of variables.

- ✓ MAP estimation of the uncertain parameters ξ

$$\xi^* = \underset{\xi}{\operatorname{argmax}} \ln p(\xi | \mathbf{y}) = \underset{\xi}{\operatorname{argmax}} \ln \sum_s \iint p(\mathbf{y}, v, \xi) d\rho d\mathbf{x}$$

Turbo-VBL
E Step

Turbo-VBI
M Step





Turbo-VBI-M Step

$$\boldsymbol{\xi}^* = \operatorname{argmax}_{\boldsymbol{\xi}} \ln p(\boldsymbol{\xi} | \mathbf{y}) = \operatorname{argmax}_{\boldsymbol{\xi}} \ln \sum_s \iint p(\mathbf{y}, \mathbf{v}, \boldsymbol{\xi}) d\mathbf{p} d\mathbf{x}$$

- ✓ Alternatively maximize a **surrogate function** of $\ln p(\boldsymbol{\xi}, \mathbf{y})$ with respect to each ξ_j

EM-based Surrogate Function

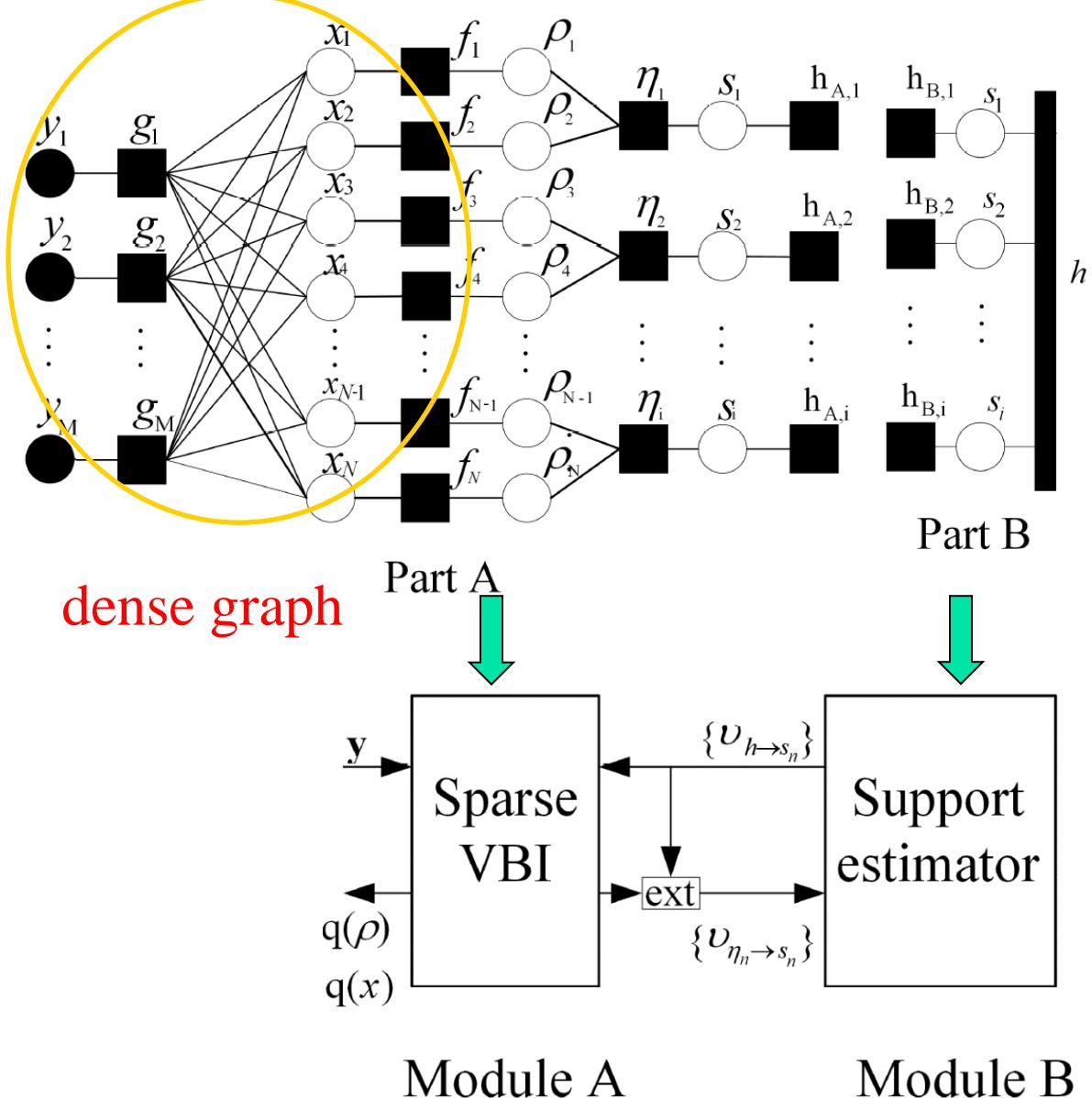
$$u(\boldsymbol{\xi}; \dot{\boldsymbol{\xi}}) = u^{\text{EM}}(\boldsymbol{\xi}; \dot{\boldsymbol{\xi}}) + \sum_{j \in \mathcal{I}_c^1} \tau_j \left\| \boldsymbol{\xi}_j - \dot{\boldsymbol{\xi}}_j \right\|^2$$

$$u^{\text{EM}}(\boldsymbol{\xi}; \dot{\boldsymbol{\xi}}) = \int p(v | y, \dot{\boldsymbol{\xi}}) \ln \frac{p(v, y, \boldsymbol{\xi})}{p(v | y, \dot{\boldsymbol{\xi}})} dv$$

Intractable !

- ✓ Find an alternative probability density function $q(v; \dot{\boldsymbol{\xi}})$ in the Turbo-VBI-E Step

Turbo-VBI-E Step



Module A

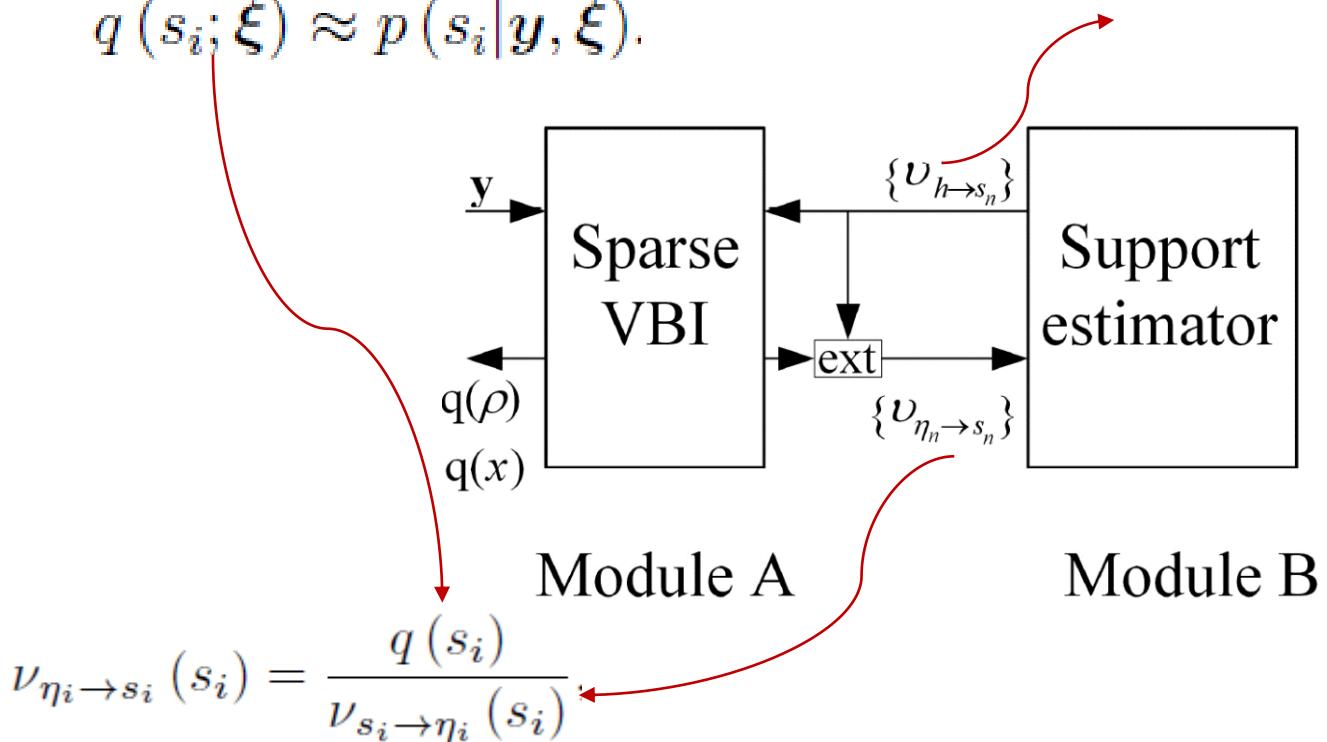
Module A performs the sparse VBI

$$q(x|\xi) \approx p(x|y, \xi)$$

$$q(\rho; \xi) \approx p(\rho|y, \xi);$$

$$q(s_i; \xi) \approx p(s_i|y, \xi).$$

$$h_{A,i}(s_i) \triangleq \nu_{h \rightarrow s_i}(s_i), i = 1, \dots, Q$$





Module A performs the sparse VBI

- ✓ minimizing the Kullback-Leibler divergence

$$\mathcal{A}_{\text{VBI}} : q^*(\boldsymbol{v}; \boldsymbol{\xi}) = \arg \min_{q(\boldsymbol{v}; \boldsymbol{\xi})} \int q(\boldsymbol{v}; \boldsymbol{\xi}) \ln \frac{q(\boldsymbol{v}; \boldsymbol{\xi})}{\hat{p}(\boldsymbol{v}|\boldsymbol{y}, \boldsymbol{\xi})} d\boldsymbol{v}$$

$$\text{s.t. } q(\boldsymbol{v}; \boldsymbol{\xi}) = \prod_{k \in \mathcal{H}} q(\boldsymbol{v}^k; \boldsymbol{\xi}), \int q(\boldsymbol{v}^k; \boldsymbol{\xi}) d\boldsymbol{v}^k = 1,$$

- ✓ Stationary Solution

$$q^*(\boldsymbol{v}^k) = \arg \min_{q(\boldsymbol{v}^k)} \int \prod_{l \neq k} q^*(\boldsymbol{v}^l) q(\boldsymbol{v}^k) \ln \frac{\prod_{l \neq k} q^*(\boldsymbol{v}^l) q(\boldsymbol{v}^k)}{\hat{p}(\boldsymbol{v}|\boldsymbol{y}, \boldsymbol{\xi})}$$

$$q(\boldsymbol{v}^k) \propto \exp \left(\langle \ln p(\boldsymbol{v}, \boldsymbol{y} | \boldsymbol{\xi}) \rangle_{\prod_{l \neq k} q(\boldsymbol{v}^l)} \right)$$

- ✓ The alternating optimization (AO) algorithm

$$q(\boldsymbol{x}) = \mathcal{CN}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

$$\boldsymbol{\Sigma} = \left(\text{diag} \left(\left\langle \frac{\tilde{a}_1}{\tilde{b}_1}, \dots, \frac{\tilde{a}_N}{\tilde{b}_N} \right\rangle \right) + \mathbf{A}(\boldsymbol{\theta})^H \text{diag}(\boldsymbol{\kappa}) \mathbf{A}(\boldsymbol{\theta}) \right)^{-1}$$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma} \mathbf{A}(\boldsymbol{\theta})^H \text{diag}(\boldsymbol{\kappa}) \mathbf{y}.$$



$$q(\rho) = \prod_{n=1}^N \Gamma\left(\rho_n; \tilde{a}_n, \tilde{b}_n\right)$$

$$\tilde{a}_n = \langle s_i \rangle a_n + \langle 1 - s_i \rangle \bar{a}_n + 1 = \tilde{\pi}_i a_n + (1 - \tilde{\pi}_i) \bar{a}_n + 1,$$

