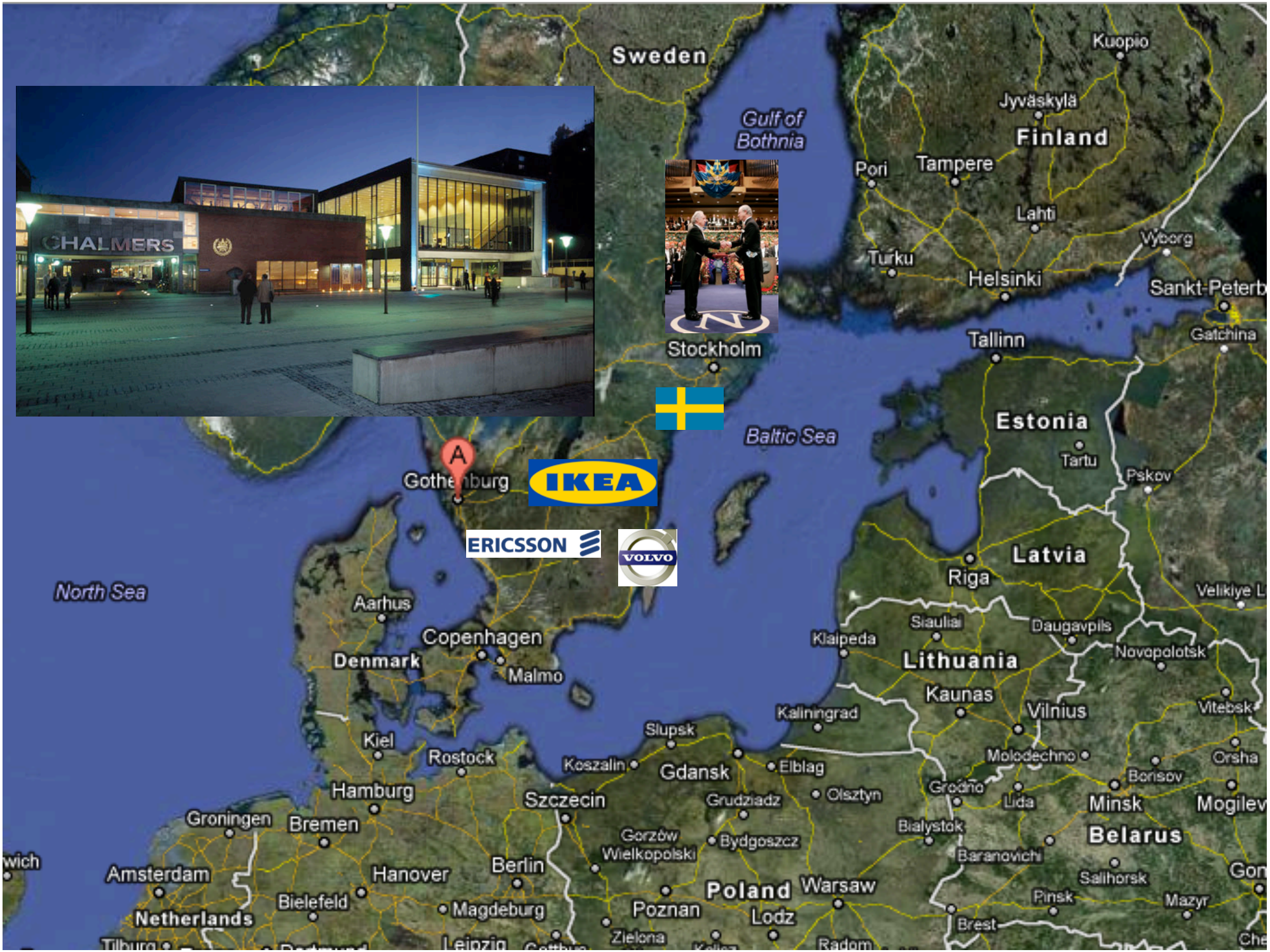
Network Processing with Bayesian Graphical Models

Henk Wymeersch
Department of Signals and Systems
Chalmers University of Technology
http://tinyurl.com/hwymeers
email: henkw@chalmers.se


CHALMERS
Sweden
Gulf of Bothnia
Kuopio
Jyväskylä
Finland
Pori
Tampere
Lahti
Turku
Helsinki
Vyborg
Sankt-Peterb
Gatchina
Tallinn
Stockholm
Estonia
Tartu
Pskov
Baltic Sea
A
Gothenburg
IKEA
ERICSSON
VOLVO
Latvia
Riga
North Sea
Aarhus
Copenhagen
Denmark
Malmo
Siauliai
Klaipeda
Daugavpils
Lithuania
Novopolotsk
Kaunas
Vilnius
Vitebsk
Kaliningrad
Slupsk
Kiel
Rostock
Koszalin
Gdansk
Elblag
Molodechno
Orsha
Hamburg
Szczecin
Grodno
Lida
Borisov
Minsk
Mogilev
Olsztyn
Grudziadz
Groningen
Bremen
Bialystok
Belarus
Gorzów Wielkopolski
Bydgoszcz
Baranovichi
Amsterdam
Hanover
Berlin
Salihorsk
Bielefeld
Poland
Warsaw
Pinsk
Mazyr
Netherlands
Magdeburg
Poznan
Lodz
Brest
Zielona
Radom

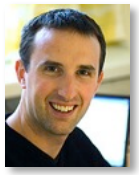
Communication System Group

Professor
Erik Ström
Synchronization, positioning, channel estimation, modulation

Professor
Erik Agrell
Optical communications, error-correcting coding, (coded) modulation theory

Professor
Thomas Eriksson
Data compression, hardware modeling

Associate Professor
Tommy Svensson
Coded modulation, medium access, cooperative communications



Assistant Professor
Fredrik Brännström
Error-control coding, modulation, iterative processing

Assistant Professor
Giuseppe Durisi
Network information theory, compressed sensing

Assistant Professor
Alexandre Graell i Amat
Coding, iterative processing, compressed sensing, cooperative communications

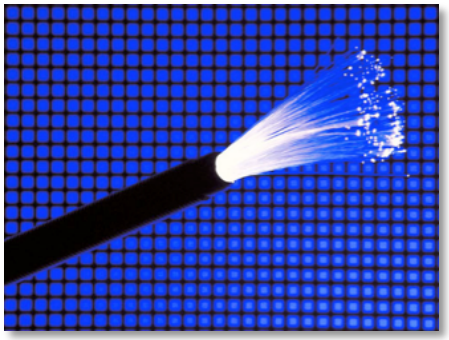
Associate Professor
Henk Wymeersch
Positioning, optics, Bayesian inference

Post Doc
Rocco di Taranto
Location-aware communications

Post Doc
Per Landin
Power amplifiers

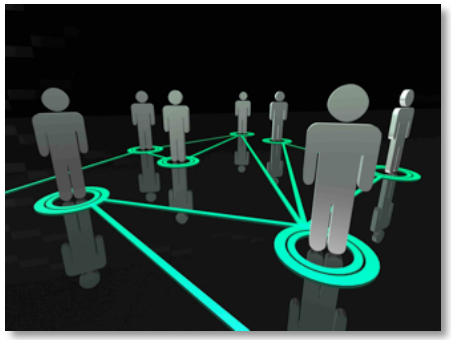
25 Ph.D. students

Our research

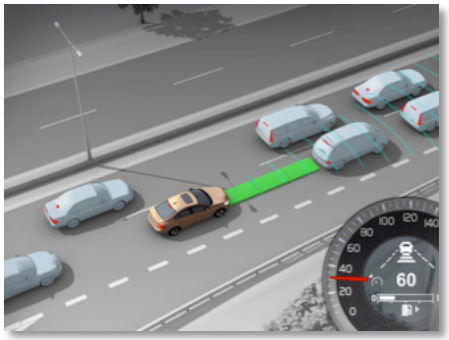


Fiber-optical communication

Wireless communication



Cooperative networks



Vehicular communication

Today's tutorial has applications in all these areas

Goals

At the end of this tutorial, you should be able to:

1. *Recognize and formulate* inference problems
2. *Solve* inference problems using factor graphs
3. *Design* algorithms for centralized and distributed processing
4. *Describe* the bigger picture

Do-it-yourself problems

- Some problems will not be solved in the tutorial
- Problems are shown by

- Most solutions are at the end of the tutorial

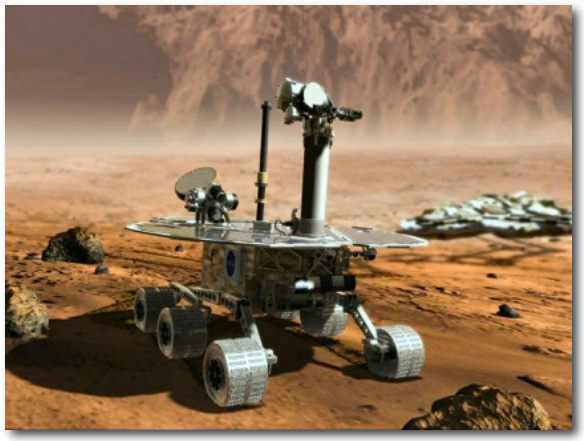
Outline

- Applications
- Background and terminology
- Bayesian detection
- **Tool 1: Bayesian graphical models**
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- **Tool 2: Belief consensus**
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

Outline

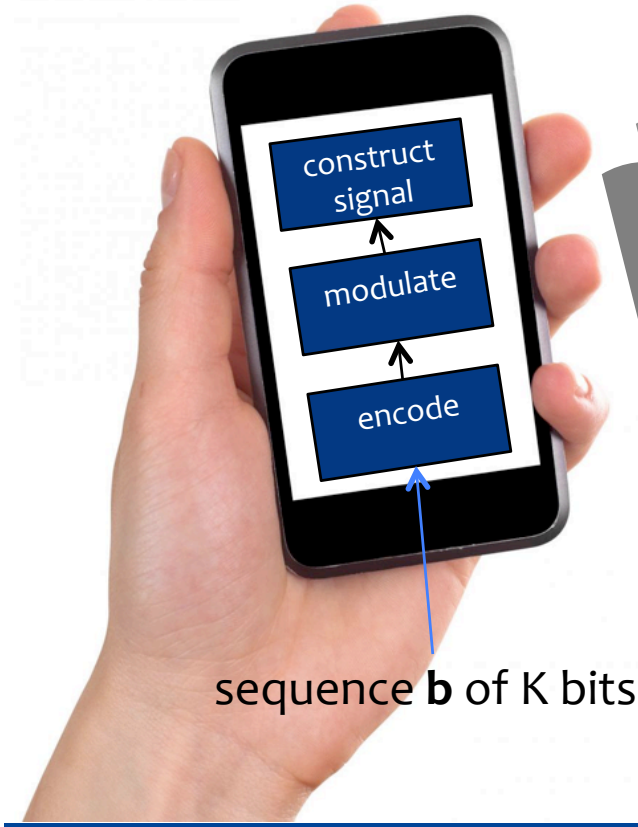
- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

Applications



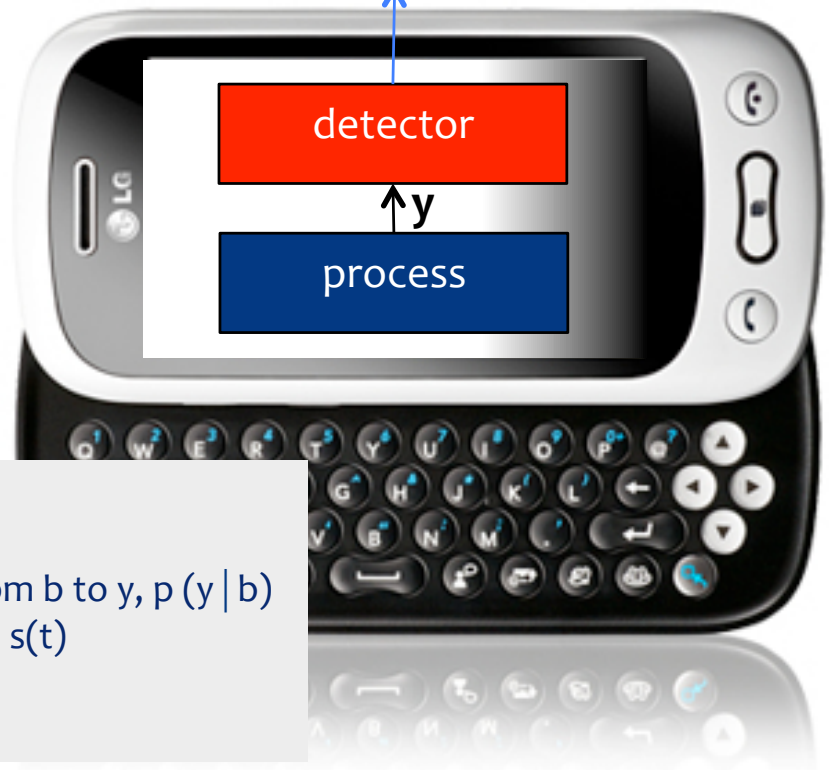
Application 1: receiver design

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



sequence **b** of K bits

estimate of **b**



The receiver knows
A priori distribution p(b)
Probabilistic mapping from b to y, p (y | b)

- Conversion from b to s(t)
- Channel (linear)
- Noise statistics

Application 2: cooperative localization

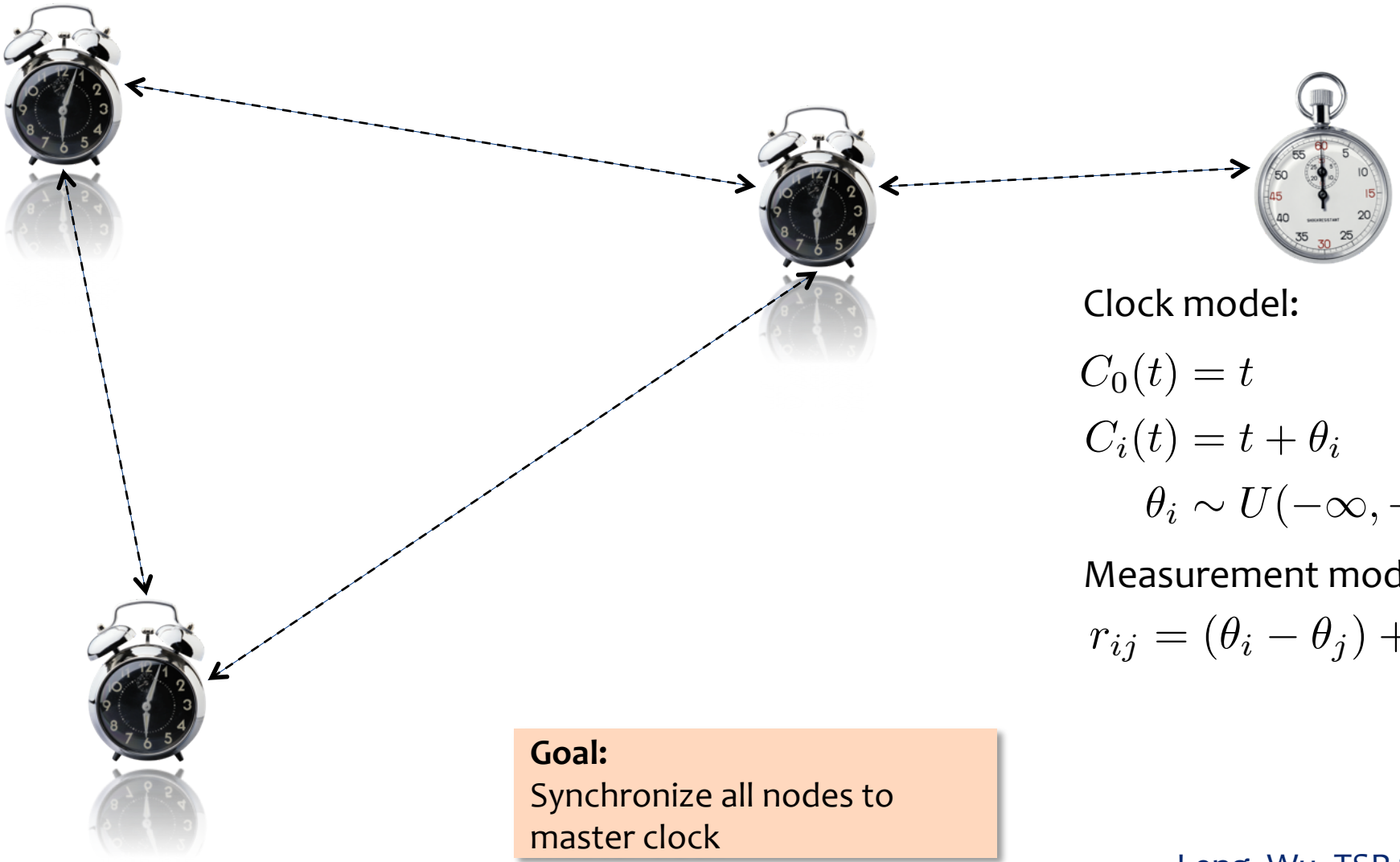
Scenario

- N mobile nodes (agents) with **unknown** positions $x_1(t),\ldots, x_N(t)$
- M reference nodes (anchors) with **known** positions $x_A(t)$, $x_B(t)$, $x_C(t)$, …
- **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

Application 3: network synchronization



Clock model:

$$C_0(t) = t$$

$$C_i(t) = t + \theta_i$$

$$\theta_i \sim U(-\infty, +\infty)$$

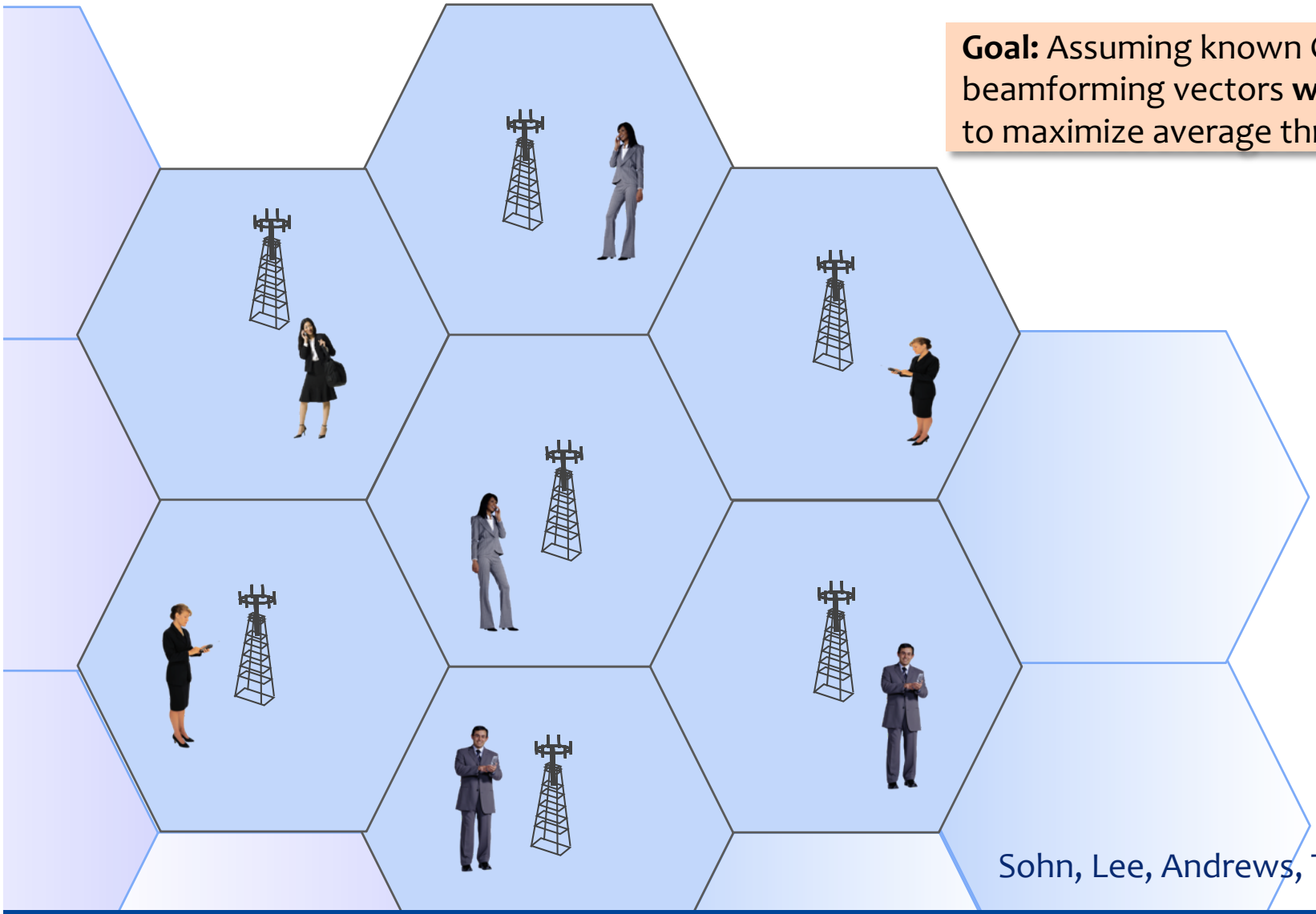
Measurement model:

$$r_{ij} = (\theta_i - \theta_j) + n_{ij}$$

Goal:
Synchronize all nodes to master clock

Leng, Wu, TSP 2011

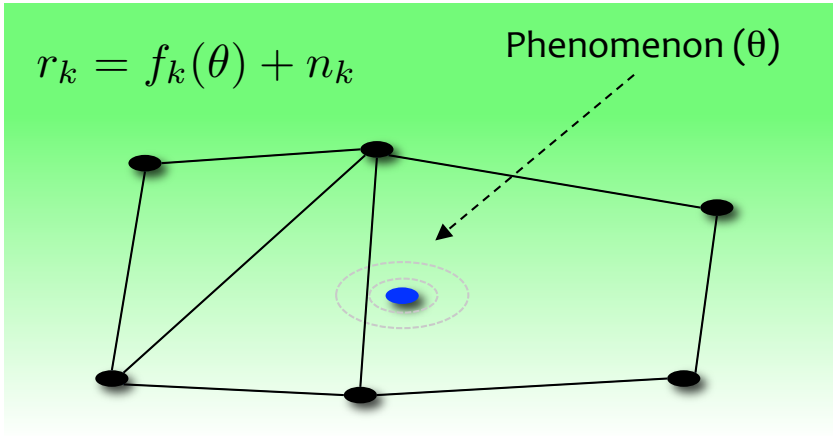
Application 4: distributed beamforming

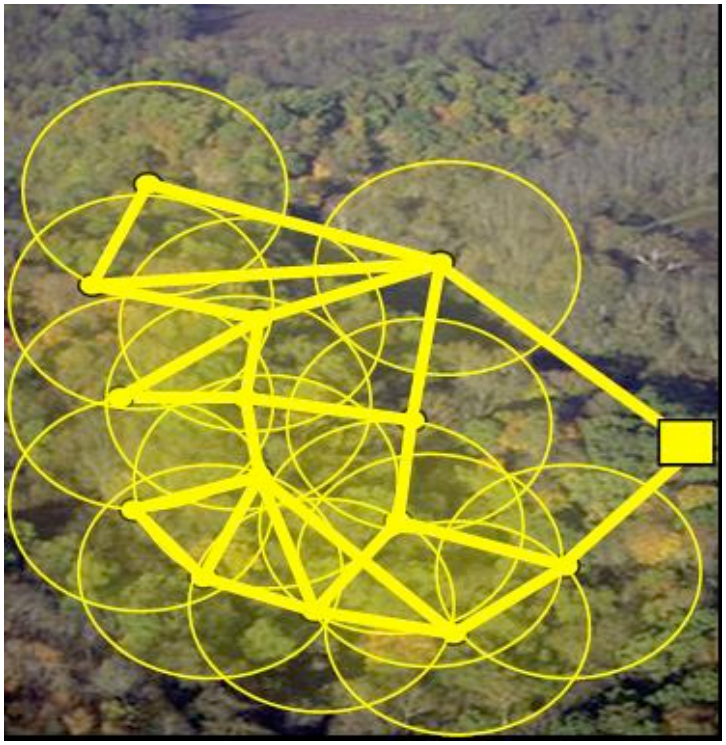


Goal: Assuming known CSI, design beamforming vectors $\mathbf{w}_j$ for every BS to maximize average throughput.

Sohn, Lee, Andrews, TCOM, 2011

Application 5: wireless sensor network





Goal:
Jointly compute estimate of θ

Outline

- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

Background information

Bayesian inference

- Bayesian inference

  Unknown **x**, observation **y**, model θ

$$p(\mathbf{x},\mathbf{y}|\theta) = p(\mathbf{x}|\mathbf{y},\theta)p(\mathbf{y}|\theta)$$
$$= p(\mathbf{y}|\mathbf{x},\theta)p(\mathbf{x}|\theta)$$

$$p(x_k|\mathbf{y},\theta) = \sum_{\text{not } x_k} p(\mathbf{x}|\mathbf{y},\theta)$$

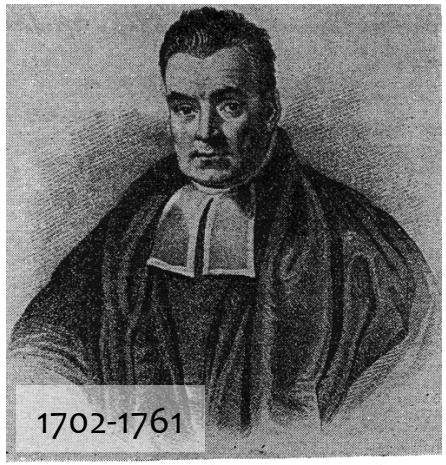
posterior of **x**

likelihood of θ

likelihood of **x** given θ

prior of **x**

marginal posterior of x_k



Rev. T. Bayes

Example: Nick is worried he is sick. He read in the newspaper 0.1% of the people is his city have contracted a deadly disease. He goes to the hospital to get tested. The doctor tells him the test is 95% reliable (i.e., it will give a correct result 95% of the time). The test is applied to Nick and turns out to be positive. With what probability does Nick have the disease?

The deadly disease

- We need to compute *p(Nick is sick|test is positive)*
- Applying Bayes' rule

$$p(S|P) = \frac{p(P|S) \times p(S)}{p(P)} \qquad p(H|P) = \frac{p(P|H) \times p(H)}{p(P)}$$

$$= \frac{0.95 \times 0.001}{p(P)} \qquad = \frac{0.05 \times 0.999}{p(P)}$$

$$= \frac{0.00095}{p(P)} \qquad = \frac{0.04995}{p(P)}$$

- So that

$$p(H|P) + p(S|P) = 1$$

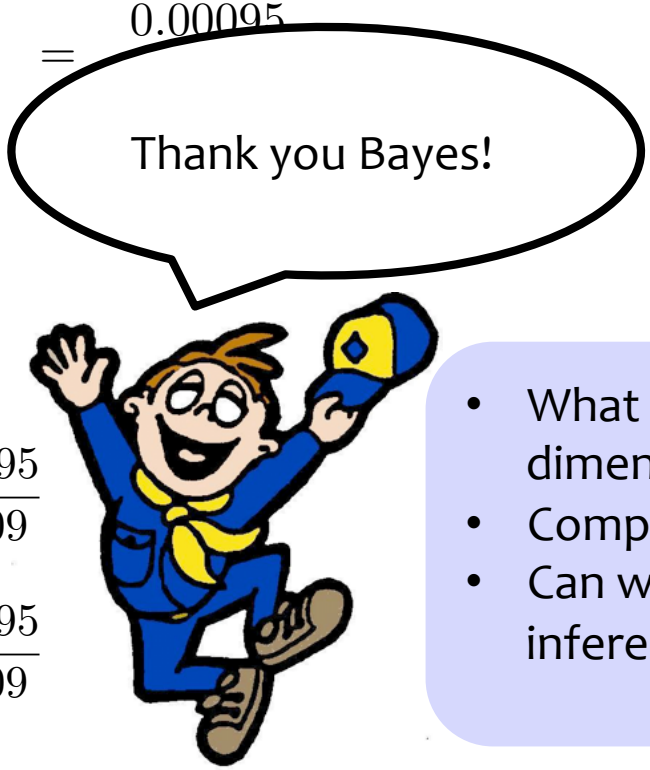
$$\frac{0.04995 + 0.00095}{p(P)} = 1$$

$$0.0509 = p(P)$$

$$\Rightarrow$$

$$P(S|P) = \frac{0.00095}{0.0509} = 0.02$$

$$P(H|P) = \frac{0.04995}{0.0509} = 0.98$$



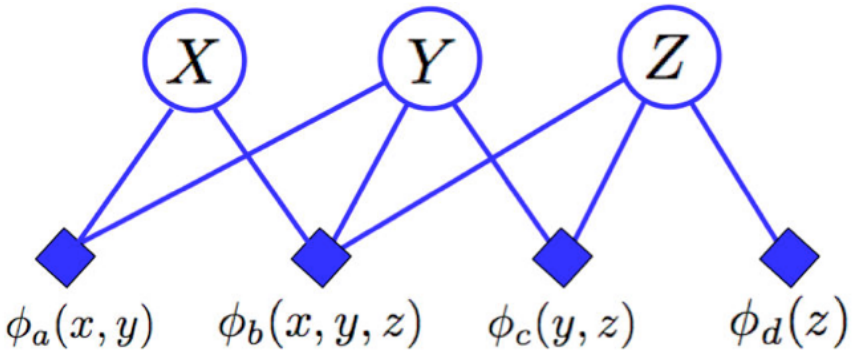
- What about high-dimensional problems?
- Complexity?
- Can we automate inference?

Graphical models

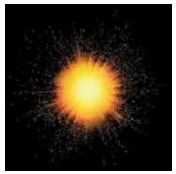
- Allow
  - Bayesian inference in rigorous, efficient, yet almost automated way
  - Modularity of problems and solutions
  - Graphical view of algorithms

- Applications
  - Every field that involves estimation/detecting something
  - Examples: communications, computer vision, bioinformatics

- New view of
  - Viterbi algorithm
  - BCJR algorithm
  - Turbo decoder
  - LDPC decoder
  - Kalman filter
  - Particle filter
  - ….