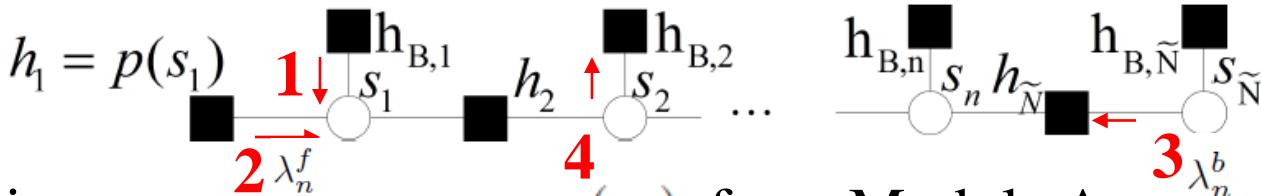
$$\tilde{b}_n = \langle |x_n|^2 \rangle + \langle s_i \rangle b_n + \langle 1 - s_i \rangle \bar{b}_n = |\mu_n|^2 +$$

$$\Sigma_n + \tilde{\pi}_i b_n + (1 - \tilde{\pi}_i) \bar{b}_n, , \forall n \in \mathcal{I}_i, \forall i.$$

$$q(s) = \prod_{i=1}^Q (\tilde{\pi}_i)^{s_i} (1 - \tilde{\pi}_i)^{1-s_i},$$

$$\tilde{\pi}_i = \frac{1}{C} \prod_{n \in \mathcal{I}_i} \frac{\pi_i b_n^{a_n}}{\Gamma(a_n)} e^{(a_n - 1)\langle \ln \rho_n \rangle - b_n \langle \rho_n \rangle},$$

Module B: Calculate $h_{A,i}(s_i) \triangleq \nu_{h \rightarrow s_i}(s_i)$, $i = 1, \dots, Q$



1. given messages $\nu_{\eta_i \rightarrow s_i}(s_i)$ from Module A

$$1 \quad h_{B,n}(s_n) \propto (\pi_n^{in})^{s_n} (1 - \pi_n^{in})^{1-s_n}, \forall n$$

$$\pi_n^{in} = \frac{\nu_{\eta_n \rightarrow s_n}(1)}{\nu_{\eta_n \rightarrow s_n}(1) + \nu_{\eta_n \rightarrow s_n}(0)}.$$

2. use the forward-backward message passing algorithm to update the messages

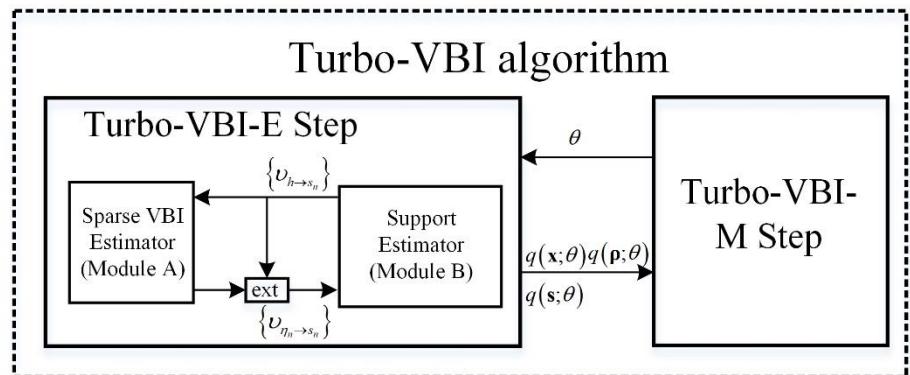
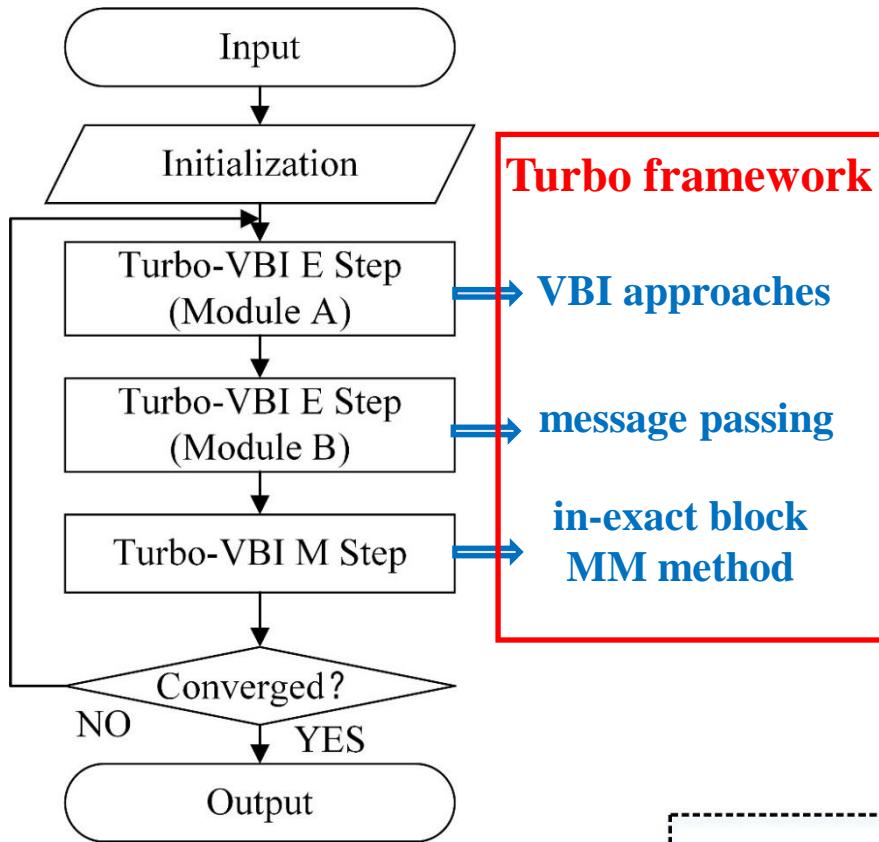
$$2 \quad \lambda_n^f = \frac{p_{01} (1 - \pi_{n-1}^{in}) (1 - \lambda_{n-1}^f) + p_{11} \pi_{n-1}^{in} \lambda_{n-1}^f}{(1 - \pi_{n-1}^{in}) (1 - \lambda_{n-1}^f) + \pi_{n-1}^{in} \lambda_{n-1}^f}$$

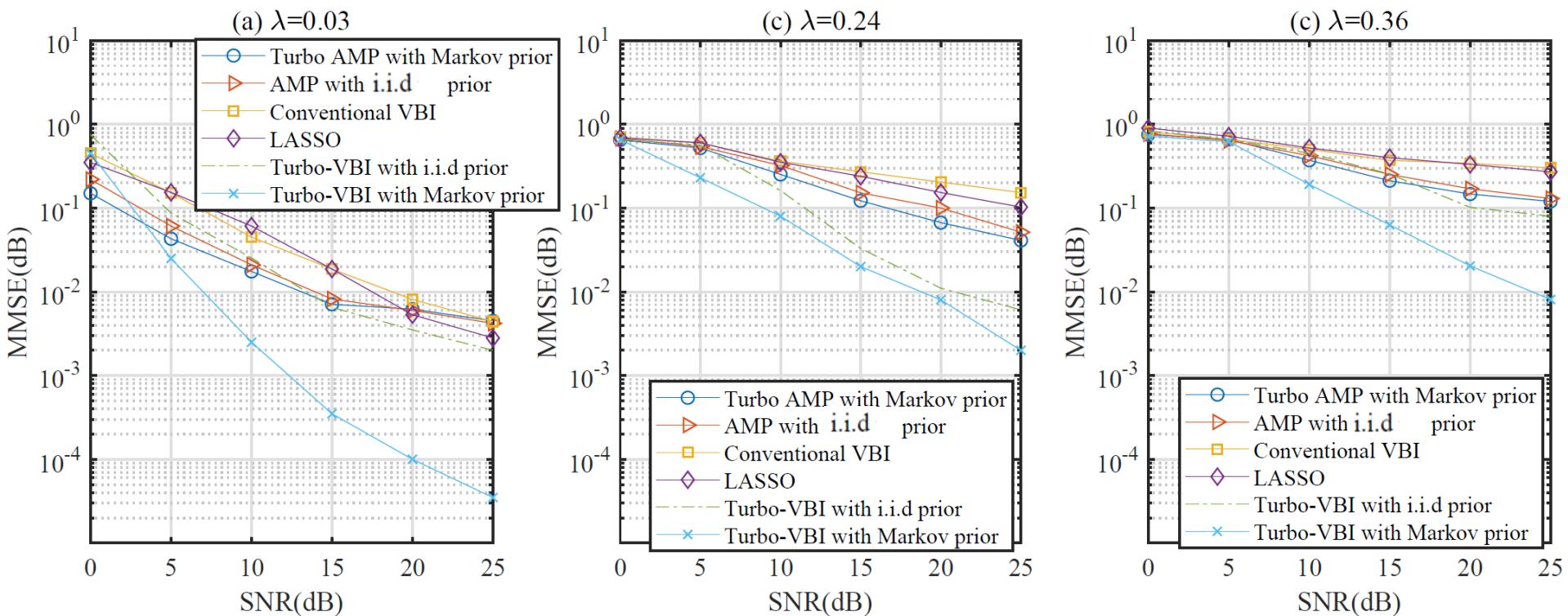
$$3 \quad \lambda_n^b = \frac{p_{10} (1 - \pi_{n+1}^{in}) (1 - \lambda_{n+1}^b) + p_{11} \pi_{n+1}^{in} \lambda_{n+1}^b}{p_0 (1 - \pi_{n+1}^{in}) (1 - \lambda_{n+1}^b) + p_1 \pi_{n+1}^{in} \lambda_{n+1}^b}$$

$$4 \quad \pi_n = \frac{\lambda_n^f \lambda_n^b}{(1 - \lambda_n^f) (1 - \lambda_n^b) + \lambda_n^f \lambda_n^b}.$$

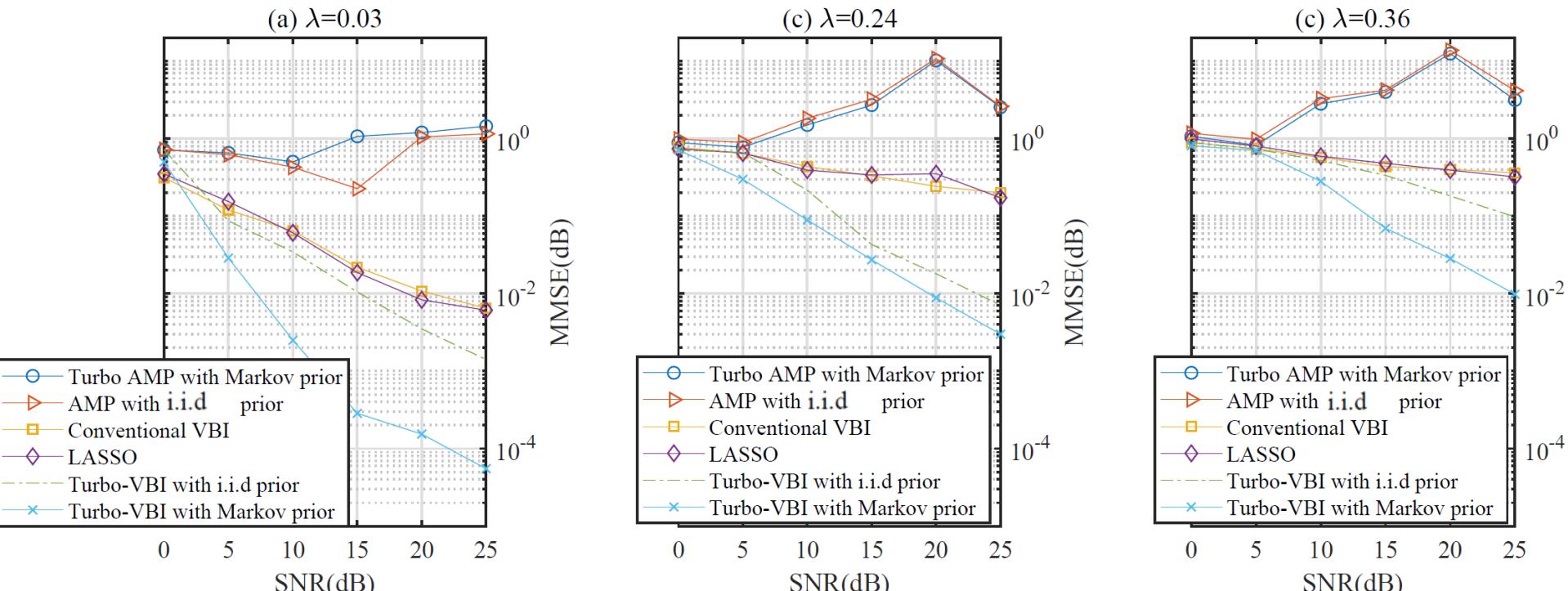
$$\nu_{h_n \rightarrow s_n}(s_n) = (\pi_n)^{s_n} (1 - \pi_n)^{1-s_n}$$

The subscript should be exchanged.