Graphical models: a biased history




Forney


Pearl


MacKay


Kschischang
Frey, Loeliger

1948 1963 1973 1981 1988 1993 1995 1996 2001 2005


Shannon


Gallager


Tanner


Berrou & Glavieux


Wiberg


Yedidia
Freeman
Weiss

Capacity
LDPC
trellis
Tanner graph
BP
turbo
LDPC
FG and SPA
decoding on graphs
GBP

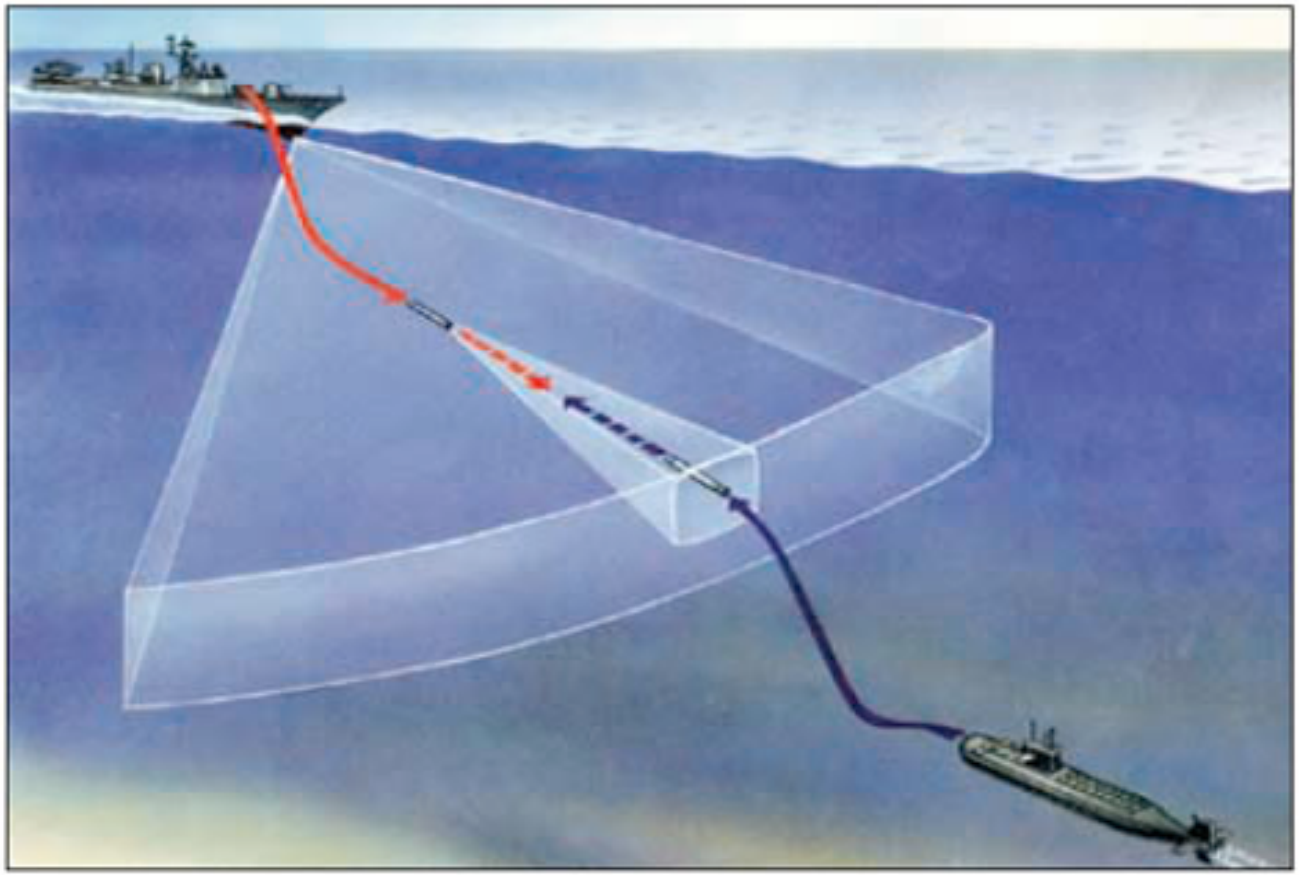
Background material

- Papers
  - "Factor graphs and the sum-product algorithm," Kschischang, Frey, Loeliger, *IEEE Transaction in Information Theory*, 2001
  - "Constructing free energy approximations and generalized belief propagation algorithms," Yedidia, Freeman, Weiss, *IEEE Transactions on Information Theory*, 2005
  - Martin J. Wainwright and Michael I. Jordan (2008) "Graphical Models, Exponential Families, and Variational Inference", *Foundations and Trends in Machine Learning*: Vol. 1: No 1–2, pp 1-305.
- Books
  - *Probabilistic Graphical Models: Principles and Techniques*, Koller and Friedman, MIT press, 2009
  - *Graphical models for machine learning and digital communication*, Frey, MIT press, 1998
  - *Information theory, inference and learning algorithms*, MacKay, Cambridge University Press, 2003
  - *Wireless communication systems: advanced techniques for signal reception*, Wang, Poor, Prentice Hall, 2003
  - *Iterative Receiver Design*, Wymeersch, Cambridge University Press, 2007

Outline

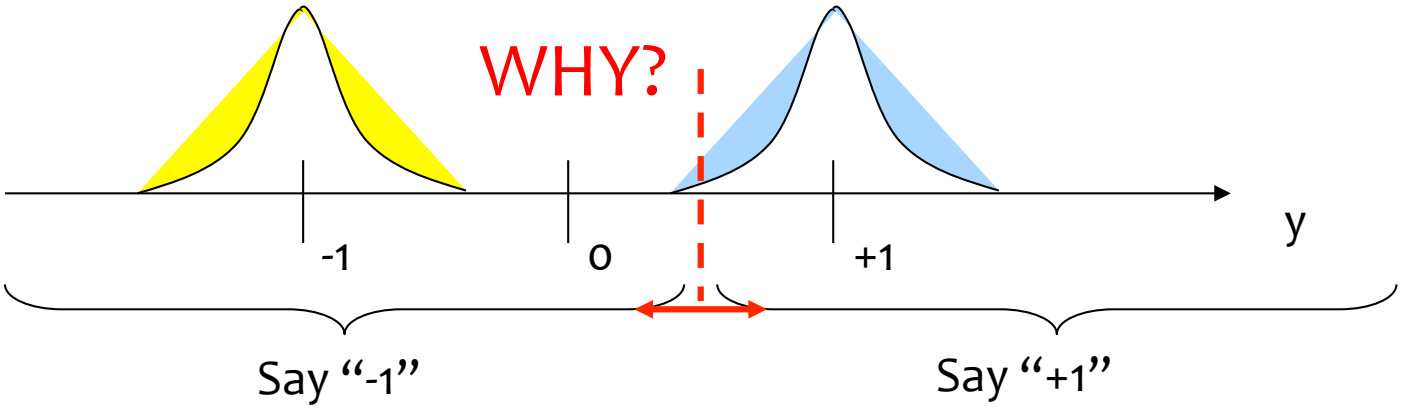
- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

Detection Theory



Optimal detection

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

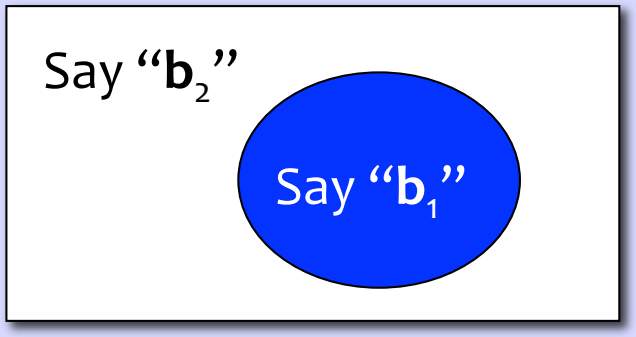


Optimal detection

- For every possible **y**, find best $\mathbf{b}^*(\mathbf{y})$
- What is "best"?
  - Correct: $\mathbf{b}^*(\mathbf{y})=\mathbf{b}$ has cost $C_1 \geq 0$
  - Wrong: $\mathbf{b}^*(\mathbf{y})=\mathbf{b}$ has cost $C_2 > C_1$
  - General: cost function $c(\boldsymbol{b},\boldsymbol{b}^*(\boldsymbol{y}))$: $\mathcal{B}\times\mathcal{B}\rightarrow R$
  - Minimize expected cost (why not max cost?)

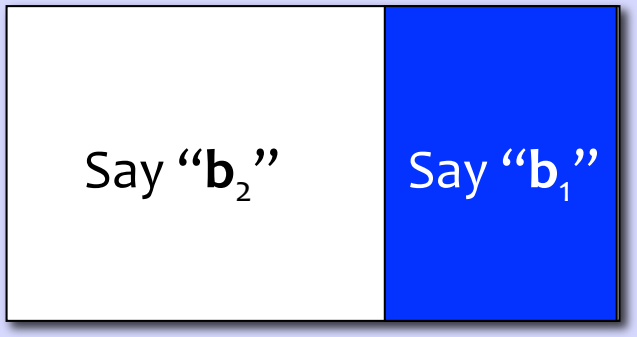
For a given cost function $c(\boldsymbol{b},\boldsymbol{b}^*(\boldsymbol{y}))$

$\mathcal{Y}$, set of possible **y**



⇒ Expected cost C

$\mathcal{Y}$, set of possible **y**



⇒ Expected cost C'

Optimal detection

- Expected cost for given detector **b*(y)**

$$\begin{aligned}\mathcal{C} &= \mathbb{E}_{\mathbf{B},\mathbf{Y}}\left\{c(\mathbf{B},\mathbf{b}^*(\mathbf{Y}))\right\} \\ &= \sum_{\mathbf{b},\mathbf{y}} p_{\mathbf{B},\mathbf{Y}}(\mathbf{b},\mathbf{y})c(\mathbf{b},\mathbf{b}^*(\mathbf{y})) \\ &= \sum_{\mathbf{y}} p_{\mathbf{Y}}(\mathbf{y}) \underbrace{\left(\sum_{\mathbf{b}} p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}|\mathbf{y})c(\mathbf{b},\mathbf{b}^*(\mathbf{y}))\right)}_{\mathcal{C}(\mathbf{y})}\end{aligned}$$

<Bayes' Rule: p(a,b)=p(a)p(b|a)>

Minimize this w.r.t. **b*(y)**

MAP detection

- Special case: cost 1 when incorrect decision, cost 0 otherwise

$$\begin{aligned} \mathcal{C}(\mathbf{y}) &= \sum_{\mathbf{b}} p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}|\mathbf{y}) c(\mathbf{b}, \mathbf{b}^*(\mathbf{y})) \\ &= \sum_{\mathbf{b} \neq \mathbf{b}^*(\mathbf{y})} p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}|\mathbf{y}) \\ &= 1 - p_{\mathbf{B}|\mathbf{Y}}(\mathbf{b}^*(\mathbf{y})|\mathbf{y}) \end{aligned}$$

- Leads to the maximum a posteriori (MAP) rule

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

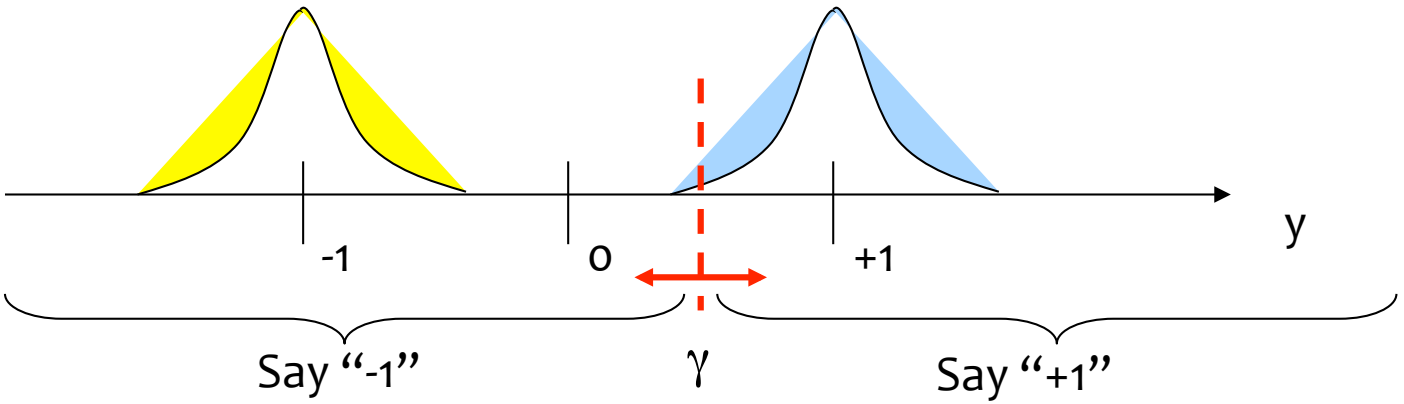
MAP detection - example

- $b \in \{-1,+1\}$, $p_B(1)=p_1$, $p_B(-1)=1-p_1$
- $y=b+n$, $n \sim N_n(0,1) \Rightarrow p_{Y|B}(y|b) = N_y(b,1)$

$$
\begin{aligned}
b^*(y) &= \arg\max_b p_{B|Y}(b|y) \\
&= \arg\max_b p_{Y|B}(y|b)p_B(b) \\
&= \arg\max_b \ln\left(p_{Y|B}(y|b)p_B(b)\right) \\
&= \arg\max_b -\frac{|y-b|^2}{2} + \ln p_B(b) \\
&= \arg\max_b yb + \ln p_B(b)
\end{aligned}
$$

Say "+1"
$\Leftrightarrow y+\ln(p_1) > -y+\ln(1-p_1)$
$\Leftrightarrow y > 0.5\ln((1-p_1)/p_1)$
$\Leftrightarrow y > \gamma$

Note:
$\gamma=0$ when $p_1=0.5$
$\gamma>0$ when $p_1<0.5$
$\gamma<0$ when $p_1>0.5$



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 **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 **b**: complexity exponential in length of **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 $p(b_k \mid \mathbf{y})$

Solution

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

‡Computing $p(\mathbf{b}|\mathbf{y})$ (up to constant) for given **y** and **b** is easy

How about continuous variables?

- Suppose $\mathbf{b} \in C^N$
- Terminology: "detection" → "estimation"
- Different cost functions lead to different estimators

- MAP estimator (for $\delta \to 0$)

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

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

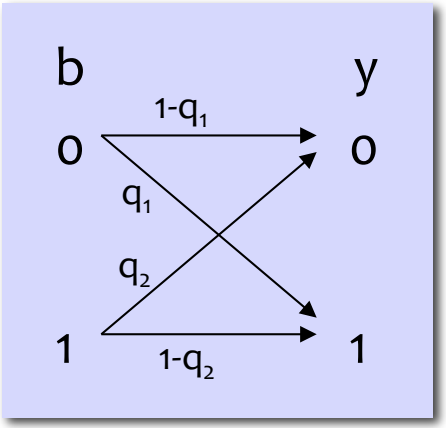
- MMSE estimator

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

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

Two examples

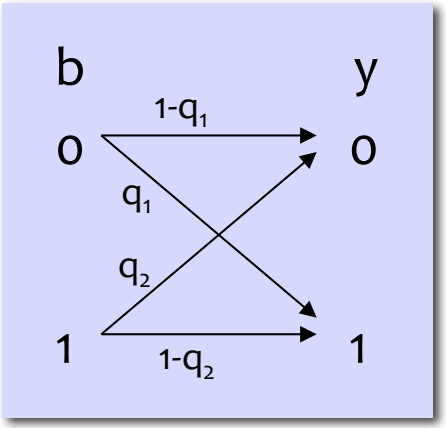
- Detection
  - Repetition code: **b**=[00], or **b**=[11], b a priori uniform
  - Binary asymmetrical channel
  - We receive **y**=[01]
  - Find MAP estimate of b



- Estimation
  - Coin with bias q, $p_Q(q) = U(1/2-\delta,1/2+\delta)$
  - Toss N times, observe **y** = sequence of heads and tails, with N_H heads
  - Find MAP estimate of q (prob of heads)

Example: detection

- Repetition code: **b**=[00], or **b**=[11], b a priori uniform
- Binary asymmetrical channel
- **y**=[01]
- Find MAP estimate of b



$$
\begin{aligned}
b^*([0,1]) &= \arg\max_b p_{B|\mathbf{Y}}(b|0,1) \\
&= \arg\max_b p_{\mathbf{Y}|B}(0,1|b) \\
&= \arg\max_b p_{\mathbf{Y}|B_1,B_2}(0,1|b,b) \\
&= \arg\max_b \underbrace{p_{Y_1|B_1}(0|b)p_{Y_2|B_2}(1|b)}_{\phi(b)}
\end{aligned}
$$

$\Rightarrow \phi(0)=(1-q_1)q_1$ and $\phi(1)=(1-q_2)q_2$

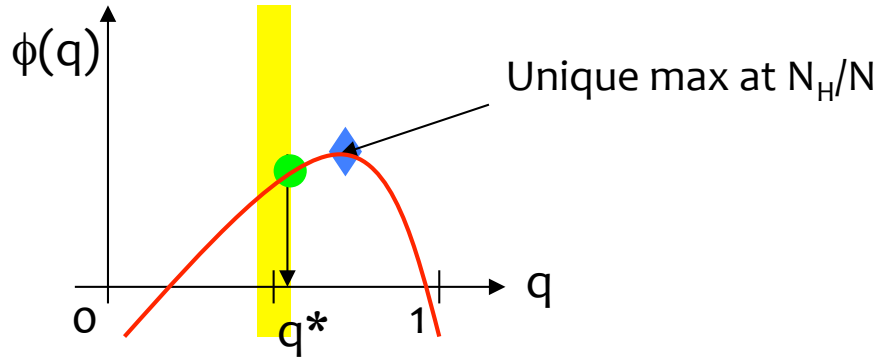
Example:

- Suppose $q_1=0.01$, $q_2=0.05$, then $\phi(0)=0.0099$ and $\phi(1)=0.0475$
- Decide: "1"

Example: estimation

- Coin with bias q, $p_Q(q) = U(1/2-\delta, 1/2+\delta)$
- Toss N times, observe **y** = sequence of heads and tails, with N_H heads
- Find MAP estimate of q (prob of heads)

$$
\begin{aligned}
q^*(\mathbf{y}) &= \arg\max_{q\in[0,1]} p_{Q|\mathbf{Y}}(q|\mathbf{y}) \\
&= \arg\max_{q\in[0,1]} p_{\mathbf{Y}|Q}(\mathbf{y}|q)p_Q(q) \\
&= \arg\max_{q\in[1/2-\delta,1/2+\delta]} q^{N_\mathrm{H}}(1-q)^{(N-N_\mathrm{H})} \\
&= \arg\max_{q\in[1/2-\delta,1/2+\delta]} \underbrace{N_\mathrm{H}\ln(q) + (N-N_\mathrm{H})\ln(1-q)}_{\phi(q)}
\end{aligned}
$$



Outline

- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

Factor graphs and message passing

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 y
    $$p(\mathbf{y})$$

- Complexity exponential in dimension of 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

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

- Solve the three problems

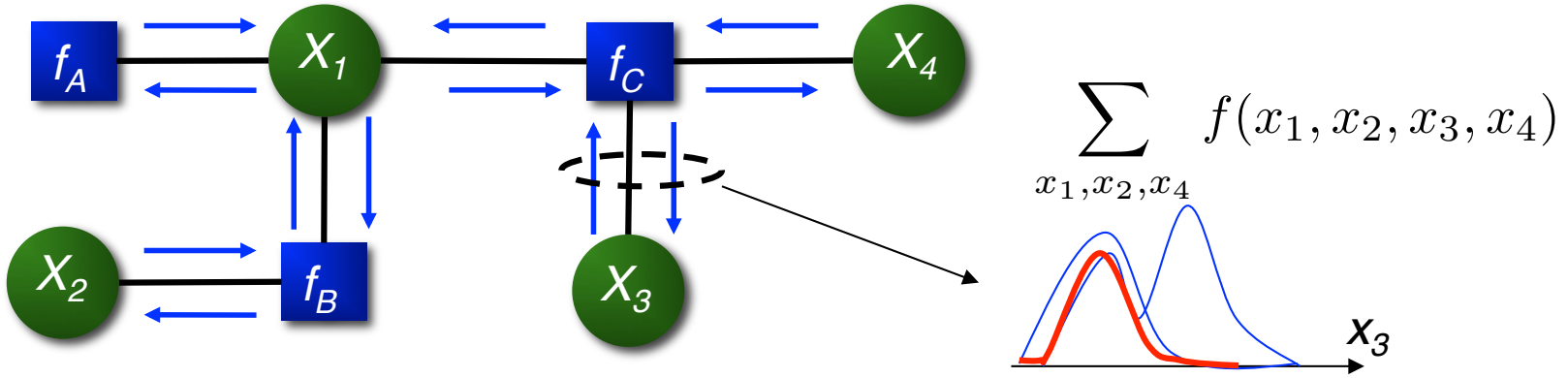
$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x}, \mathbf{y})$$

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

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

Factor graphs: high level

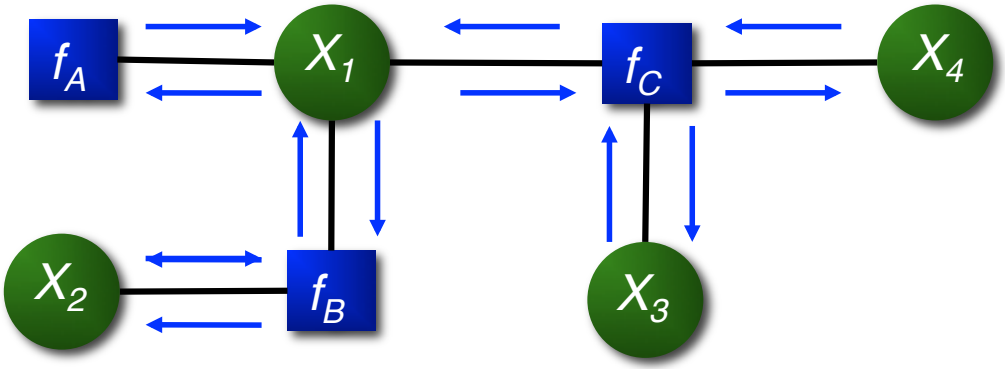
- Factorization $f(x_1, x_2, x_3, x_4) = f_A(x_1) f_B(x_1, x_2) f_C(x_1, x_3, x_4) \geq 0$

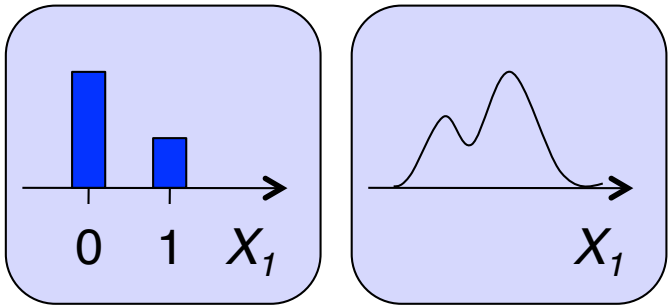


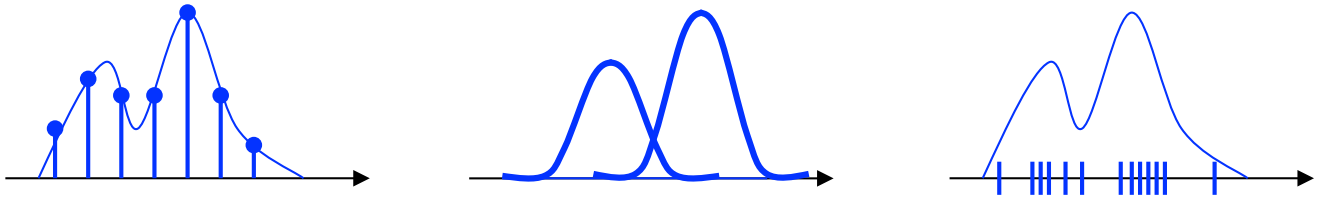
- The sum-product algorithm
  - Variable vertex to factor vertex
    $$\mu_{X_1 \to f_C}(x_1) = \mu_{f_A \to X_1}(x_1) \times \mu_{f_B \to X_1}(x_1)$$
  - Factor vertex to variable vertex
    $$\mu_{f_C \to X_1}(x_1) = \sum_{x_3, x_4} f_C(x_1, x_3, x_4) \times \mu_{X_3 \to f_C}(x_3) \times \mu_{X_4 \to f_C}(x_4)$$
- Marginals
    $$g_{X_3}(x_3) = \mu_{f_C \to X_3}(x_3) \times \mu_{X_3 \to f_C}(x_3)$$
    $$= \sum_{x_1, x_2, x_4} f(x_1, x_2, x_3, x_4)$$

Factor graphs: high level

- Messages are functions



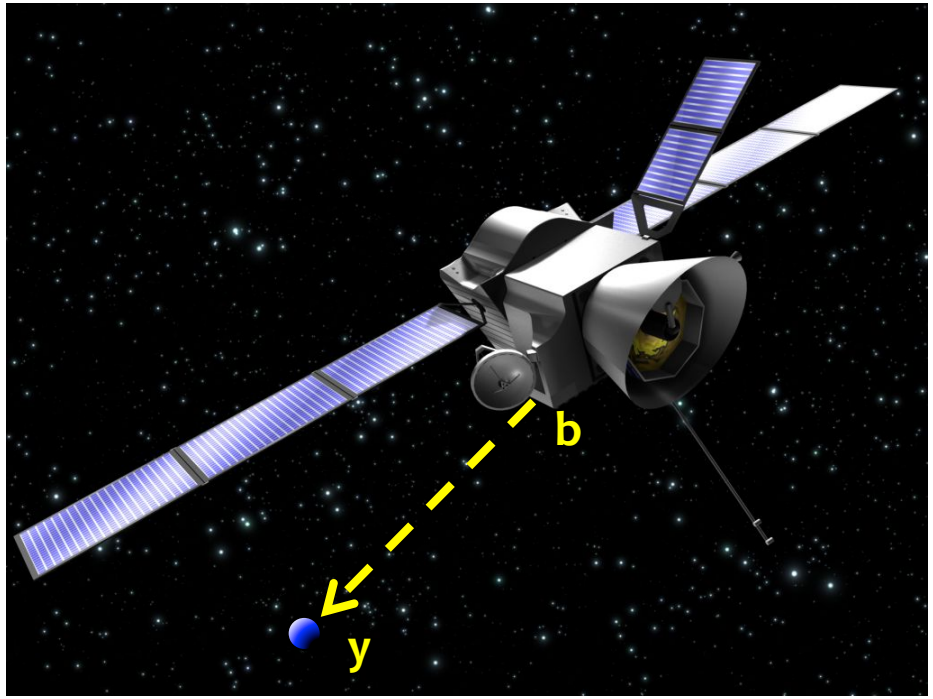


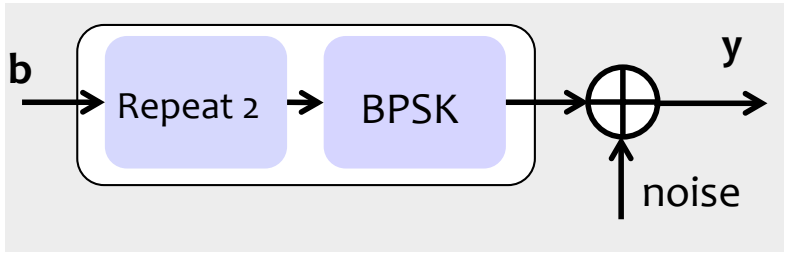


- Messages are often normalized and transformed (e.g., to LLR)
- When factor graph has cycles, SPA may fail
- Replacing "sum" with "max" yields max-marginals (MPA)

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

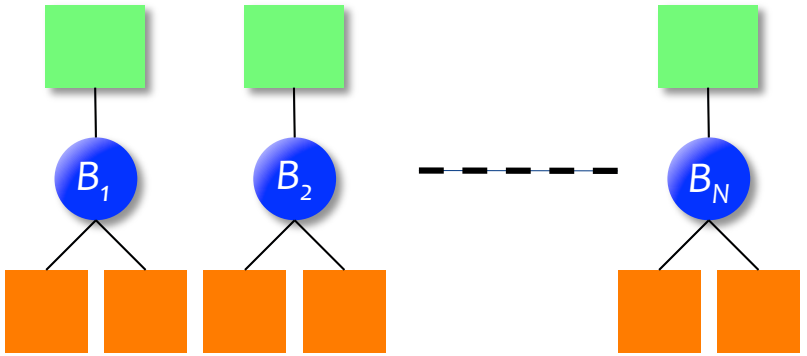
Factor graphs: high level





$$p(\mathbf{b}|\mathbf{y}) \propto p(\mathbf{b})p(\mathbf{y}|\mathbf{b})$$

$$= \prod_{k=1}^{N} p(b_k)p(y_{2k-1}|b_k)p(y_{2k}|b_k)$$



Optimal sequence detection:

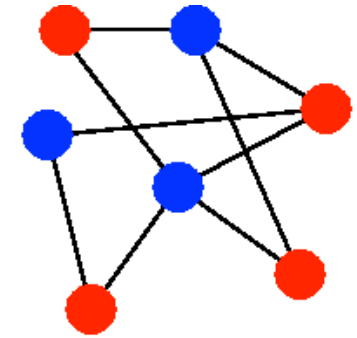
$$\hat{\mathbf{b}} = \arg\max_{\mathbf{b}\in\mathbb{B}^N} p(\mathbf{b}|\mathbf{y})$$

Optimal bit-by-bit detection:

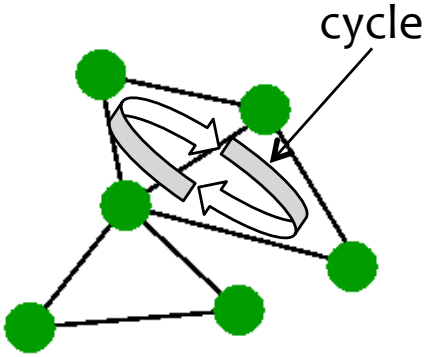
$$\hat{b}_k = \arg\max_{b_k\in\mathbb{B}} p(b_k|\mathbf{y})$$

What is a factor graph?

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



A) A Bipartite Graph



B) A non–Bipartite Graph

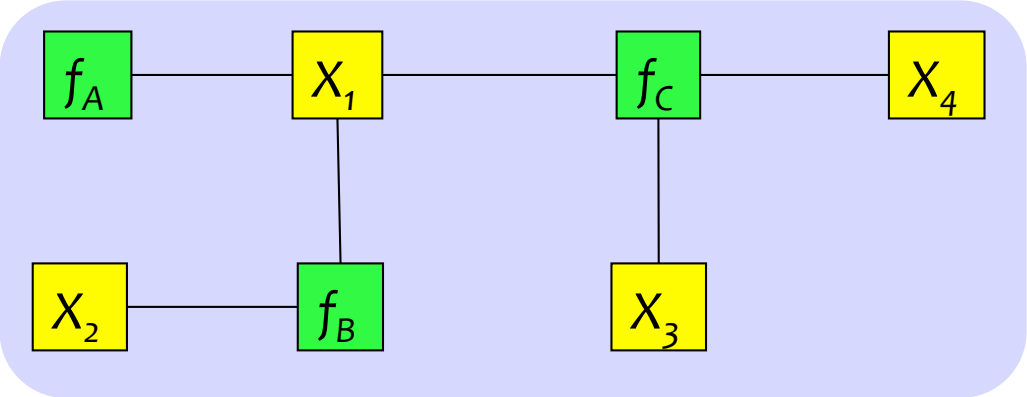
- Factor graph

A non-unique graph that represents the factorization of a real-valued function f(**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

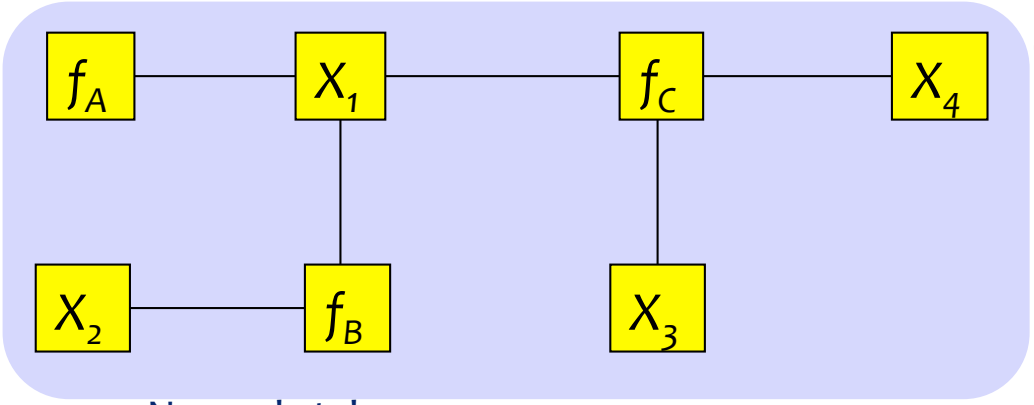


Construction

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

Conventional vs normal FG

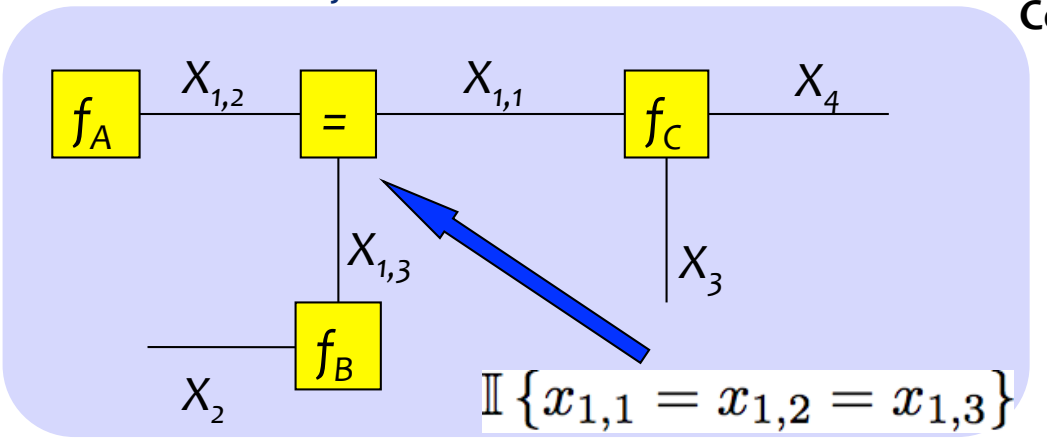
- Factorization $f(x_1, x_2, x_3, x_4) = f_A(x_1) f_B(x_1, x_2) f_C(x_1, x_3, x_4)$
- Conventional style