NMSE of the channel estimate versus the SNR for ULA.



NMSE of the channel estimate versus the SNR for 2D planar array.



Turbo Inverse-Free VBI

See “lecture note appendix 1”



Dynamic Bayesian Learning

Lixiang Lian, An Liu(#), and Vincent K. N. Lau, 'Exploiting Dynamic Sparsity for Downlink FDD-Massive MIMO Channel Tracking', IEEE Transactions on Signal Processing, vol. 67, no. 8, pp. 2007-2021, 15 April 15, 2019.



System model

$$y^{(t)} = A^{(t)} x^{(t)} + e^{(t)}, \quad t = 1, \dots, T,$$

The goal is to recover the time series $\{x^{(1)}, \dots, x^{(T)}\}$

Prior

$$x_n^{(t)} = s_n^{(t)} \cdot \theta_n^{(t)} \quad \theta_n^{(t)} = (1 - \alpha)(\theta_n^{(t-1)} - \zeta) + \alpha w_n^{(t)} + \zeta$$

$$\theta_n^{(t)} = (1 - \alpha)(\theta_n^{(t-1)} - \zeta) + \alpha w_n^{(t)} + \zeta$$

$\{s_n^{(t)}\}_{t=1}^T$: Markov chain

$$p(x_n^{(t)} | s_n^{(t)}, \theta_n^{(t)}) = \delta(x_n^{(t)} - s_n^{(t)} \theta_n^{(t)})$$

$$p(x_n^{(t)}) = (1 - \lambda)\delta(x_n^{(t)}) + \lambda \mathcal{CN}(x_n^{(t)}; \zeta, \sigma^2).$$

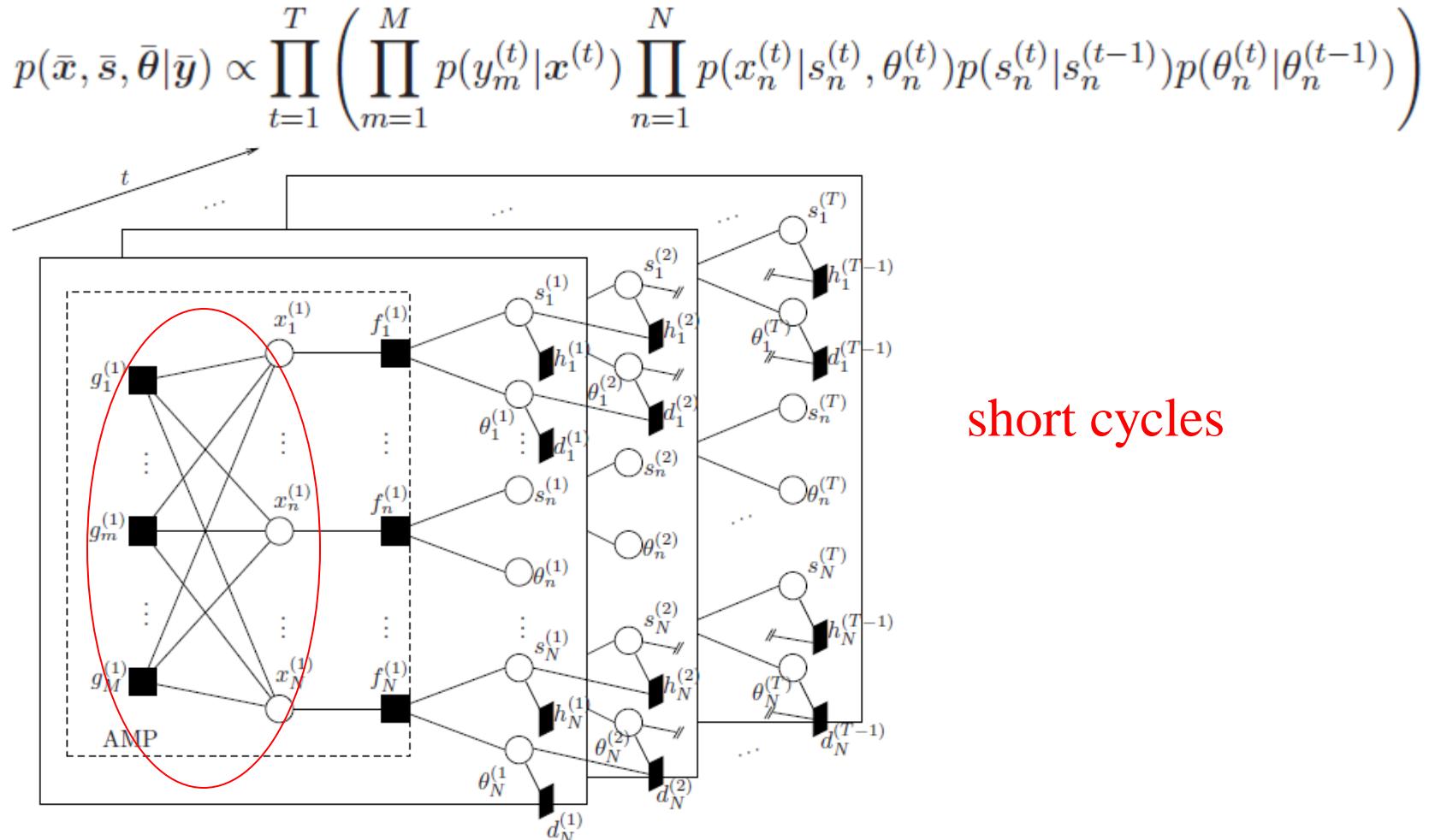
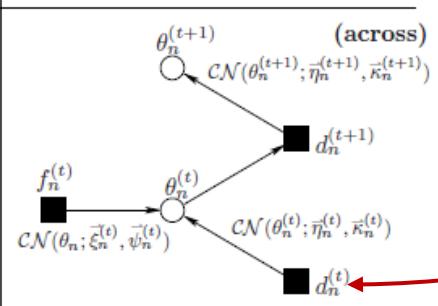
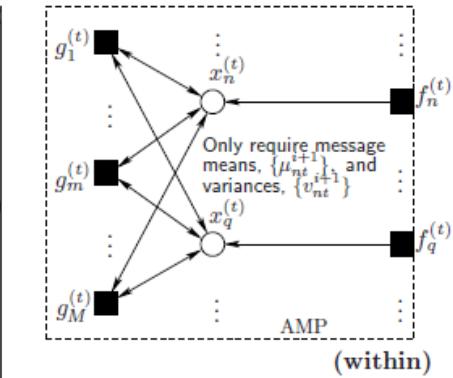
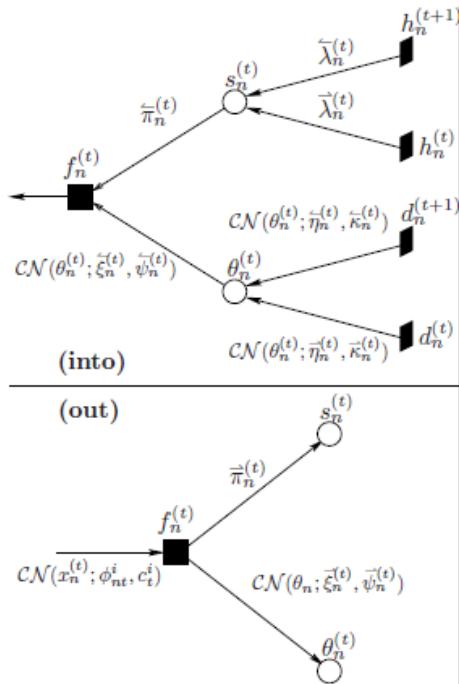


Fig. 1: Factor graph representation of the joint posterior distribution of (5).

Message scheduling

(into), (within), (out),
and (across)



Note that Message passing over each frame can be approximated by Turbo-CS/VBI/AMP

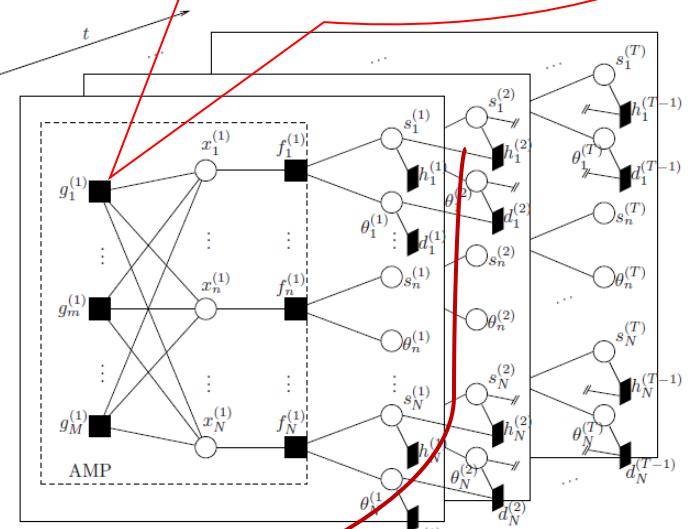


Fig. 1: Factor graph representation of the joint posterior distribution of (5).

transmitting messages across
neighboring frames



Application of SBL: FDD Massive MIMO Channel Estimation with Arbitrary 2D- Array Geometry

Jisheng Dai, An Liu and Vincent K.N. Lau, 'FDD Massive MIMO Channel Estimation with Arbitrary 2D-Array Geometry', IEEE Transactions on Signal Processing, vol. 66, no. 10, pp. 2584-2599, May 15, 15 2018.

FDD downlink channel estimation

System model

$$\mathbf{y}_k = \mathbf{X} \mathbf{h}_k + \mathbf{n}_k,$$

Training pilot symbols

$$\mathbf{X} \in \mathbb{C}^{T \times N}$$

$$\text{tr}(\mathbf{X} \mathbf{X}^H) = PTN$$

$$\mathbf{h}_k = \sum_{c=1}^{N_c} \sum_{s=1}^{N_s} \xi_{c,s}^k \mathbf{a}(\theta_{c,s}^k, \varphi_{c,s}^k),$$

$$\mathbf{a}(\theta) = \left[1, e^{-j2\pi \frac{a}{\lambda_d} \sin(\theta)}, \dots, e^{-j2\pi \frac{(N-1)d}{\lambda_d} \sin(\theta)} \right]^T$$

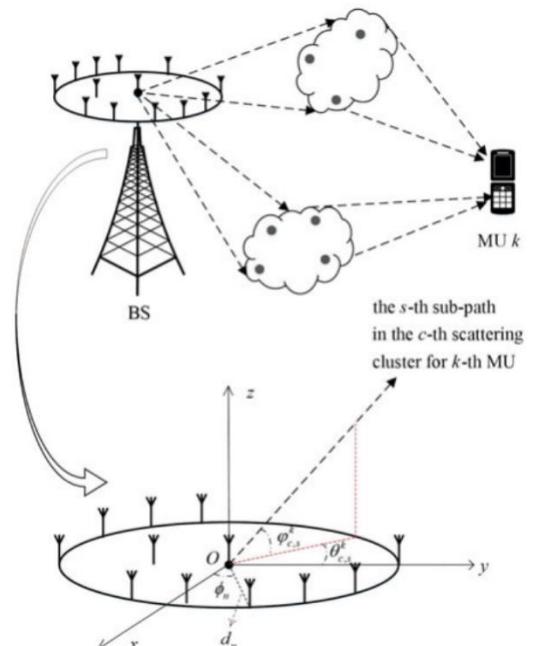


✓ sparse representation of \mathbf{h}_k $\mathbf{h}_k = \mathbf{F} \mathbf{t}_k$

$$\text{DFT matrix } \mathbf{F} \in \mathbb{C}^{N \times N}$$

$$\mathbf{F} = \left[\mathbf{f}\left(-\frac{1}{2}\right), \mathbf{f}\left(-\frac{1}{2} + \frac{1}{N}\right), \dots, \mathbf{f}\left(\frac{1}{2} - \frac{1}{N}\right) \right]$$

$$\mathbf{f}(x) = \frac{1}{\sqrt{N}} \left[1, e^{-j2\pi x}, \dots, e^{-j2\pi x(N-1)} \right]^T$$



✓ Off-Grid Basis for Massive MIMO Channels

the true azimuth AoDs $\{\theta_l, l = 1, 2, \dots, L\}$.