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

- Normal style



Construction

1. Vertex for every factor f_k (labeled with f_k)
2. An edge for every variable x_i connecting the factors in which it appears
3. x_i appears in 1 factor f_k: half-edge connected to vertex f_k, label X_i
4. x_i appears in 2 factors f_k,f_l: edge between f_k and f_l, label X_i
5. x_i appears in M>2 factors: a vertex (label "="), M edges (labeled $X_{i,1}$,…,$X_{i,M}$), connected to the M factors

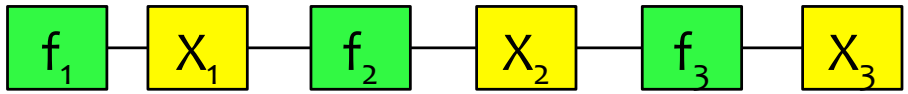
Indicator function: 1 when argument is true, 0 otherwise

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

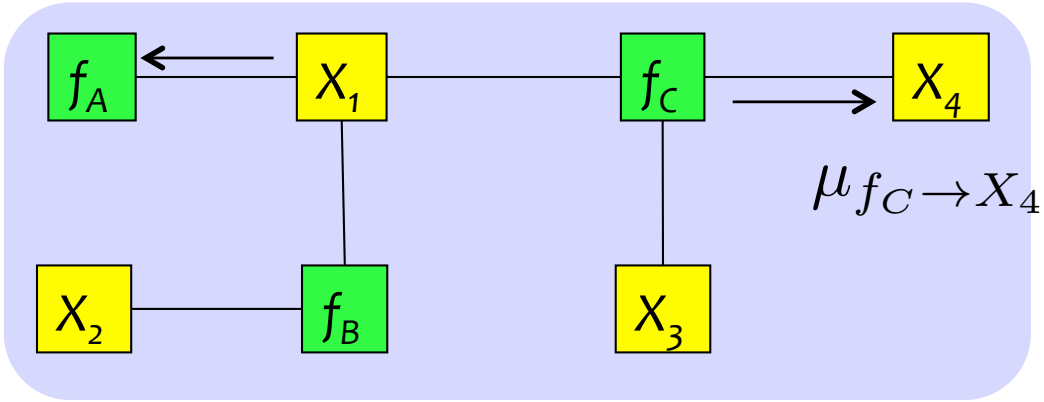


3. Can you come up with a reasonable distribution?

Message passing on factor graphs

- Messages are functions

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



- If X_4 is binary and X_1 ternary

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

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

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(**x**)

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

- Max-marginal of a function f(**x**)

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

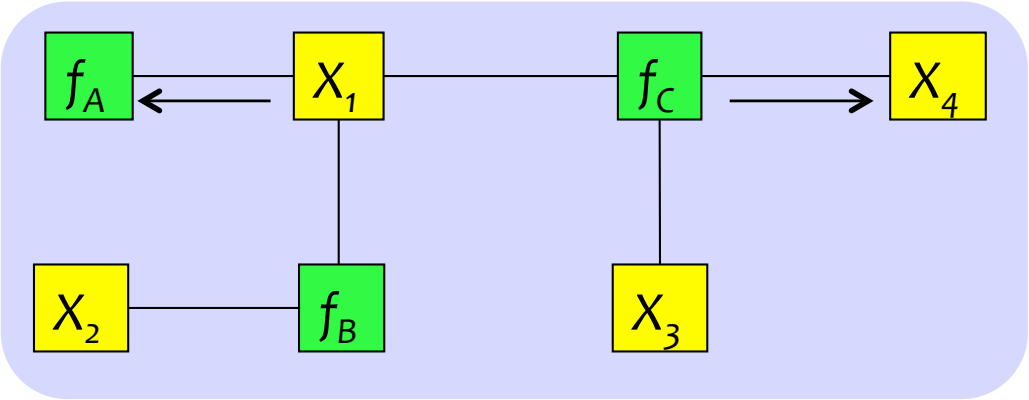
When $f(\mathbf{x}) = p(\mathbf{x}, \mathbf{y})$
Then

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

and

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

The sum-product algorithm (=belief propagation)



1. Message from variable to factor vertex

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

2. Message from factor to variable vertex

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

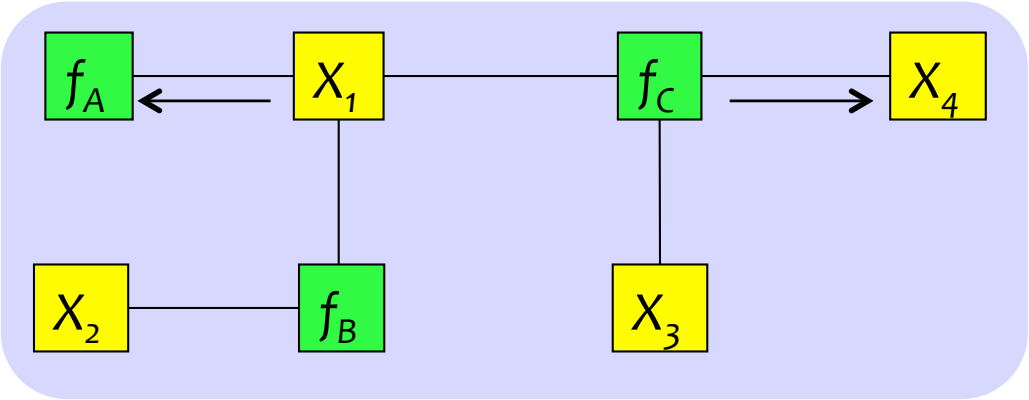
3. Sum-marginal

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

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

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

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

2. Message from factor to variable vertex

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

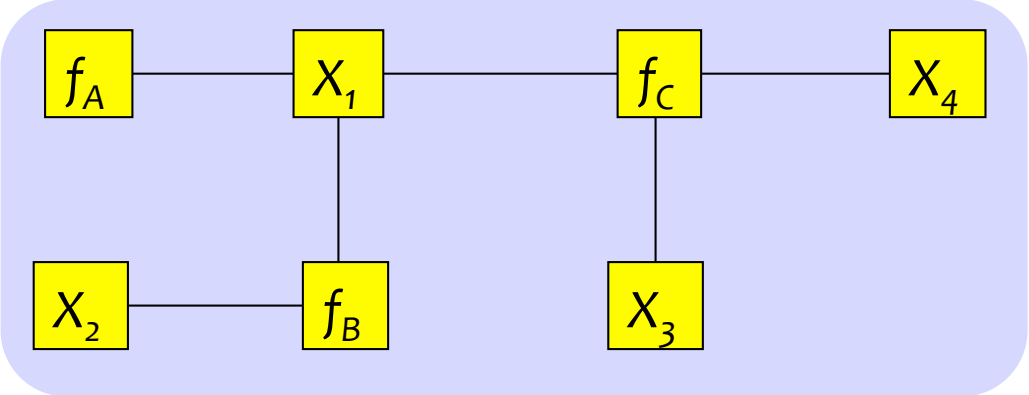
3. Max-marginal

$$g_{X_1}^{\max}(x_1) = \mu_{f_A \to X_1}(x_1) \times \mu_{X_1 \to f_A}(x_1)$$
$$g_{X_4}^{\max}(x_4) = \mu_{f_C \to X_4}(x_4) \times \mu_{X_4 \to f_C}(x_4)$$

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: $[\mu(0),\mu(1)]$



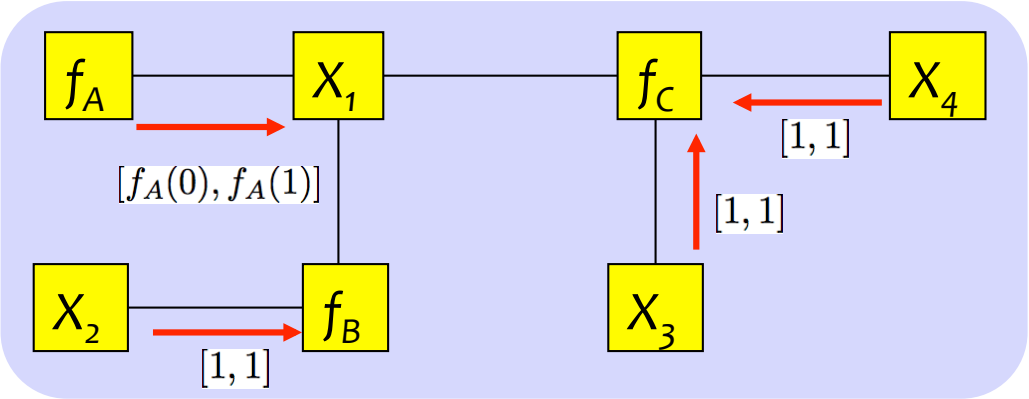
3 phases

1. Initialization
2. Message computation
3. Termination

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

Phase 1: initialization

- Messages start from the edge of the graph
  - Variable vertices of degree 1 send the message "1" over the domain: **m**=[1,1]
  - Factor vertices of degree 1 (say $f_k(x_i)$) send message: **m**=[$f_k(0)$,$f_k(1)$]



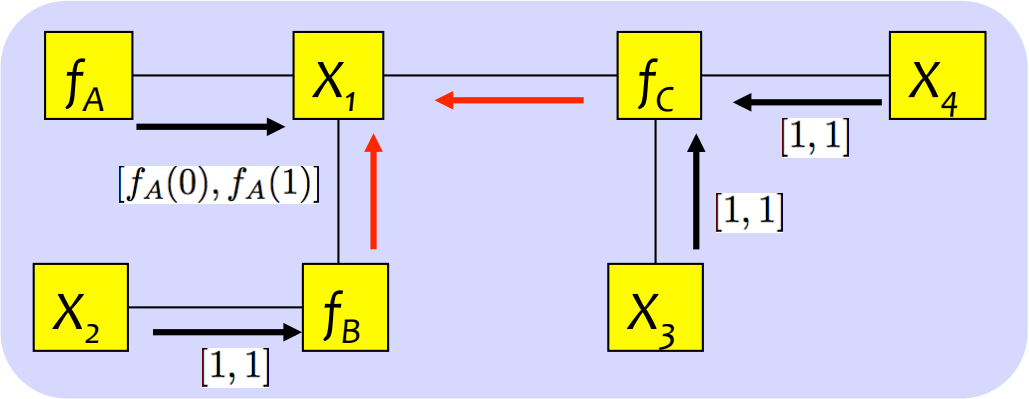
3 phases

1. Initialization
2. Message computation
3. Termination

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

Phase 2: message computation

- Compute outgoing message when all other incoming messages are available



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

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

$$= \sum_{x_3, x_4 \in \{0,1\}} f_C(x_1, x_3, x_4)$$

---

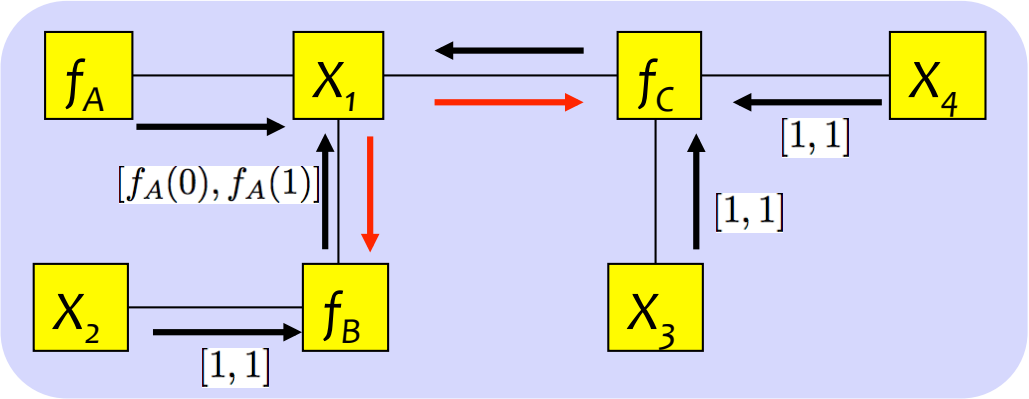
3 phases

1. Initialization
2. Message computation
3. Termination

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

Phase 2: message computation

- Compute outgoing message when all other incoming messages are available



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

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

---

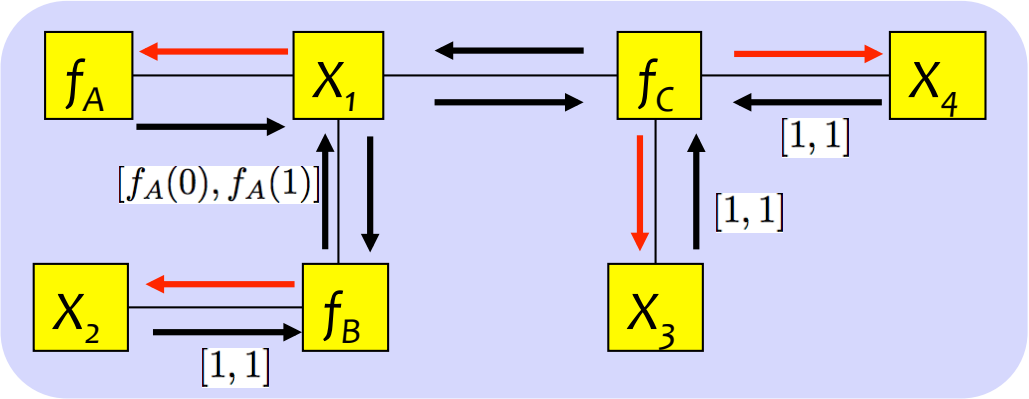
3 phases

1. Initialization
2. Message computation
3. Termination

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

Phase 2: message computation

- Compute outgoing message when all other incoming messages are available



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

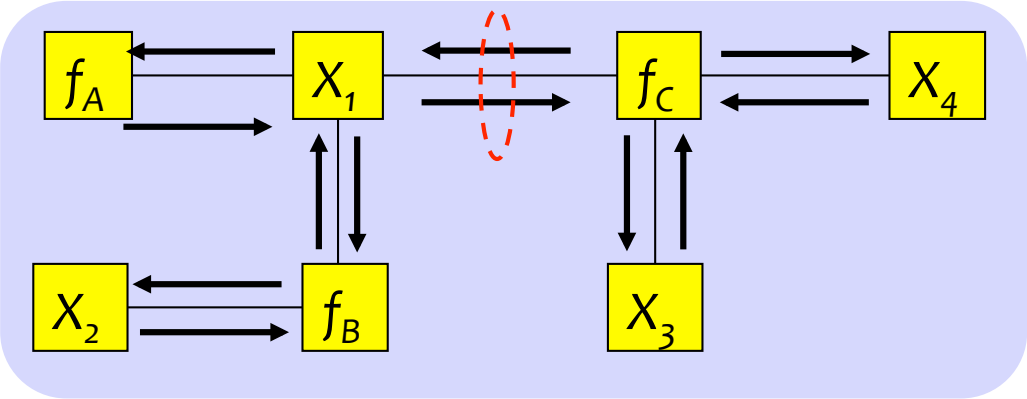
3 phases

1. Initialization
2. Message computation
3. Termination

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

Phase 3: termination

- Compute outgoing message when all other incoming messages are available



$$\mu_{X_1 \to f_C}(x_1)\mu_{f_C \to X_1}(x_1) = \sum_{x_2,x_3,x_4} f(x_1,x_2,x_3,x_4)$$
$$= \mu_{X_1 \to f_A}(x_1)\mu_{f_A \to X_1}(x_1)$$