$$\theta_l = \hat{\vartheta}_{n_l} + \beta_{n_l}$$

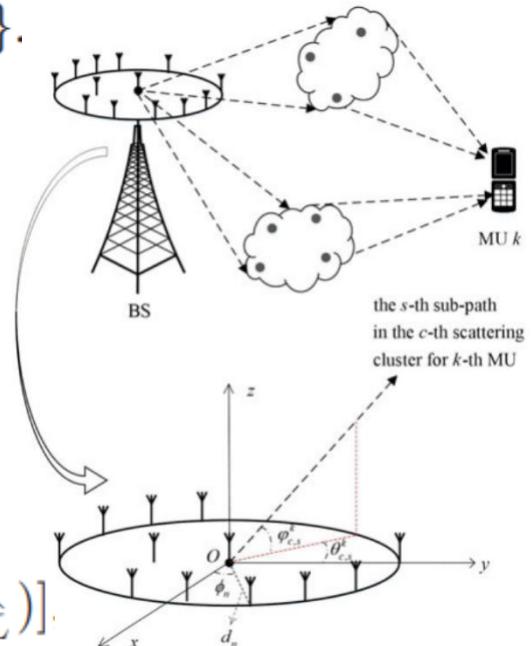
\mathbf{h} can be rewritten as $\mathbf{h} = \mathbf{A}(\boldsymbol{\beta})\mathbf{w}$

where

$$\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_{\hat{L}}]^T$$

$$\mathbf{A}(\boldsymbol{\beta}) = [\mathbf{a}(\hat{\vartheta}_1 + \beta_1), \mathbf{a}(\hat{\vartheta}_2 + \beta_2), \dots, \mathbf{a}(\hat{\vartheta}_{\hat{L}} + \beta_{\hat{L}})]$$

$$\beta_{n_l} = \begin{cases} \theta_l - \hat{\vartheta}_{n_l}, & l = 1, 2, \dots, L \\ 0, & \text{otherwise} \end{cases}$$



✓ The received signal can be rewritten by

$$\mathbf{y} = \mathbf{X}\mathbf{A}(\boldsymbol{\beta})\mathbf{w} + \mathbf{n} = \Phi(\boldsymbol{\beta})\mathbf{w} + \mathbf{n}$$



Sparse Bayesian Learning Formulation

$$p(\mathbf{y}|\mathbf{w}, \alpha, \boldsymbol{\beta}) = \mathcal{CN}(\mathbf{y}|\Phi(\boldsymbol{\beta})\mathbf{w}, \alpha^{-1}\mathbf{I})$$

- ✓ a Gamma hyperprior for the noise precision $\alpha = \sigma^{-2}$

$$p(\alpha) = \Gamma(\alpha; a, b)$$

- ✓ a noninformative uniform prior for $\boldsymbol{\beta}$

$$\boldsymbol{\beta} \sim U\left([- \frac{\pi}{2}, \frac{\pi}{2}]^{\hat{L}}\right)$$

- ✓ a non-stationary Gaussian prior distribution with a distinct precision γ_i for each element of \mathbf{w}

$$p(\mathbf{w}|\boldsymbol{\gamma}) = \mathcal{CN}(\mathbf{w}|\mathbf{0}, \text{diag}(\boldsymbol{\gamma}^{-1})) \quad p(\boldsymbol{\gamma}) = \prod_{i=1}^{\hat{L}} \Gamma(\gamma_i; 1 + a, b).$$



- ✓ find the most-probable values α^* , γ^* and β^* together by maximizing the posteriori $p(\alpha, \gamma, \beta | y)$

$$(\alpha^*, \gamma^*, \beta^*) = \arg \max_{\alpha, \gamma, \beta} \ln p(y, \alpha, \gamma, \beta)$$

high-dimensional
non-convex function

$$\mathcal{U}(\alpha, \gamma, \beta | \dot{\alpha}, \dot{\gamma}, \dot{\beta}) \longrightarrow$$

surrogate function (lower bound) for
the objective function $\ln p(y, \alpha, \gamma, \beta)$

$$= \int p(w|y, \dot{\alpha}, \dot{\gamma}, \dot{\beta}) \ln \frac{p(w, y, \alpha, \gamma, \beta)}{p(w|y, \dot{\alpha}, \dot{\gamma}, \dot{\beta})} dw,$$

$$p(w|y, \alpha, \gamma, \beta) = \mathcal{CN}(w|\mu(\alpha, \gamma, \beta), \Sigma(\alpha, \gamma, \beta))$$

$$\mu(\alpha, \gamma, \beta) = \alpha \Sigma(\alpha, \gamma, \beta) \Phi^H(\beta) y,$$

$$\Sigma(\alpha, \gamma, \beta) = (\alpha \Phi^H(\beta) \Phi(\beta) + \text{diag}(\gamma))^{-1}$$

$$\alpha^{(i+1)} = \arg \max_{\alpha} \mathcal{U}(\alpha, \gamma^{(i)}, \beta^{(i)} | \alpha^{(i)}, \gamma^{(i)}, \beta^{(i)}), \quad (31)$$

$$\gamma^{(i+1)} = \arg \max_{\gamma} \mathcal{U}(\alpha^{(i+1)}, \gamma, \beta^{(i)} | \alpha^{(i+1)}, \gamma^{(i)}, \beta^{(i)}), \quad (32)$$

$$\beta^{(i+1)} = \arg \max_{\beta} \mathcal{U}(\alpha^{(i+1)}, \gamma^{(i+1)}, \beta | \alpha^{(i+1)}, \gamma^{(i+1)}, \beta^{(i)})$$

**the block MM
algorithm**



Update for α

convex

has a simple and closed-form solution

$$\alpha^{(i+1)} = \frac{T + a}{b + \eta(\alpha^{(i)}, \gamma^{(i)}, \beta^{(i)})}$$

$$\eta(\alpha, \gamma, \beta) = \text{tr} (\Phi(\beta) \Sigma(\alpha, \gamma, \beta) \Phi^H(\beta))$$

$$+ \|y - \Phi(\beta) \mu(\alpha, \gamma, \beta)\|_2^2.$$

Update for γ

convex

$$\gamma_l^{(i+1)} = \frac{a + 1}{b + [\Xi(\alpha^{(i+1)}, \gamma^{(i)}, \beta^{(i)})]_{ll}}, \forall l,$$

$$\Xi(\alpha, \gamma, \beta) = \Sigma(\alpha, \gamma, \beta) + \mu(\alpha, \gamma, \beta) \mu^H(\alpha, \gamma, \beta)$$

Update for β

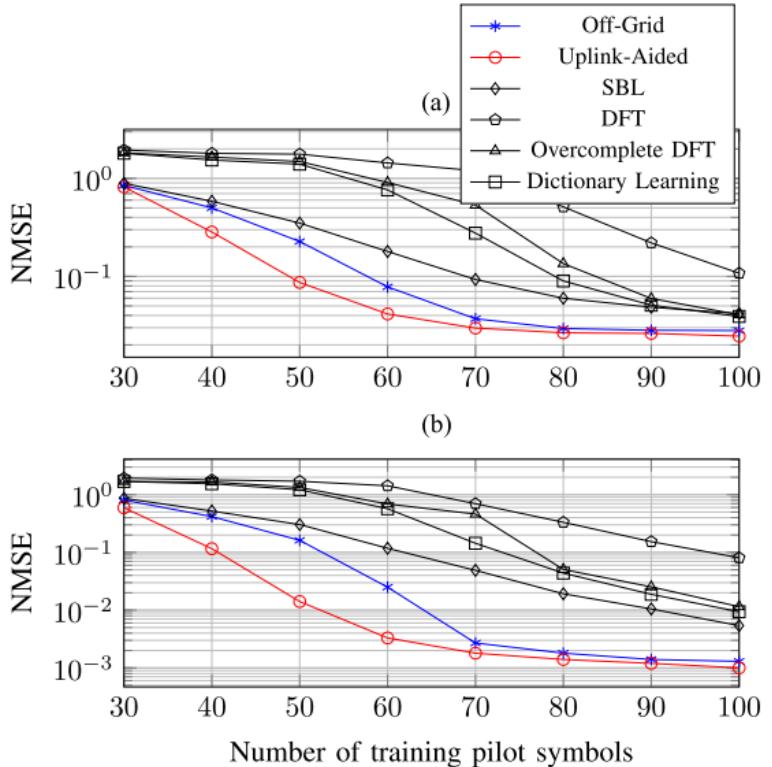


one-step update for β

$$\boldsymbol{\beta}^{(i+1)} = \boldsymbol{\beta}^{(i)} + \Delta_{\beta} \cdot \boldsymbol{\zeta}_{\beta}^{(i)}$$

a fixed stepsize to update β :

$$\boldsymbol{\beta}^{(i+1)} = \boldsymbol{\beta}^{(i)} + \frac{r_{\theta}}{100} \cdot \text{sign}(\boldsymbol{\zeta}_{\beta}^{(i)})$$



NMSE of dowlink channel estimate versus the number of training pilot symbols for ULA. (a) SNR = 0 dB; (b) SNR = 10 dB.

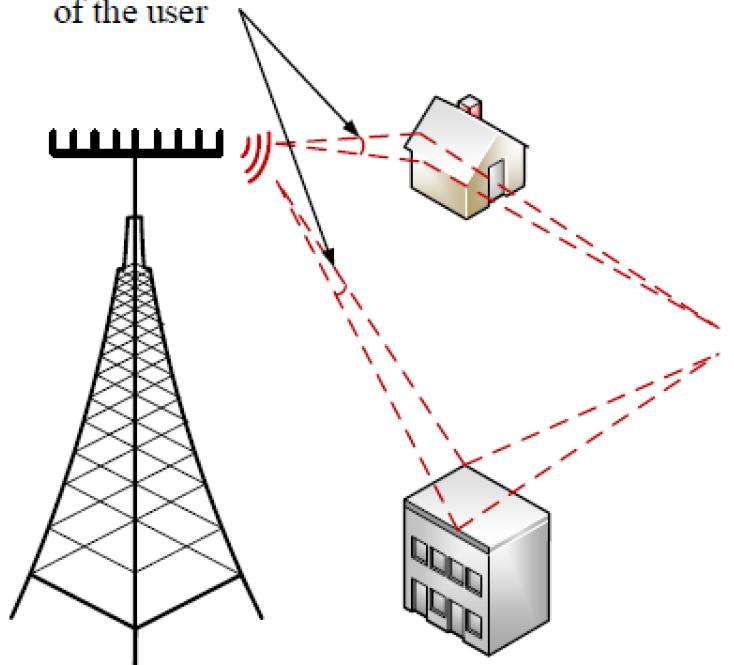


Application of Turbo CS: Downlink Massive MIMO- OFDM Channel Estimation

Xiaoyan Kuai, Lei Chen, Xiaojun Yuan and An Liu, 'Structured Turbo Compressed Sensing for Downlink Massive MIMO-OFDM Channel Estimation', in IEEE Transactions on Wireless Communications, vol. 18, no. 8, pp. 3813-3826, Aug. 2019.

System Model for Downlink Massive MIMO-OFDM

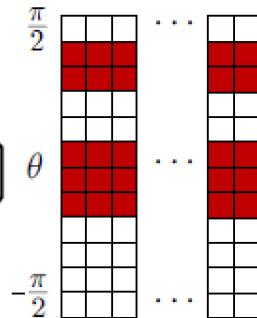
Angular of departure
of the user



Base station equipped
with massive antennas

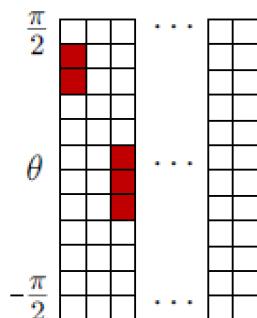
Distributed
scatterings

Massive MIMO channel
of the user in the angular-
frequency domain



Index of the pilot carriers

Massive MIMO channel
of the user in the angular-
delay domain

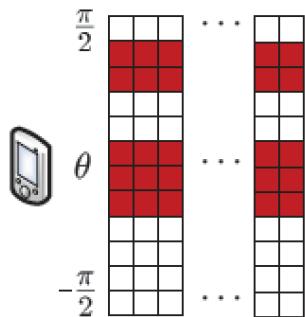


Index of the delay taps

$$\mathbf{F} \Leftrightarrow$$

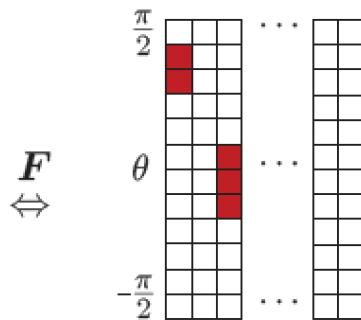
System Model for Downlink Massive MIMO-OFDM

Massive MIMO channel
of the user in the angular-
frequency domain



Index of the pilot carriers

Massive MIMO channel
of the user in the angular-
delay domain



Index of the delay taps

$$\Leftrightarrow \mathbf{F}$$

- Transmitter: BS with N antennas and P pilot subcarriers.
- Receiver: Users with a single antenna.
- Pilot Transmission: T time slots.

- The received signal is given by

$$\mathbf{Y}_f = \mathbf{X} \mathbf{F}^H \mathbf{H}_f + \mathbf{W}_f,$$

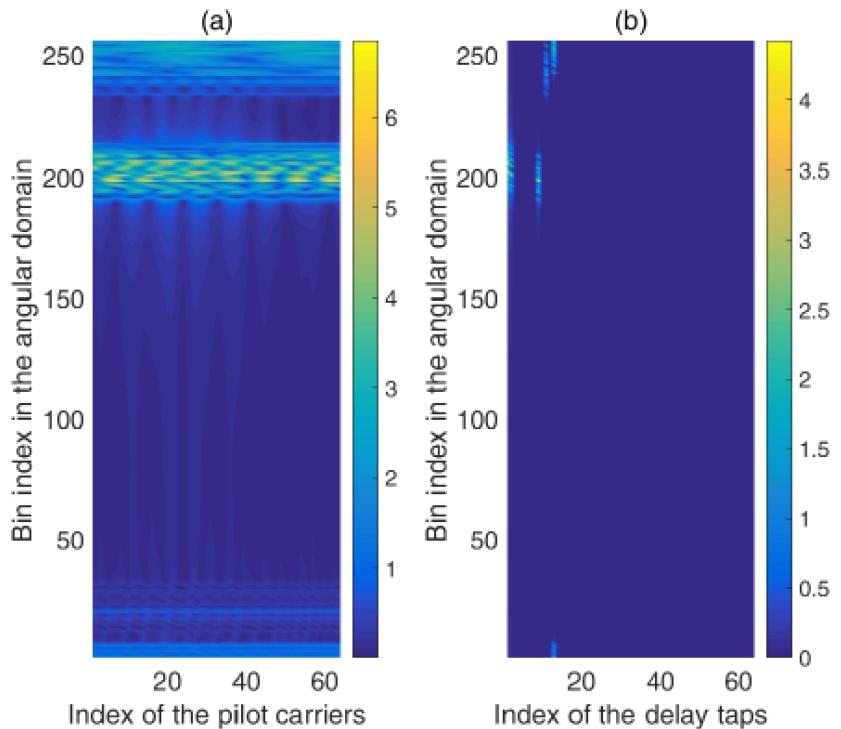
$\mathbf{H}_f \in \mathbb{C}^{N \times P}$: Channel in the **angular-frequency** domain.

- The received signal can be given by

$$\mathbf{Y}_d = \mathbf{X} \mathbf{F}^H \mathbf{H}_d + \mathbf{W}_d,$$

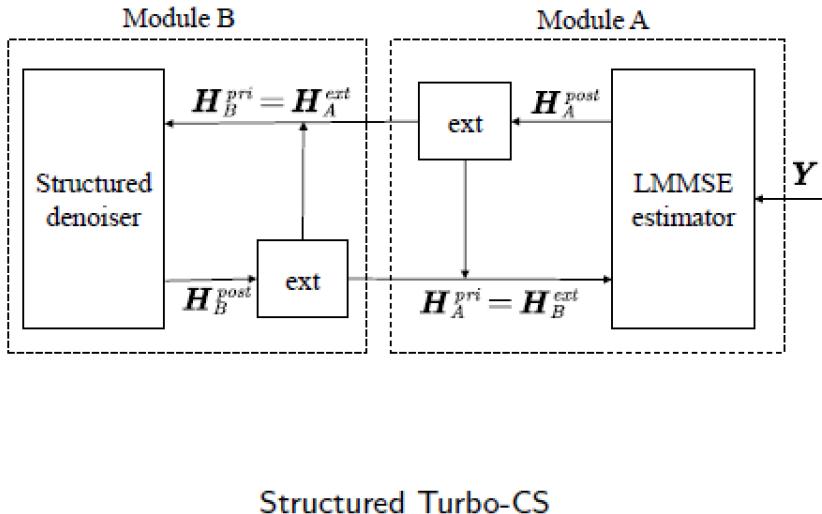
$\mathbf{H}_d \in \mathbb{C}^{N \times P}$: Channel in the **angular-delay** domain.

Structured Sparsity of Massive MIMO-OFDM Chan



- An example of channel coefficients in angular-frequency domain and angular-delay domain.
- Angular-frequency channel matrix H_f is sparse and clustered in angular domain, and the clustered structure is similar for different subcarrier indexes.
- Angular-delay channel matrix H_d has two taps: zero taps and non-zero taps. Non-zero taps are sparse and clustered in angular domain.
- Different structured estimators need to be designed!

Structured Turbo Compressed Sensing



- In structured Turbo-CS, each pair of posterior mean and variance depend on all prior input \mathbf{H}_B^{pri} .

- X. Kuai, L. Chen, X. Yuan and A. Liu, "Structured Turbo Compressed Sensing for Downlink Massive MIMO-OFDM Channel Estimation," IEEE Transactions on Wireless Communications, vol. 18, no. 8, pp. 3813-3826, Aug. 2019.

Algorithm 3 Structured Turbo-CS Algorithm

Inputs: $\mathbf{Y}, \mathbf{X}^{(p)}, \forall p, \sigma^2$

Initialize: $\mathbf{A}^{(p)} = \mathbf{X}^{(p)}\mathbf{F}^H, \mathbf{h}_A^{pri(p)}, v_A^{pri(p)}$
 for $t = 0, 1, 2, 3 \dots$ do

% Module A: LMMSE estimator

$$\begin{aligned}\mathbf{h}_A^{post(p)} &= \mathbf{h}_A^{pri(p)} \\ &+ \frac{v_A^{pri(p)}}{v_A^{pri(p)} + \sigma^2} \mathbf{A}^{(p)H} (\mathbf{y}^{(p)} - \mathbf{A}^{(p)} \mathbf{h}_A^{pri(p)}) \\ v_A^{post(p)} &= v_A^{pri(p)} - \frac{M}{N} \cdot \frac{(v_A^{pri(p)})^2}{v_A^{pri(p)} + \sigma^2} \\ v_B^{pri(p)} &= v_A^{ext(p)} = \left(\frac{1}{v_A^{post(p)}} - \frac{1}{v_A^{pri(p)}} \right)^{-1} \\ \mathbf{h}_B^{pri(p)} &= \mathbf{h}_A^{ext(p)} = v_B^{pri(p)} \left(\frac{\mathbf{h}_A^{post(p)}}{v_A^{post(p)}} - \frac{\mathbf{h}_A^{pri(p)}}{v_A^{pri(p)}} \right)\end{aligned}$$

% Module B: Structured denoiser

$\mathbf{h}_{B,n}^{post(p)}$ and $v_B^{post(p)}$ are determined by
 structured estimator

$$v_A^{pri} = v_B^{ext} = \left(\frac{1}{v_B^{post}} - \frac{1}{v_B^{pri}} \right)^{-1}$$

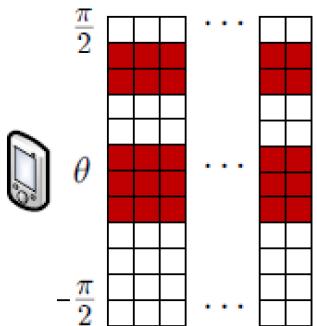
$$\mathbf{h}_A^{pri(p)} = \mathbf{h}_B^{ext(p)} = v_A^{pri(p)} \left(\frac{\mathbf{h}_B^{post(p)}}{v_B^{post(p)}} - \frac{\mathbf{h}_B^{pri(p)}}{v_B^{pri(p)}} \right)$$

end for

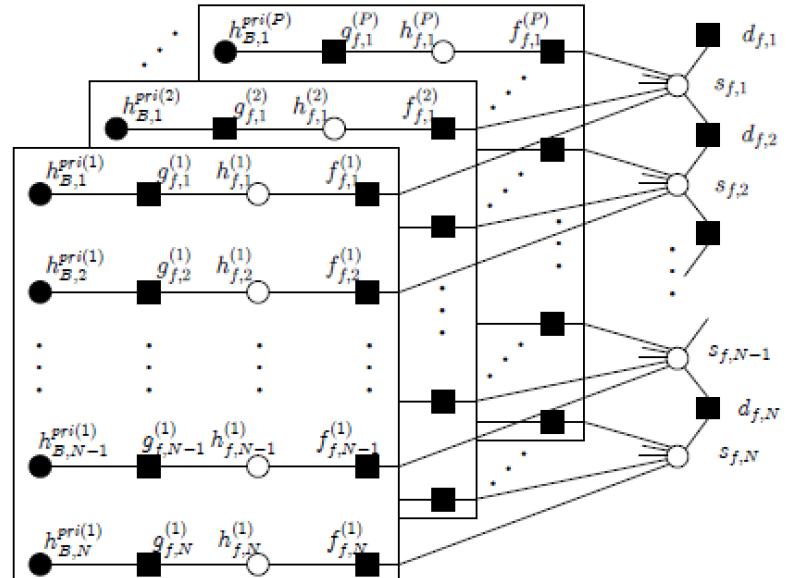
Factor Graph of the Frequency Support Denoiser

- In frequency support model, all the subcarriers share a common Markov chain.

Massive MIMO channel
of the user in the angular-
frequency domain



Index of the pilot carriers



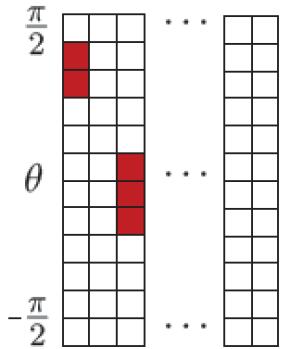
$$g_{f,n}^{(p)} = \mathcal{CN}(h_{f,n}^{(p)}; h_{B,n}^{\text{pri}(p)}, v_B^{\text{pri}(p)}), \quad f_{f,n}^{(p)} = (1-s_{f,n})\delta(h_{f,n}^{(p)}) + s_{f,n}\mathcal{CN}(h_{f,n}^{(p)}; 0, (\sigma_f^{(p)})^2)$$

$$d_{f,1} = (1 - \lambda_f)^{1-s_{f,1}} (\lambda_f)^{s_{f,1}}, \quad d_{f,n} = \begin{cases} (1-p_{10})^{1-s_{f,n}} (p_{10})^{s_{f,n}}, & s_{f,n-1}=0 \\ (p_{01})^{1-s_{f,n}} (1-p_{01})^{s_{f,n}}, & s_{f,n-1}=1 \end{cases}$$

Factor Graph of the Delay Support Denoiser

- In delay support model, another variable node is added to decide the tap is non-zero or not.