---

3 phases

1. Initialization
2. Message computation
3. Termination

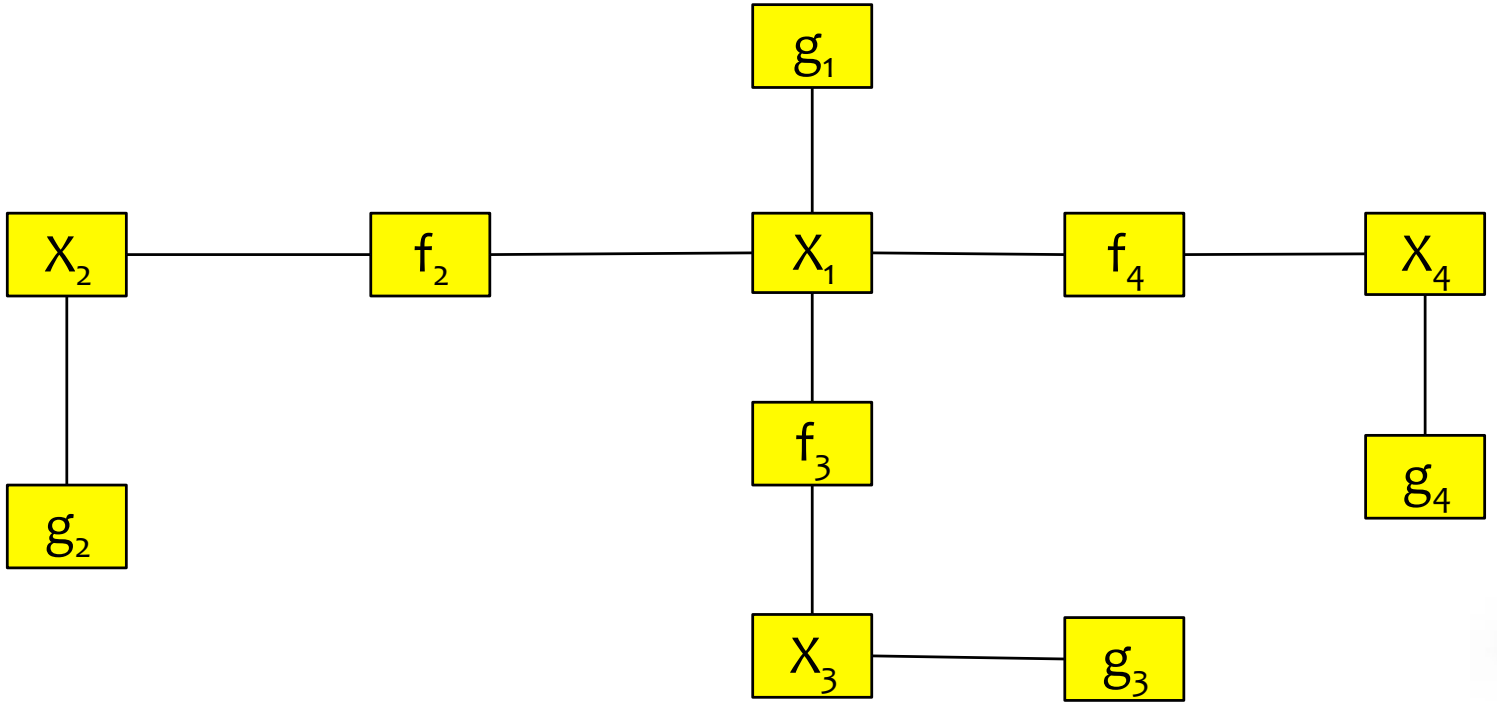
$$g_{X_i}(x_i) = \mu_{f_k \to X_i}(x_i) \times \mu_{X_i \to f_k}(x_i)$$

Example

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

  $g_i(x_i)$ $f_i(x_1, x_i)$

- Write down the sum-product rules
- Verify the solution



Sum-product and max-product algorithm

- Sum-product algorithm
  - Aim: compute (sum-) marginals $g_{X_i}(x_i) = \sum_{\sim\{x_i\}} f(\mathbf{x})$
  - Method: distributive law for real numbers
    $a(b+c) = ab + ac$

- Max-product algorithm
  - Aim: compute max-marginals $g_{X_i}(x_i) = \max_{\sim\{x_i\}} f(\mathbf{x})$
  - Method: distributive law for non-negative real numbers
    $a\max(b,c) = \max(ab, ac); a, b, c \geq 0$

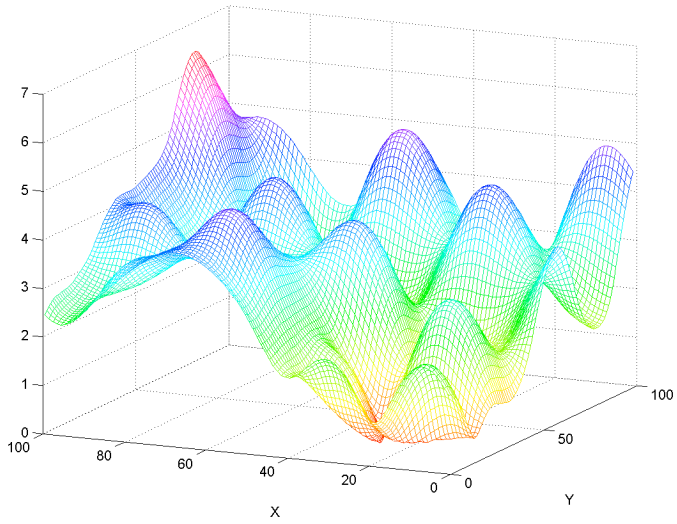
- Log-domain versions (see later)
  - Max*-sum
  - Max-sum

- Generalization: any suitable (F,⊕,⊗)

Sum-product and max-product algorithm

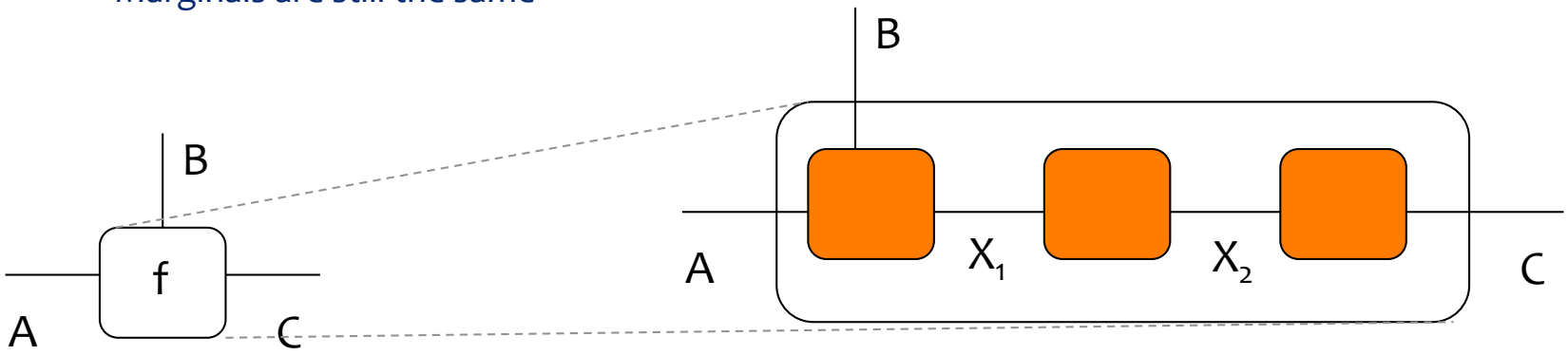
- Message passing rules
  - Replace all sums in SPA by max
  - Outcome is max-marginals

- Usage
  - assume f(**x**) has global maximum: f(**x***)
  - Concatenation of modes of max-marginals = **x***



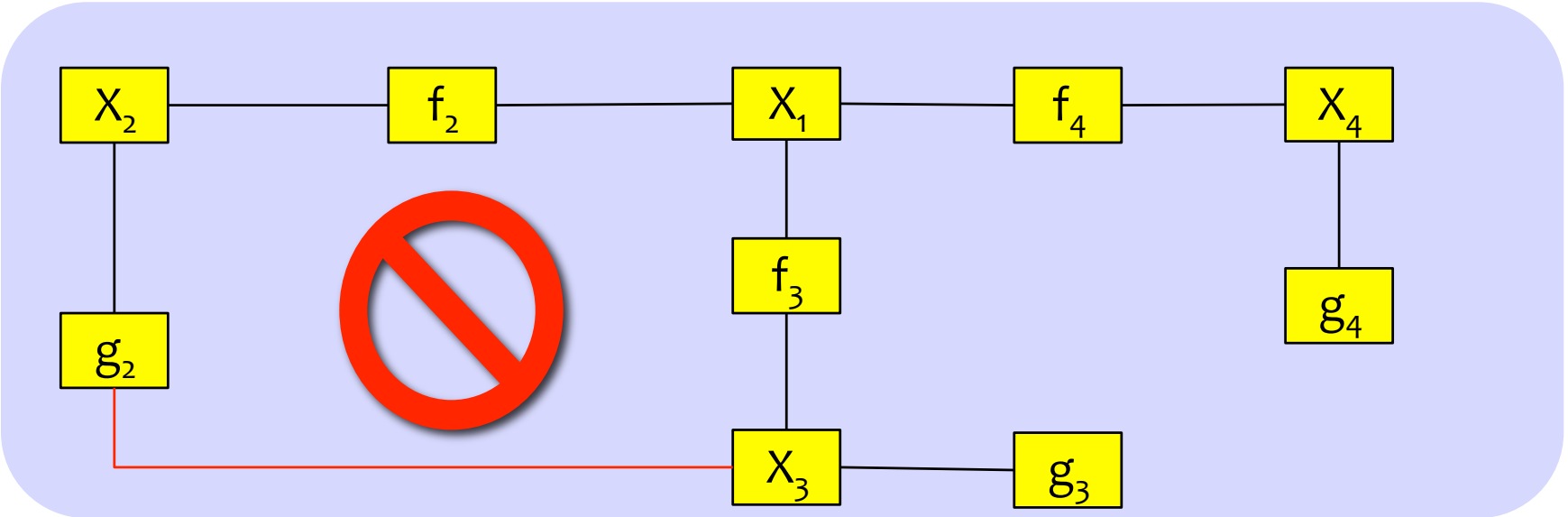
Aside: *opening* of vertices

- Given a vertex f(A,B,C) in a larger factor graph of a joint distribution p(A,B,C,… |**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

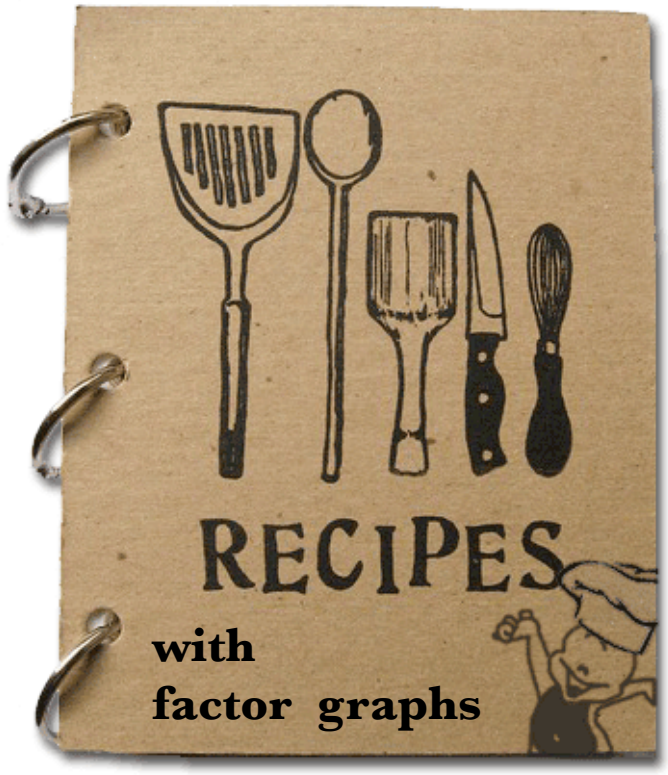
- Problem: When the factor graph has cycles: [S/M]PA gets stuck
- Solution: add artificial messages (e.g., set to "1")
- New Problem 1: XPA keeps running forever
- Solution: stop XPA after some time, and compute marginals
- New Problem 2: XPA becomes very unstable; messages tend to very small or very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



Outline

- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

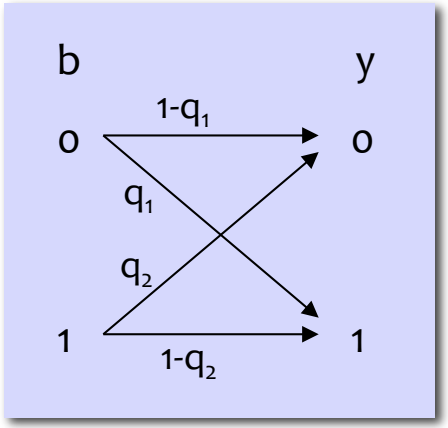
Inference using factor graphs



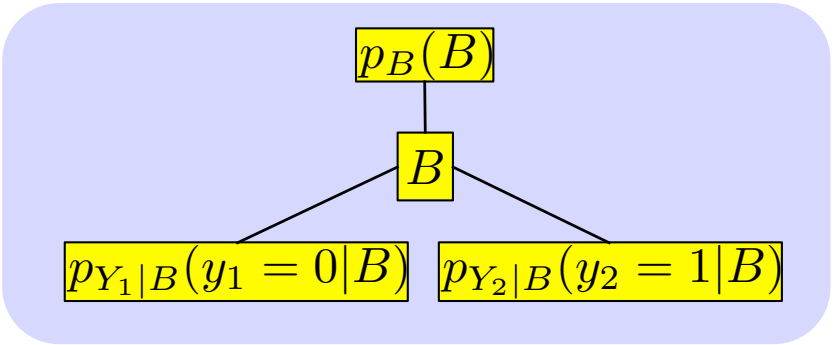
Inference using factor graphs

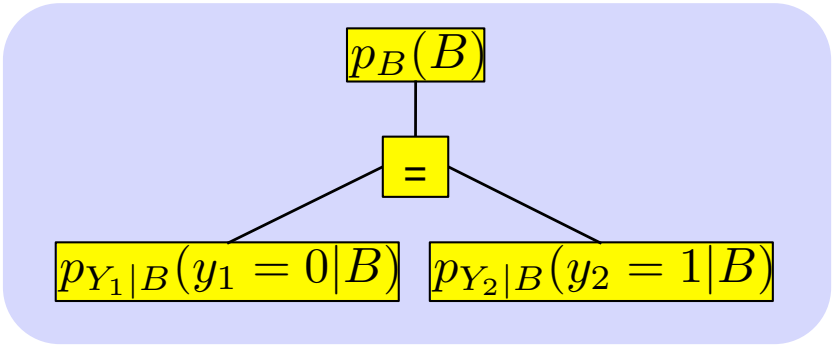
- Statistical inference
  - Parameter of interest **x**, a priori $p_X(\mathbf{x})$
  - Observation **y**, probabilistic mapping $p_{Y|X}(\mathbf{y}|\mathbf{x})$
  - Model *M*, encapsulating any additional assumptions
  - **x**,**y** can be discrete, continuous, hybrid
- 5 problems of interest
  1. Likelihood of the model $p(\mathbf{Y}=\mathbf{y}|\mathcal{M})$ √
  2. Marginal a posteriori distributions (MAPD) $p(X_k|\mathbf{Y}=\mathbf{y},\mathcal{M})$ √
  3. Certain characteristics of the MAPD (e.g., mean, mode) √
  4. A posteriori distribution (APD) $p(\mathbf{X}|\mathbf{Y}=\mathbf{y},\mathcal{M})$ X
  5. Certain characteristics of the APD (e.g., mean, mode) √

Running example



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





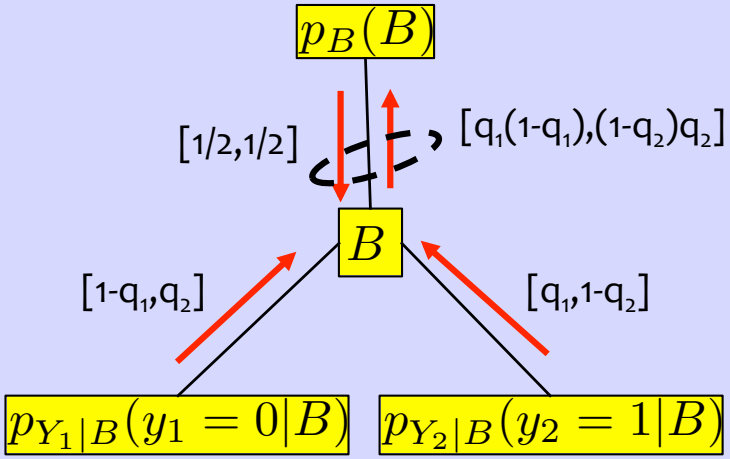
Problem 1 - likelihood of the model

Recipe

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

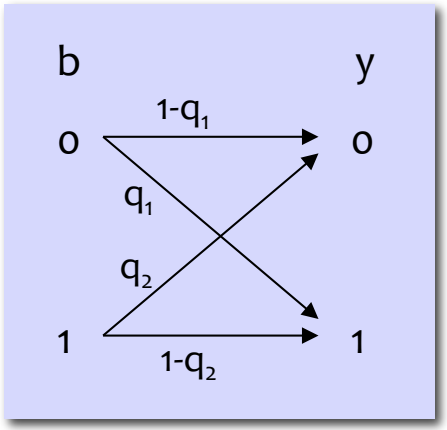
$p_{B,Y}(b=0, \mathbf{y}=[0,1]) = 0.5\, q_1(1-q_1)$
$p_{B,Y}(b=1, \mathbf{y}=[0,1]) = 0.5\, q_2(1-q_2)$