Massive MIMO channel of the user in the angular-delay domain



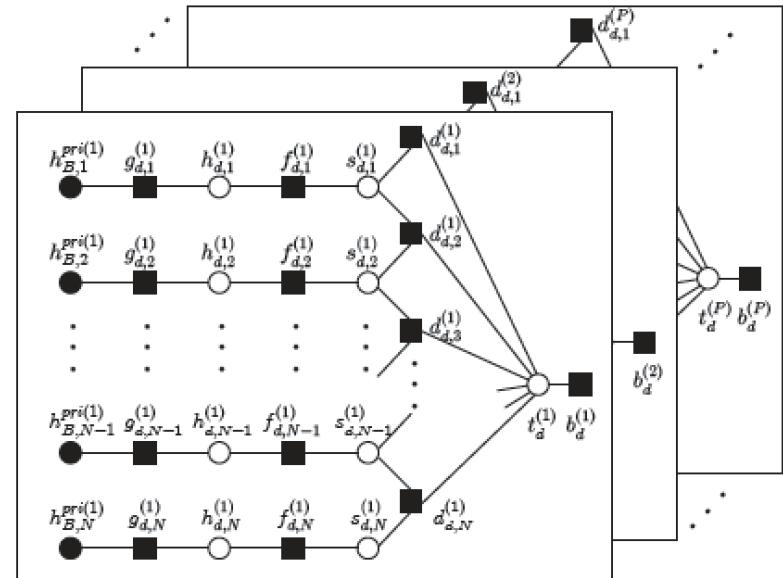
Index of the delay taps

$$g_{d,n}^{(p)} = \mathcal{CN}(h_{d,n}^{(p)}; h_{B,n}^{\text{pri}(p)}, v_B^{\text{pri}(p)}),$$

$$f_{d,n}^{(p)} = (1 - s_{d,n}^{(p)})\delta(h_{d,n}^{(p)}) + s_{d,n}^{(p)}\mathcal{CN}(h_{d,n}^{(p)}; 0, (\sigma_d^{(p)})^2)$$

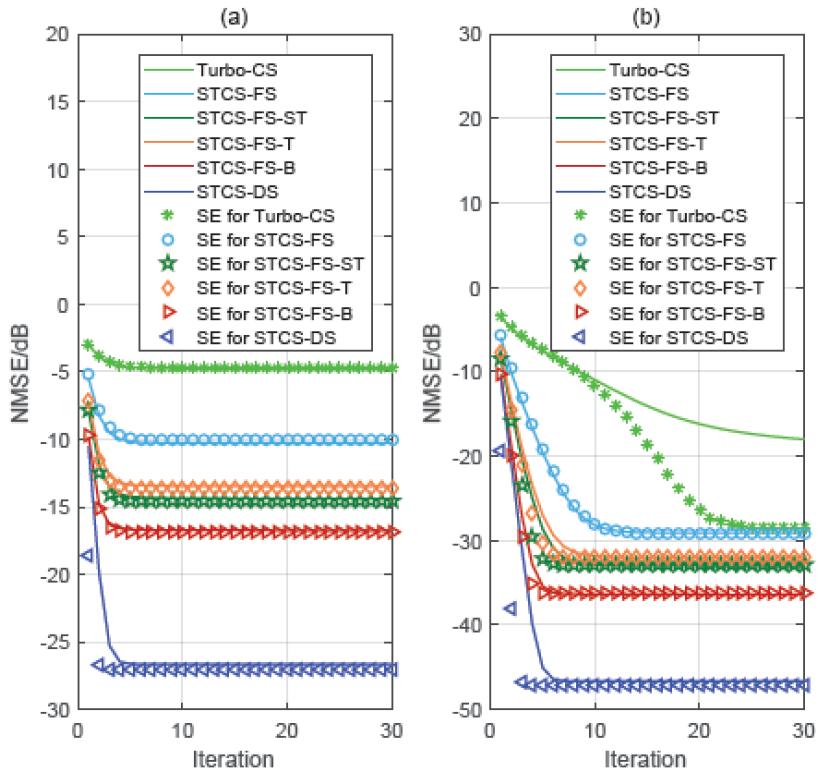
$$d_{d,1}^{(p)} = (1 - \lambda_d^{(p)})^{(1 - s_{d,1}^{(p)})t_d^{(p)}} (\lambda_d^{(p)})^{s_{d,1}^{(p)}t_d^{(p)}}$$

$$\cdot 1^{(1 - s_{d,1}^{(p)})(1 - t_d^{(p)})} 0^{s_{d,1}^{(p)}(1 - t_d^{(p)})}$$



$$b_d^{(p)} = (1 - \gamma_d^{(p)})^{1 - t_d^{(p)}} (\gamma_d^{(p)})^{t_d^{(p)}} \\ d_{d,n}^{(p)} = \begin{cases} (1 - p_{10}^{(p)})^{1 - s_{d,n}^{(p)}} (p_{10}^{(p)})^{s_{d,n}^{(p)}}, s_{d,n-1}^{(p)} = 0, t_d^{(p)} = 1 \\ (p_{01}^{(p)})^{1 - s_{d,n}^{(p)}} (1 - p_{01}^{(p)})^{s_{d,n}^{(p)}}, s_{d,n-1}^{(p)} = 1, t_d^{(p)} = 1 \\ 1^{1 - s_{d,n}^{(p)}} 0^{s_{d,n}^{(p)}}, s_{d,n-1}^{(p)} = 0, t_d^{(p)} = 0 \\ 1^{1 - s_{d,n}^{(p)}} 0^{s_{d,n}^{(p)}}, s_{d,n-1}^{(p)} = 1, t_d^{(p)} = 0 \end{cases}$$

Structured Sparse Signal Recovery



- SNR = 10dB in (a) and 30dB in (b); $N = 256$, $T = 103$, 4 nonzero delay taps with transition probability $p_{01}^{(p)} = 1/16$ and $p_{10}^{(p)} = 1/240$ (i.e., $\lambda_f = 0.25$).
- Structured Turbo-CS (STCS) based algorithms can be predicted by SE for a relatively small N .
- Structured Turbo-CS for delay support (STCS-DS) achieves superior performance by exploiting a very sparse channel coefficients in the angular-delay domain.



Application of Turbo VBI: Cloud-Assisted Cooperative Localization for Vehicle Platoons

An Liu, Lixiang Lian, Vincent Lau, Guanying Liu, and Min-Jian Zhao, 'Cloud-Assisted Cooperative Localization for Vehicle Platoons: A Turbo Approach', IEEE Transactions on Signal Processing, vol. 68, pp. 605-620, 2020.

Architecture of Cloud-Assisted Localization System

- The received pilot signals at BS n

$$z_n(t) = \sum_{k=1}^K \tilde{h}_{n,k} s_k(t) + w_n(t), t = 1, \dots, T,$$

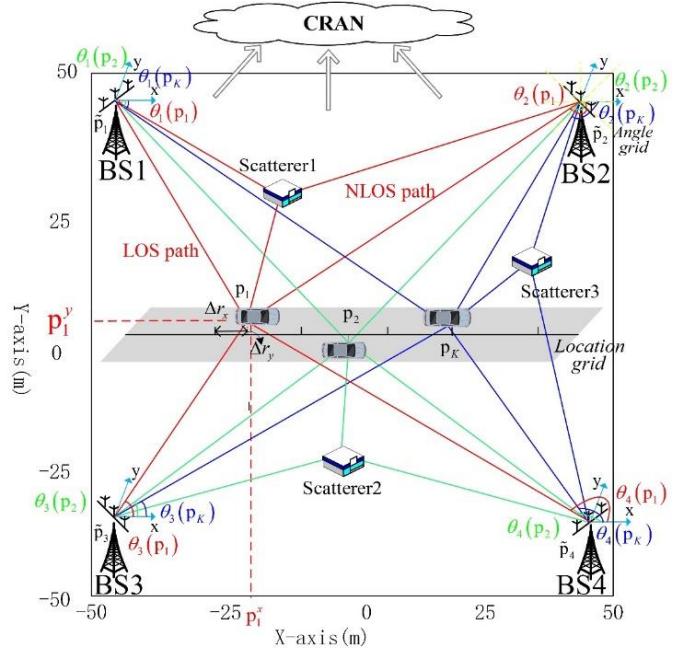
Pilot transmitted from vehicle

- AoA is related to the position of vehicle

$$\theta_n(p_k) = \arctan \left(\frac{p_k^y - \bar{p}_n^y}{p_k^x - \bar{p}_n^x} \right) + \pi \cdot \mathbb{I}(p_k^x < \bar{p}_n^x)$$

- The channel vector is modeled as

$$\tilde{h}_{n,k} = h_{n,k} a_n(\theta_n(p_k)) + \sum_{l=1}^{L_{n,k}} h_{n,k}^l a_n(\theta_{n,k}^l),$$



- The channel vector has a sparse representation given by

$$\mathbf{A}_n^L(\Delta r) = [a_n(\theta_n(r_1 + \Delta r_1)), \dots, a_n(\theta_n(r_Q + \Delta r_Q))]$$

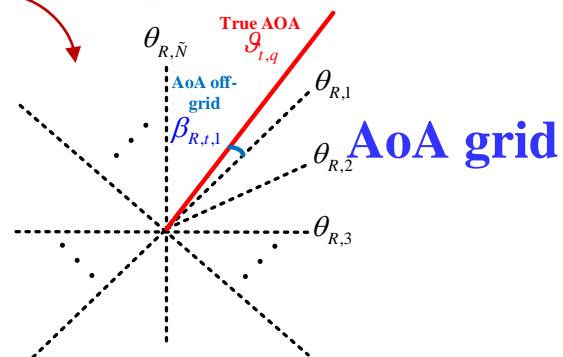
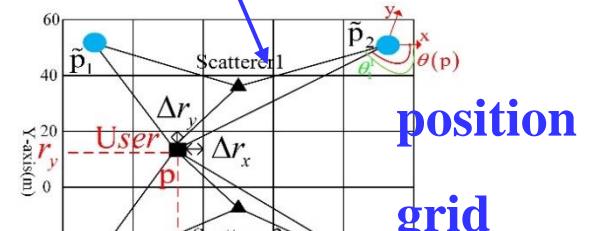
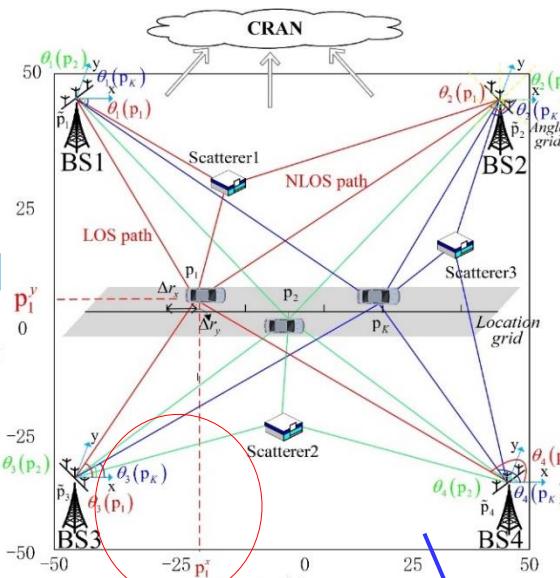
$$\mathbf{A}_n^{NL}(\Delta \vartheta_{n,k}) = \left[a_n(\vartheta_1 + \Delta \vartheta_{n,k}^1), \dots, a_n(\vartheta_{\widetilde{M}} + \Delta \vartheta_{n,k}^{\widetilde{M}}) \right]^T$$

$$\Delta \mathbf{r}_{q_k} = \tilde{\mathbf{p}}_k - \mathbf{r}_{q_k}, k = 1, \dots, K$$

$$\Delta \vartheta_{n,k}^{m_l^l} = \theta_{n,k}^l - \vartheta_{m_l^l, k}, l = 1, \dots, L_{n,k}$$

$$\Delta \mathbf{r} = [\Delta \mathbf{r}_1; \dots; \Delta \mathbf{r}_Q] \quad \Delta \vartheta_{n,k} = [\Delta \vartheta_{n,k}^1, \dots, \Delta \vartheta_{n,k}^{\widetilde{M}}]^T$$

$$\tilde{\mathbf{h}}_{n,k} = \mathbf{A}_n^L(\Delta \mathbf{r}) \mathbf{x}_{n,k} + \mathbf{A}_n^{NL}(\Delta \vartheta_{n,k}) \mathbf{y}_{n,k}$$



- CS model with unknown parameters in the measurement matrix

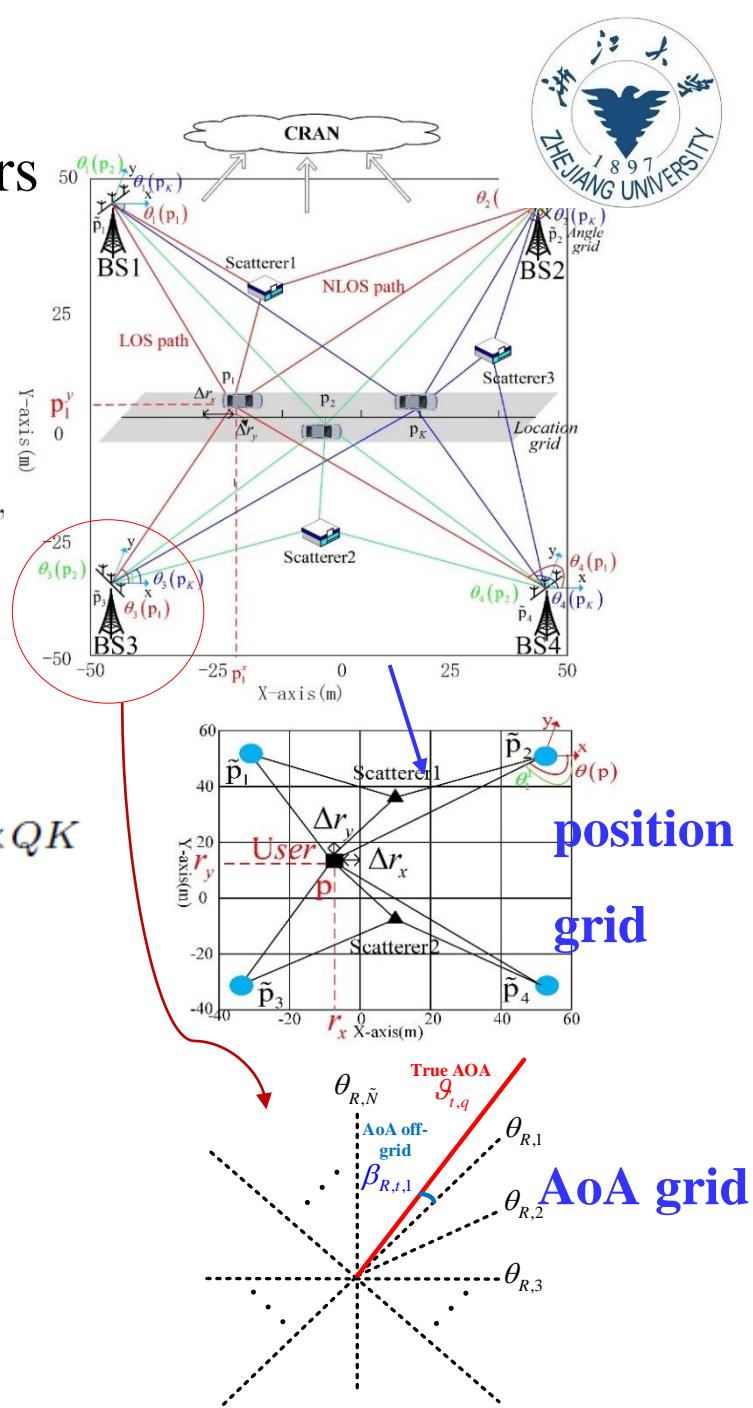
$$z_n = A_n(\Delta r, \Delta \vartheta_n) \begin{bmatrix} x_n \\ y_n \end{bmatrix} + w_n, \forall n,$$

$$A_n(\Delta r, \Delta \vartheta_n) = \begin{bmatrix} A_n^L(1) & A_n^{NL}(1) \\ \vdots & \vdots \\ A_n^L(T) & A_n^{NL}(T) \end{bmatrix} \in \mathbb{C}^{MT \times (Q+\tilde{M})K},$$

$$\boldsymbol{x}_n = [x_{n,1}; \dots; x_{n,K}]$$

$$\boldsymbol{y}_n = [y_{n,1}; \dots; y_{n,K}]$$

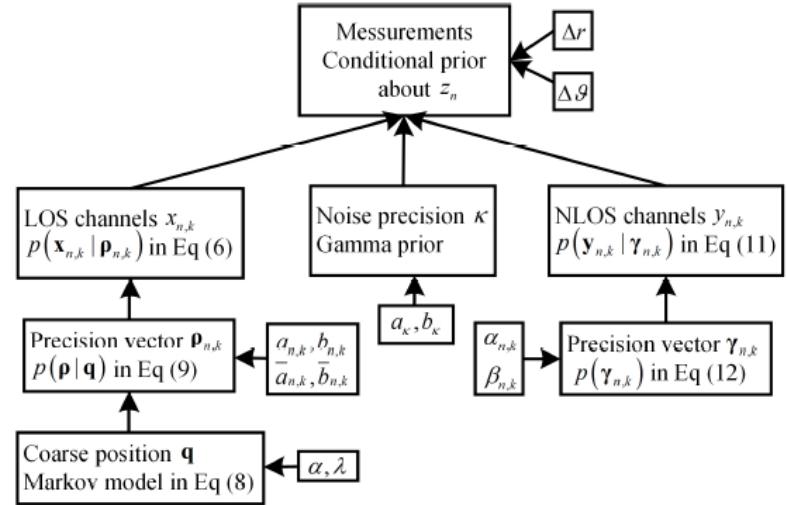
$$A_n^L(t) = [s_1(t) A_n^L(\Delta r) \quad \dots \quad s_K(t) A_n^L(\Delta r)] \in \mathbb{C}^{M \times QK}$$



Prior model for vehicle platoon cooperative localization

$$p(\mathbf{x}, \boldsymbol{\rho}, q | \phi) = \underbrace{p(q | \phi)}_{\text{Support}} \underbrace{p(\boldsymbol{\rho} | q)}_{\text{Precision}} \underbrace{p(\mathbf{x} | \boldsymbol{\rho})}_{\text{Sparse signal}}$$

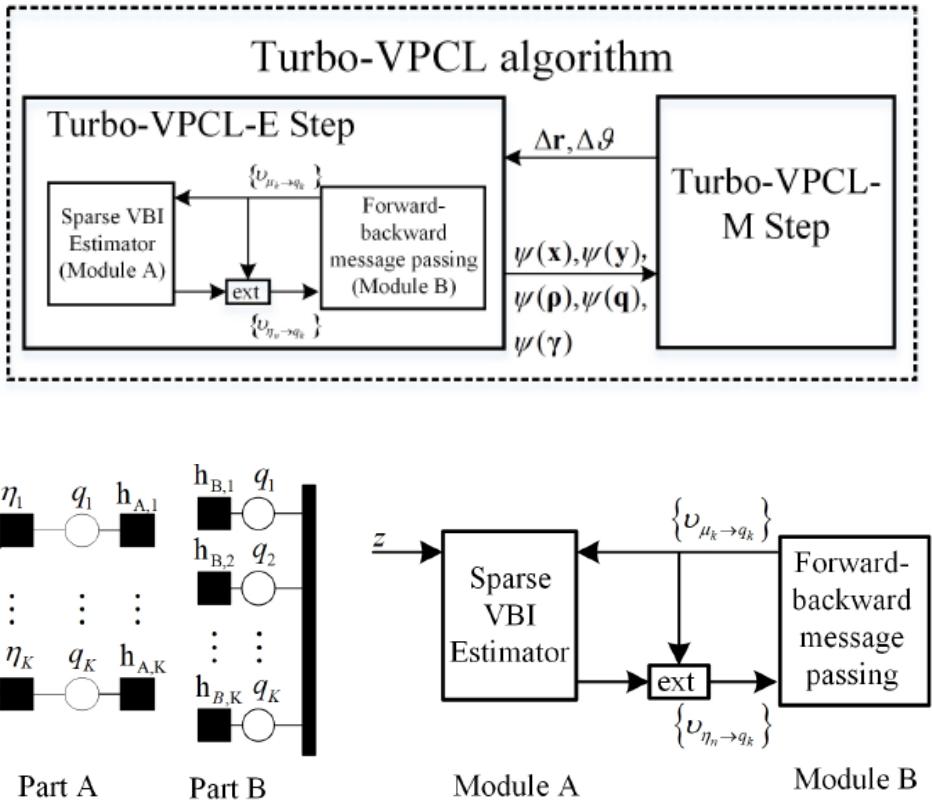
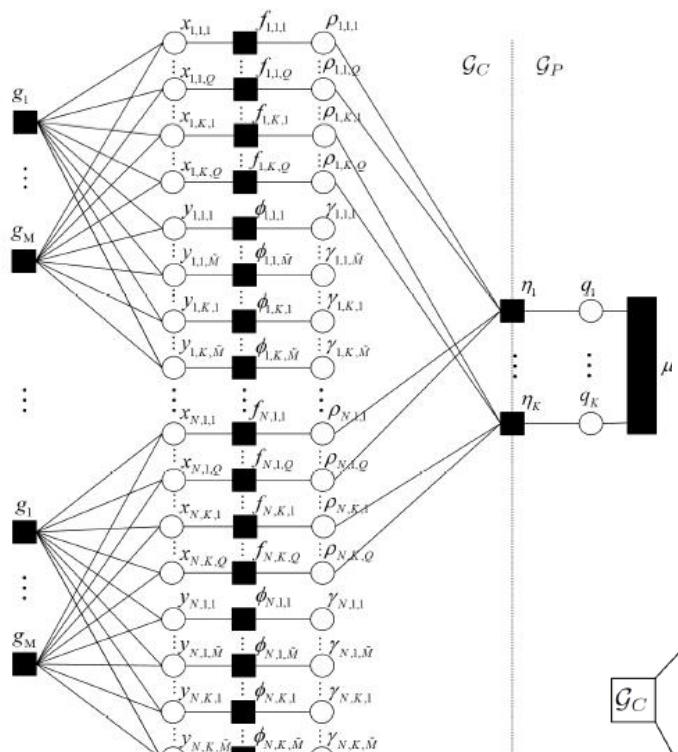
$$p(\mathbf{q}) = p(q_1) \prod_{k=1}^{K-1} p(q_{k+1} | q_k)$$



- Goal: estimate the position parameter and position offset given the observations $\mathbf{z} = [\mathbf{z}_1; \dots; \mathbf{z}_N]$

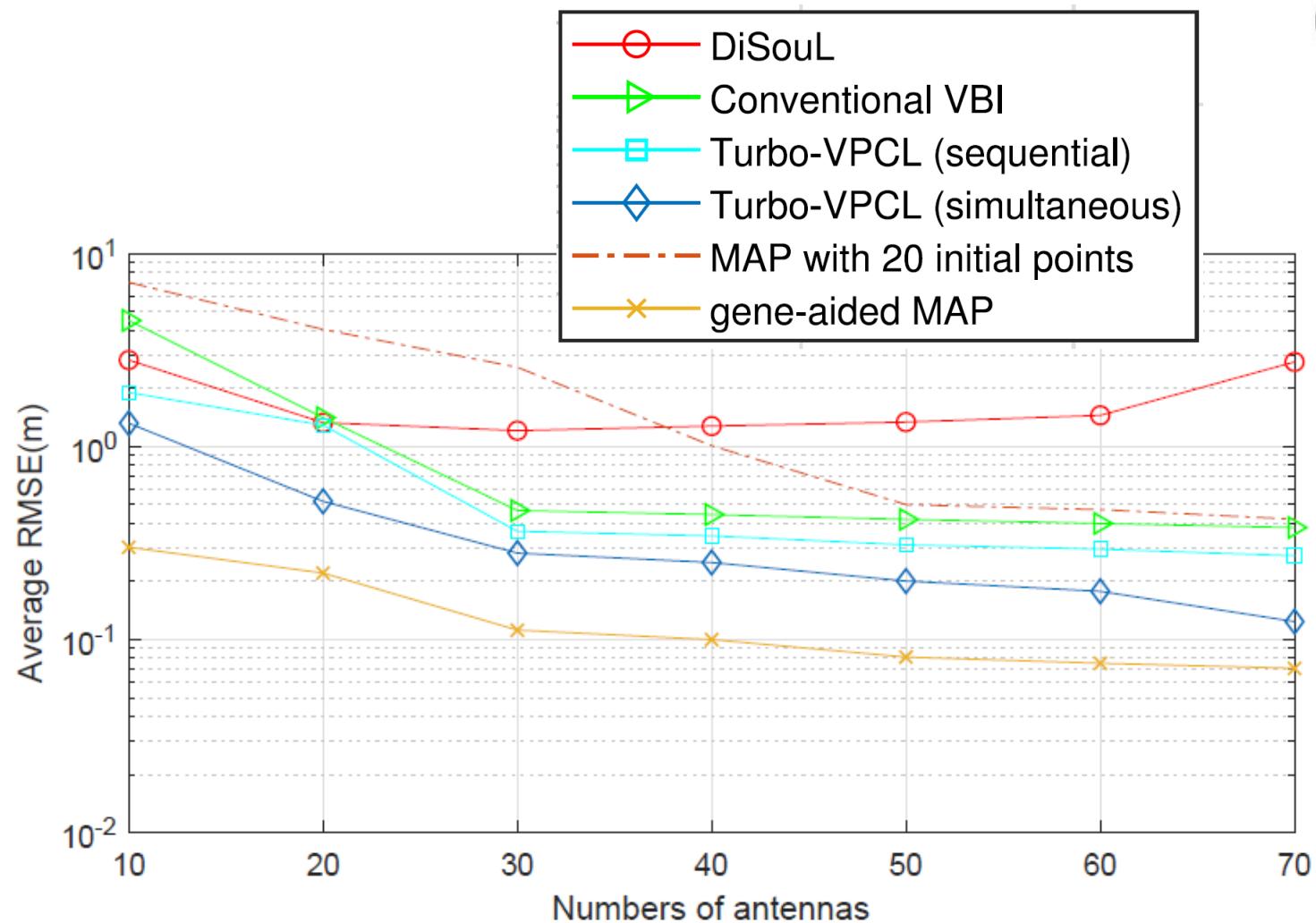
$$p(q_k | \mathbf{z}, \Delta r, \Delta \vartheta) \propto \int_{\mathbf{x}, \boldsymbol{\rho}, \mathbf{y}, \boldsymbol{\gamma}, \mathbf{q}_{-k}, \kappa} p(z, x, \rho, q, y, \gamma, \kappa | \Delta r, \Delta \vartheta) d_{-q_k}$$

$$\begin{aligned} [\Delta r^*, \Delta \vartheta^*] &= \operatorname{argmax}_{\Delta r, \Delta \vartheta} \ln p(z | \Delta r, \Delta \vartheta) \\ &= \operatorname{argmax}_{\Delta r, \Delta \vartheta} \ln \int_{\mathbf{v}} p(z, \mathbf{v} | \Delta r, \Delta \vartheta) d\mathbf{v}, \end{aligned}$$



(a)

(b)



Average RMSE of different algorithms versus number of antennas M with $L_{n,k} = 4$ and $P_T = 0$ dB.