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



Recall:

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



‡ $\Sigma_a p_{A,B}(a,b) = p_B(b)$

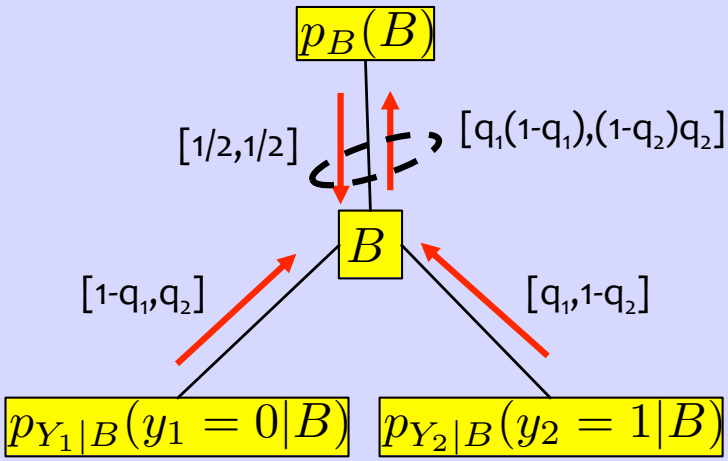
Problems 2,3 - marginal a posteriori distributions

Recipe

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

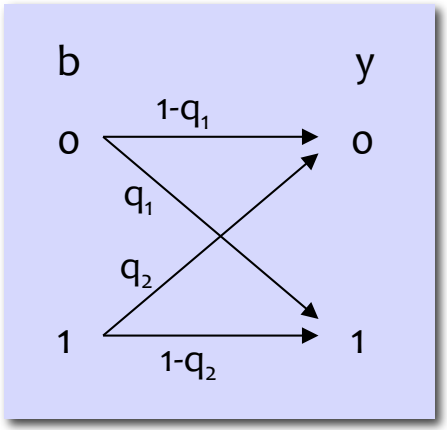
$p_{B,Y}(b=0, y=[0,1]) = 0.5\, q_1(1-q_1)$
$p_{B,Y}(b=1, y=[0,1]) = 0.5\, q_2(1-q_2)$

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



Recall:

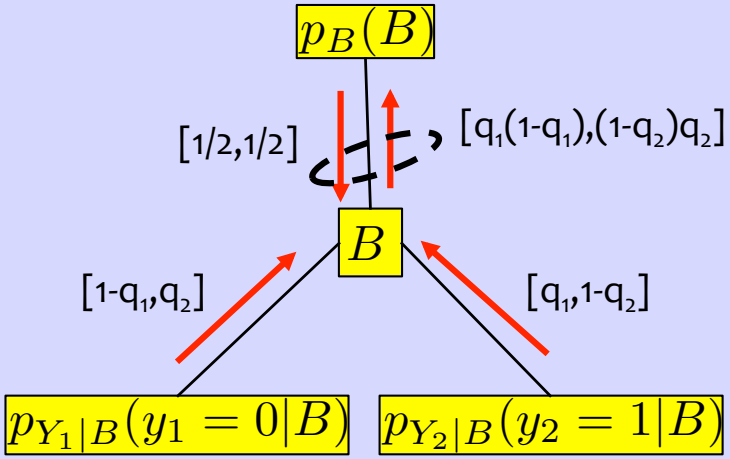
- $p(a|b) \propto p(a,b)$
- $p(a) = \Sigma_b p(a,b)$

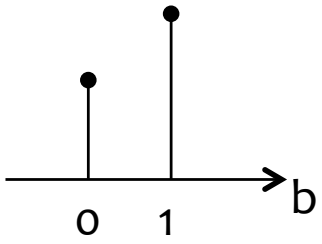


Problem 5 – mode of the a posteriori distribution

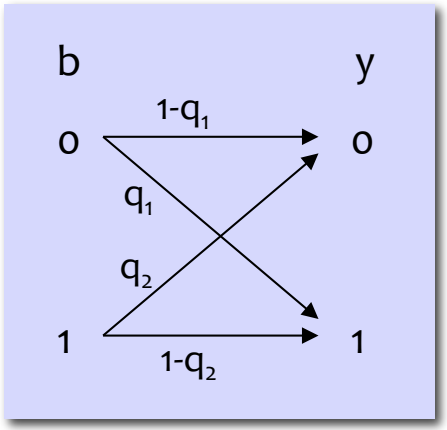
Recipe

1. Factorize the joint distribution $p_{X,Y}(\mathbf{x},\mathbf{y}) = p_{Y|X}(\mathbf{y}|\mathbf{x})p_X(\mathbf{x})$ (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 MPA, this yields max-marginals $q_k(x_k)$
4. The the mode of the max-marginals and concatenate

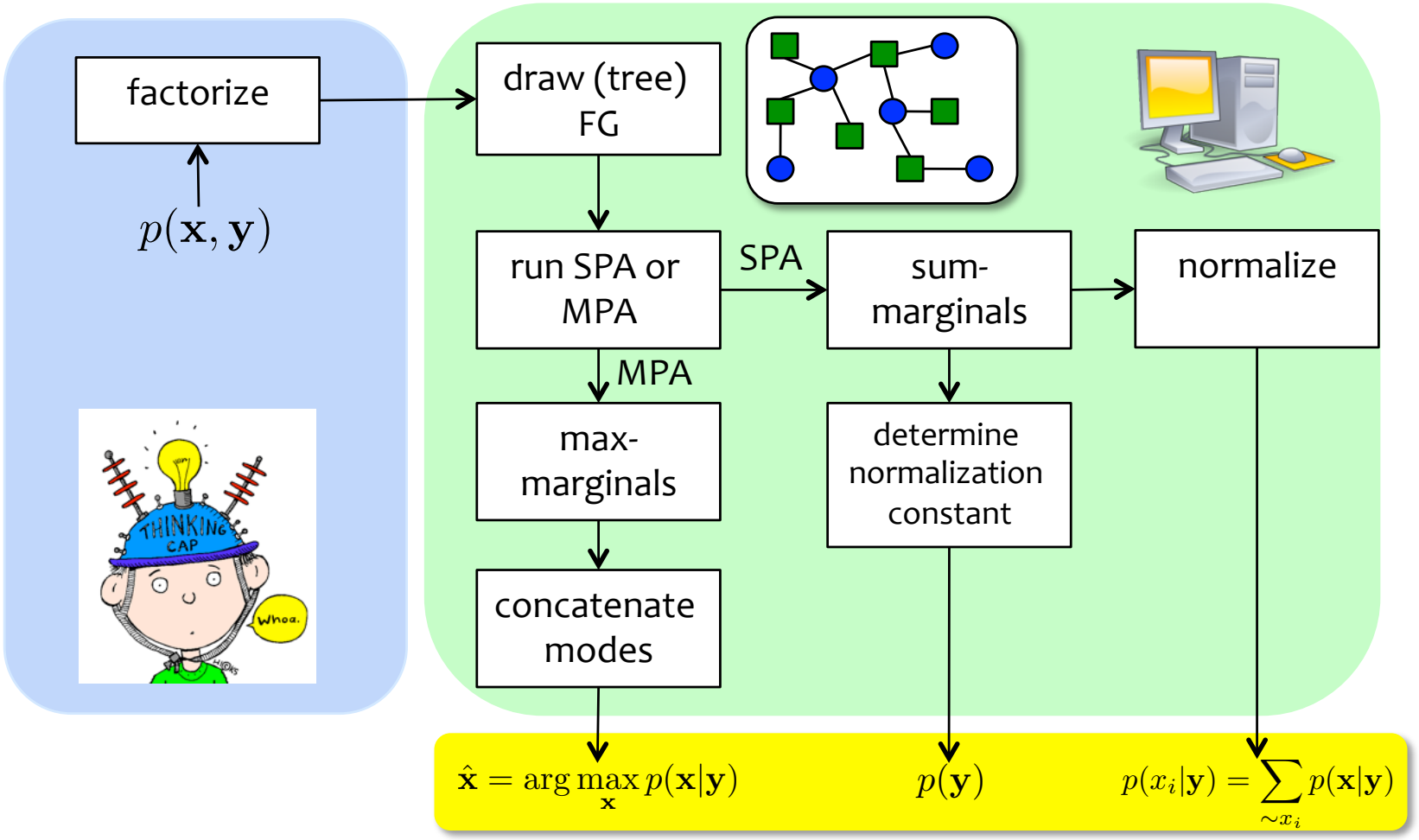




q(b=0) = 0.5 $q_1(1-q_1)$
q(b=1) = 0.5 $q_2(1-q_2)$



Inference recipe



Factor Graphs and Inference: Examples

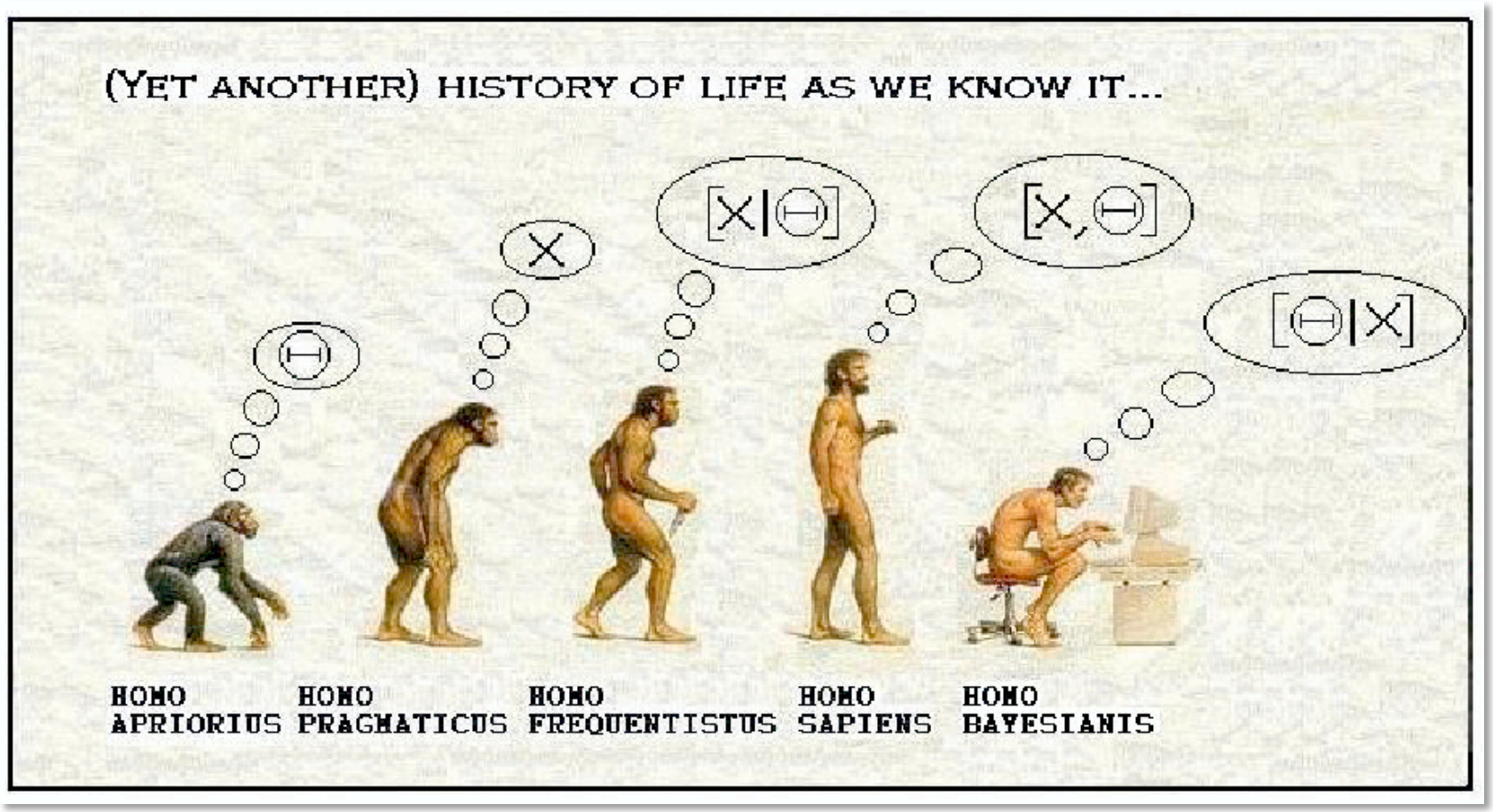


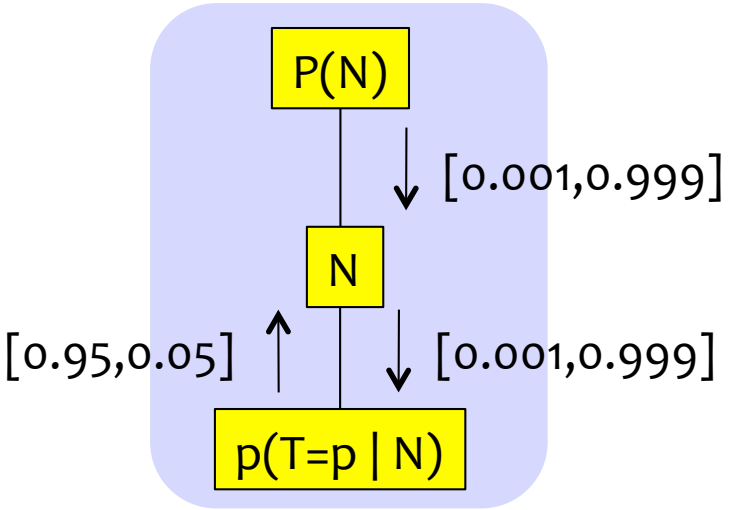
Image: Mike West, Duke University

The deadly disease - revisited

- Problem

  Nick is worried he is sick. He read in the newspaper 0.1% of the people is his city have contracted a deadly disease. He goes to the hospital to get tested. The doctor tells him the test is 95% reliable (i.e., it will give a correct result 95% of the time). The test is applied to Nick and turns out to be positive. With what probability does Nick have the disease?