Application of Dynamic Bayesian Learning: Robust Multi-user Channel Tracking Scheme for 5G New Radio

See “lecture note appendix 2” and
Yubo Wan, Guanying Liu, An Liu and Min-Jian Zhao, “Robust Multi-user Channel
Tracking Scheme for 5G New Radio”, submitted to IEEE Trans. Wireless
Communications, 2023.



Joint Target Detection and Channel Estimation for Integrated Sensing and Communication Systems

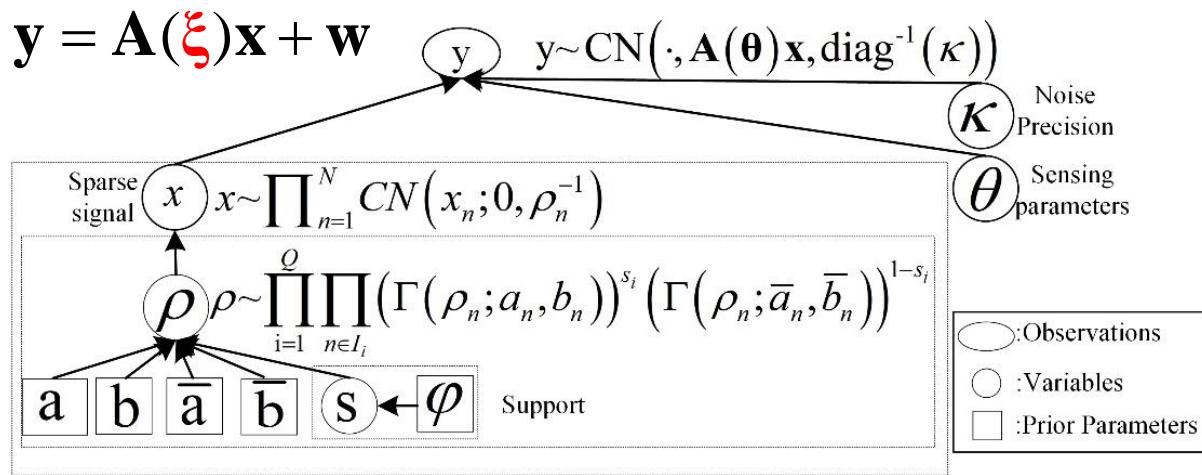
See “lecture note appendix 3” and

- [1] Zhe Huang, Kexuan Wang, An Liu, Yunlong Cai, Rui Du and Tony Xiao Han, “Joint Pilot Optimization, Target Detection and Channel Estimation for Integrated Sensing and Communication Systems”, IEEE TWC, online: <https://arxiv.org/abs/2202.02688>
- [2] W. Xu, A. Liu, et al. “Joint scattering environment sensing and channel estimation based on non-stationary Markov random field”, Submitted to IEEE TWC. [Online]. Available: <https://arxiv.org/abs/2302.02587>.



Summary

EM-based Problem Formulation



- ✓ Computing the conditional marginal posteriors

$$p(s_i | \mathbf{y}, \xi) \propto \sum \iint p(\mathbf{x}, \rho, \mathbf{s} | \phi) p(\mathbf{y} | \mathbf{x}, \xi) d\rho d\mathbf{x}$$

$$p(\mathbf{x} | \mathbf{y}, \xi) \propto \sum_{\mathbf{s}} \int p(\mathbf{y}, \mathbf{x}, \rho, \mathbf{s} | \xi) d\rho = \sum_{\mathbf{s}} \int p(\mathbf{x}, \rho, \mathbf{s} | \phi) p(\mathbf{y} | \mathbf{x}, \xi) d\rho$$

$\mathbf{v} = \{x, \rho, s\}$ is the collection of variables.

- ✓ MAP estimation of the uncertain parameters ξ

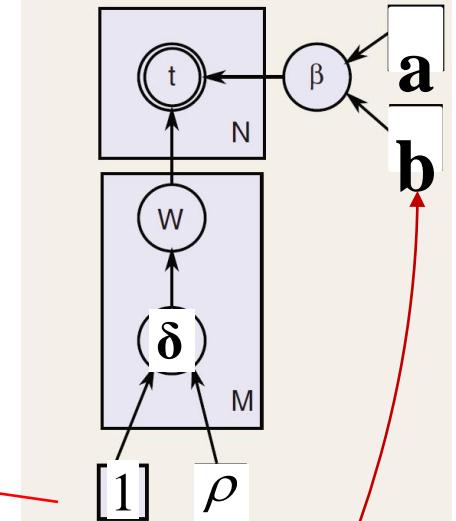
$$\xi^* = \underset{\xi}{\operatorname{argmax}} \ln p(\xi | \mathbf{y}) = \underset{\xi}{\operatorname{argmax}} \ln \sum_{\mathbf{s}} \iint p(\mathbf{y}, \mathbf{v}, \xi) d\rho d\mathbf{x}$$



Prior distribution for SBL/VBI $\mathbf{t} = \Phi\mathbf{w} + \mathbf{n}$

$$p(\mathbf{w}|\boldsymbol{\delta}) = CN(\mathbf{w}|\mathbf{0}, diag(\boldsymbol{\delta}))$$

$$p(\boldsymbol{\delta}) = \prod_i \Gamma(\delta_i; 1, \rho), \quad \rho = 0.01$$



$$p(\mathbf{n}) = CN(\mathbf{n}|\mathbf{0}, \sigma^2 \mathbf{I}) = CN(\mathbf{n}|\mathbf{0}, \beta^{-1} \mathbf{I})$$

$$p(\mathbf{t}|\mathbf{w}) = CN(\mathbf{t}|\Phi\mathbf{w}, \beta^{-1} \mathbf{I})$$

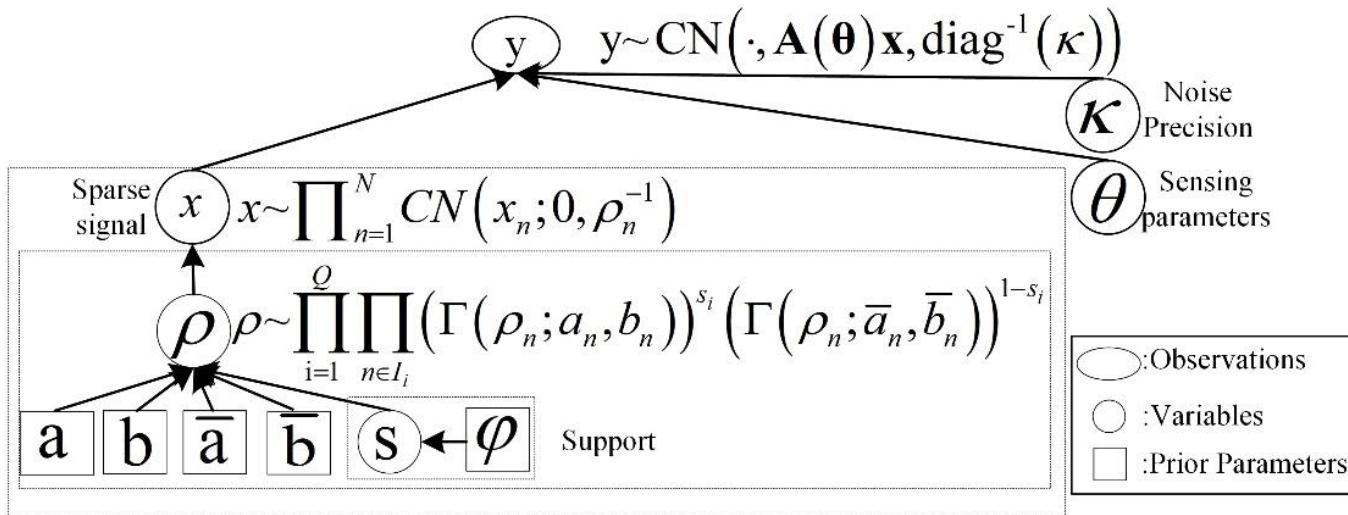
$$p(\beta) = \Gamma(\beta; a, b), \quad a = b = 0.001$$

Prior distribution for Turbo-VBI



$$\mathbf{y} = \mathbf{A}(\xi) \mathbf{x} + \mathbf{w}$$

$$p(\mathbf{x}, \rho, s | \phi) = \underbrace{p(s | \phi)}_{\text{Support}} \underbrace{p(\rho | s)}_{\text{Precision}} \underbrace{p(\mathbf{x} | \rho)}_{\text{Sparse signal}}$$



Prior distribution for Turbo-CS/AMP

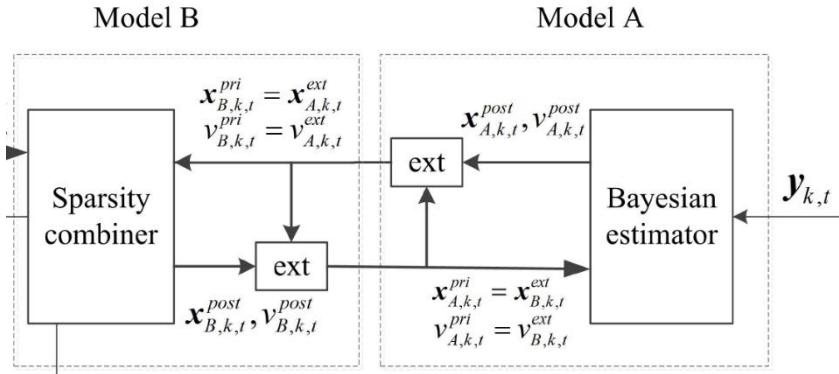


$$p(x, s|\phi) = \underbrace{p(s|\phi)}_{\text{Support}} \underbrace{p(x|s)}_{\text{Sparse signal}}$$

$$p(x_n|s_n) = s_n g_n(x_n) + (1 - s_n) \delta(x_n)$$

Usually Gaussian Distribution

Unified Approach: Turbo Compressive Sensing

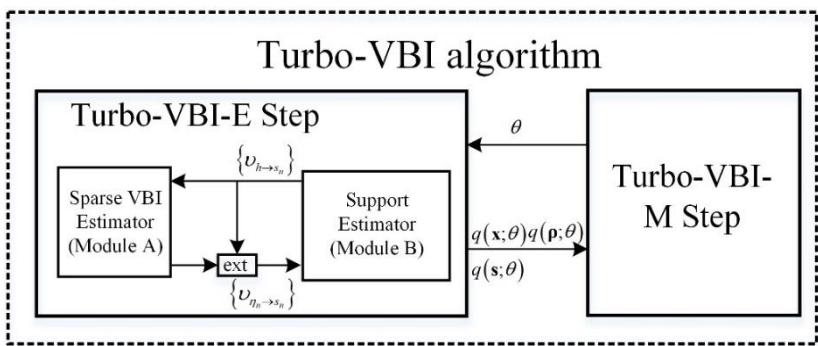


Module A:
LMMSE/VBI/AMP ...

Module B:
Message Passing ...

M-step:
In exact MM + AO +
gradient update

Unknown parameters
learned by M-step



How to choose proper methods for each step/module?
 How to properly partition the variables into hidden variables
 (Bayesian estimation in E-step) and unknown parameters (learned
 in the M-step)?



Extensions & Open Problems

- Extension to more general prior distributions
 - Often requires discrete approximation and sampling techniques
 - Stochastic optimization may play a role
- Extension to more general observation model
 - Generalized AMP (allow nonlinear models)
 - BigAMP for bilinear model
- Open problems
 - What if the factor graph in Module B has loops?
 - How to further reduce the complexity?
 - Mixed model-based and model-free approach?