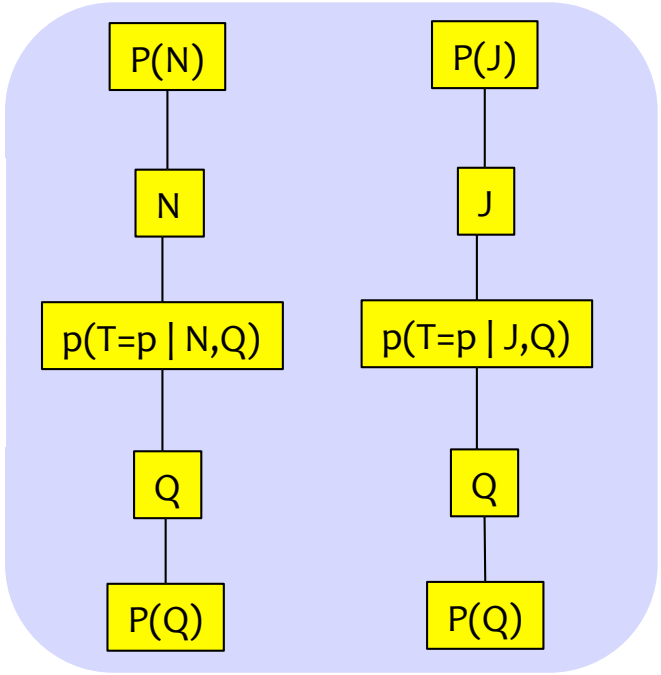
- Factor graph
  - Observation: "test is positive" = "T=p"
  - Unknown: "Nick's health" = "N" ∈ {h,s}
  - Determine p(N|T=p) = p(N,T=p)/P(T=p)
  - Draw factor graph of p(N, T=p), perform SPA

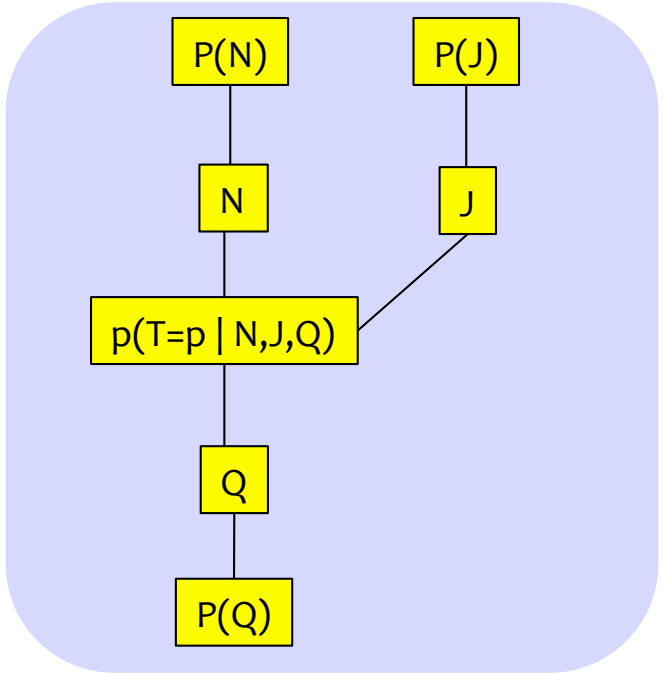


$$
\begin{aligned}
p(N=s,T=p) &= 0.95 \times 0.001 \\
p(N=h,T=p) &= 0.05 \times 0.999 \\
p(N=h|T=p) &= \frac{p(N=h,T=p)}{p(T=p)} \\
&= \frac{p(N=h,T=p)}{p(N=h,T=p)+p(N=s,T=p)} \\
&= 0.98
\end{aligned}
$$

The deadly disease -- variations

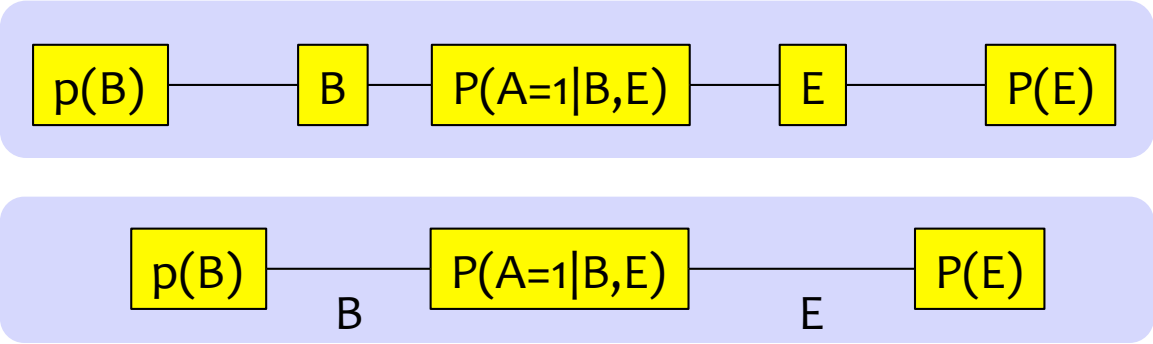
- Variation 1
  - Quality of test; Q ∈ {Good,Bad}
- Variation 2
  - Nick and Jane both get tested, result is positive
    - Independently *OR*
    - From the same test (e.g., same bad/good batch)



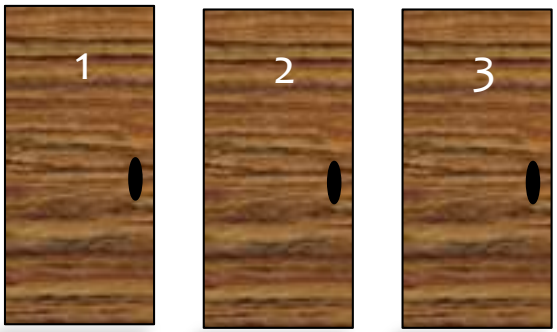


Example: The burglar alarm problem

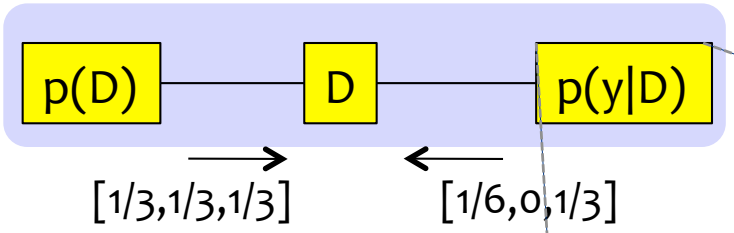
- Setup
  - A neighbor calls you at work, saying your burglar alarm is ringing
  - You know this could be due to either a burglar or an earthquake
  - Burglar and earthquake are a priori equally likely (probability 1/100)
- The following can trigger alarm, with the associate probabilities:
  - neither burglar nor earthquake = 1/1000
  - Earthquake, but no burglar = 1/10
  - Burglar, but no earthquake = 1/7
  - Both burglar and earthquake = 9/10
- Questions
  - What is the likelihood of the alarm ringing?
  - What is the APD of a burglary happening
  - What is the APD of an earthquake happening?



Example: *The Monty Hall Problem*



- Problem
  - Behind one door is a prize. Pick a door (say, door 1).
  - Host reveals one other door (say, door 2), showing a chicken.
  - Should you switch to door 3?
- Factor graph solution
  - Unknown: door holding the prize: $D \in \{1,2,3\}$
  - Observation: Choice C =1, Fake F = 2: y=[C=1,F=2]



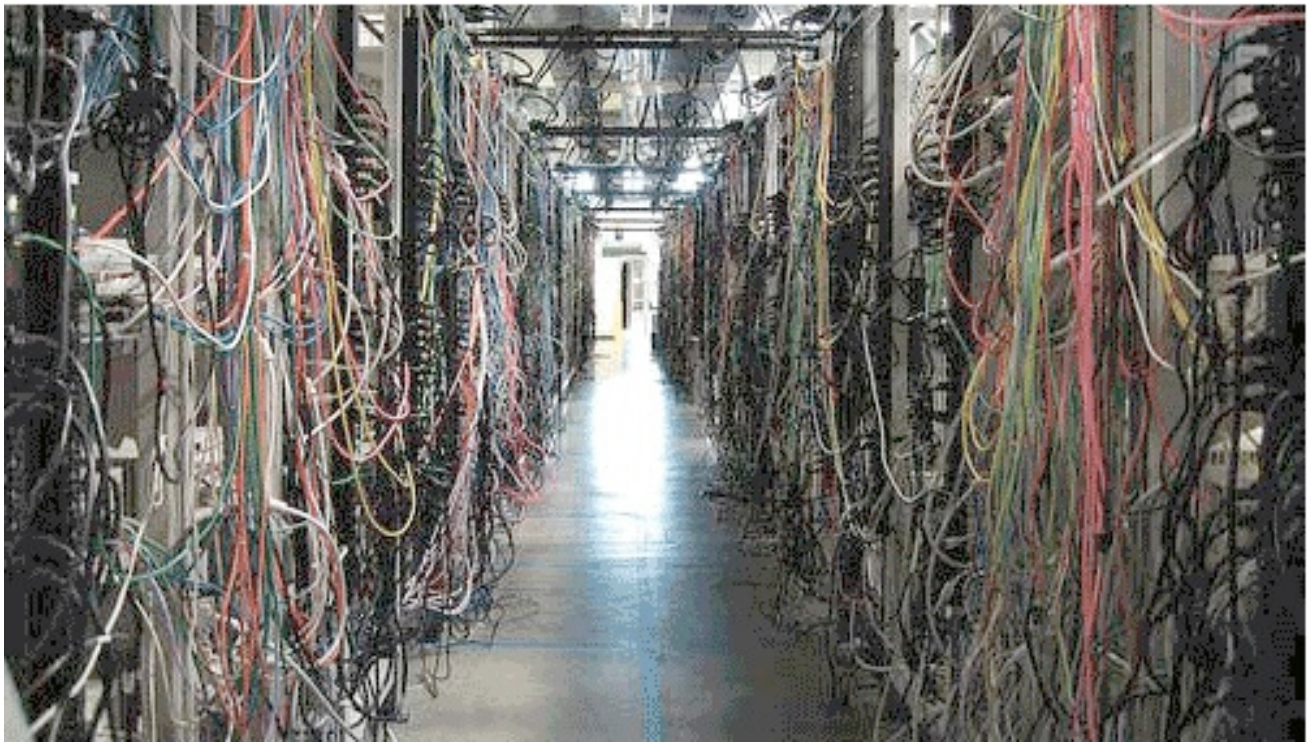
$p(D=1|C=1,F=2)=C * 1/6$
$p(D=2|C=1,F=2)=C * 0$
$p(D=3|C=1,F=2)=C * 1/3$

$p(C=1,F=2|D=1)=p(F=2|D=1,C=1)p(C=1|D=1)=1/2*1/3$
$p(C=1,F=2|D=2)=p(F=2|D=2,C=1)p(C=1|D=2)=0*1/3$
$p(C=1,F=2|D=3)=p(F=2|D=3,C=1)p(C=1|D=3)=1*1/3$

Outline

- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

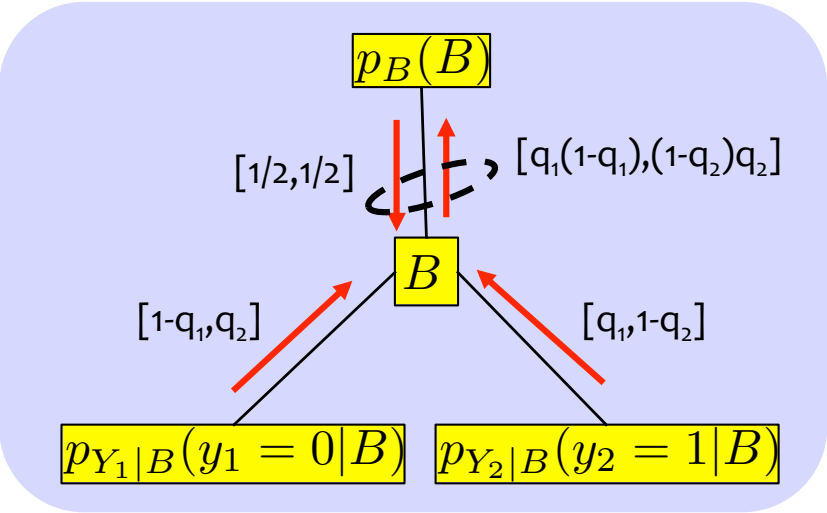
Practicalities



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

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

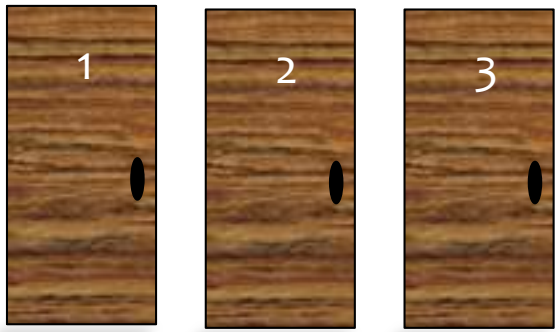
Messages representations

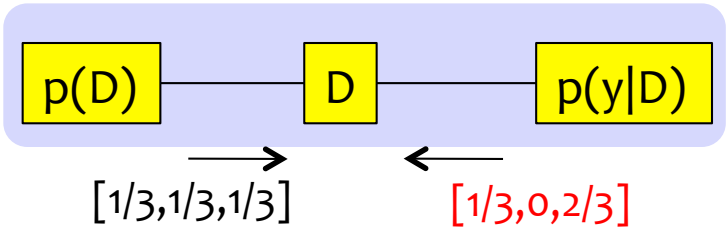
- Normalization
  - M = [a; b] becomes [a/(a+b); b/(a+b)] with normalization constant (NC) 1/(a+b)
  - Does not affect outcome of SPA (as long as we keep track of NCs)
  - Forgetting NCs does not affect marginal posteriors
  - Forgetting NCs affects likelihood of model

- 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





- Problem
  - Behind one door is a prize. Pick a door (say, door 1).
  - Host reveals one other door (say, door 2), showing a chicken.
  - Should you switch to door 3?
- Factor graph solution
  - Unknown: door holding the prize: $D \in \{1,2,3\}$
  - Observation: Choice C =1, Fake F = 2: y=[C=1,F=2]

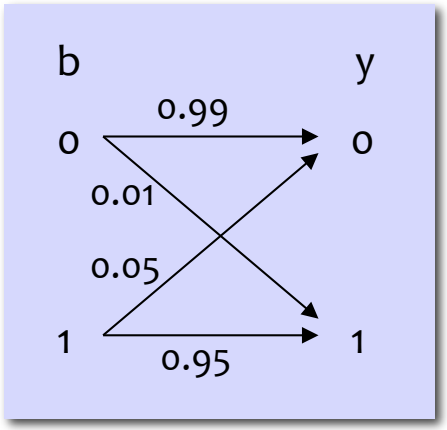
$p(D=1|C=1,F=2)=C * 1/6$
$p(D=2|C=1,F=2)=C * 0$
$p(D=3|C=1,F=2)=C * 1/3$

SWITCH!!!

$p(D=1|C=1,F=2)=C * 1/3$
$p(D=2|C=1,F=2)=C * 0$
$p(D=3|C=1,F=2)=C * 2/3$

SWITCH!!!

Example: normalization

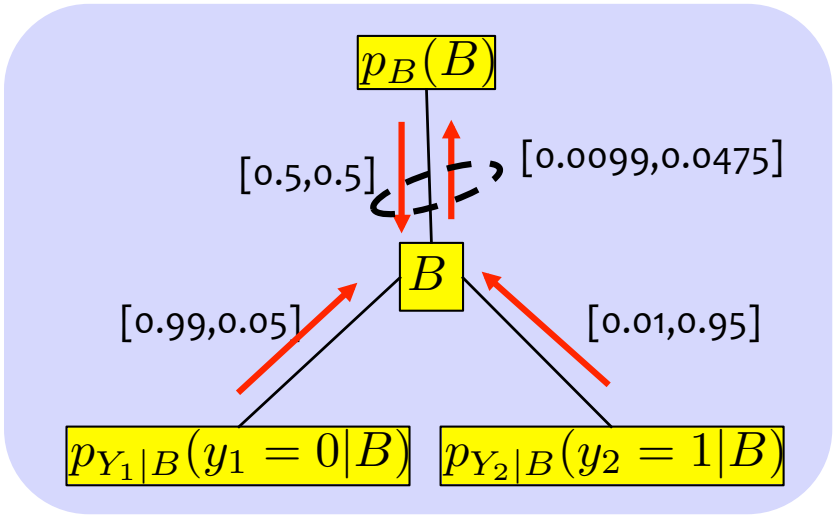


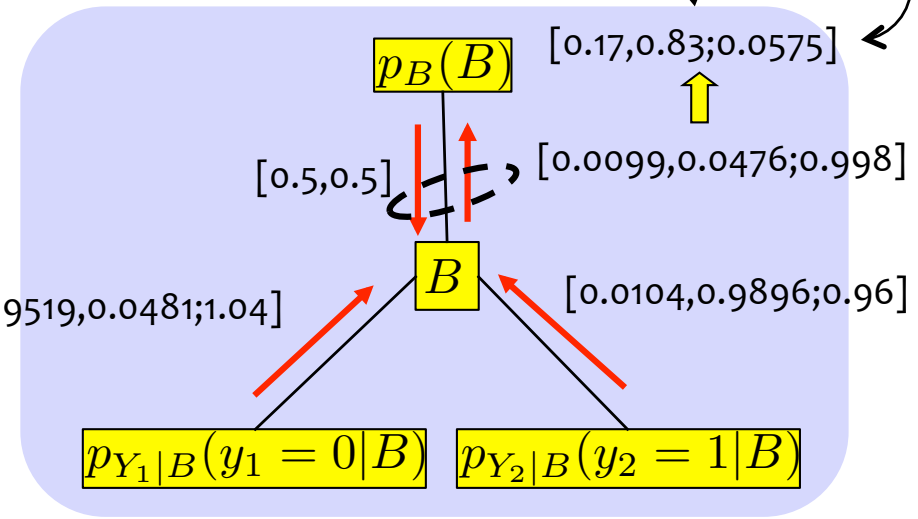
- Dividing all elements of the message with a constant, such that sum=1
  - Messages can be interpreted as distributions
  - Example: **m**=[0.001,0.02] → normalized = [0.0476,0.9524]
  - Normalization constant 0.021: **m'** = [0.0476,0.9524;0.021]

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





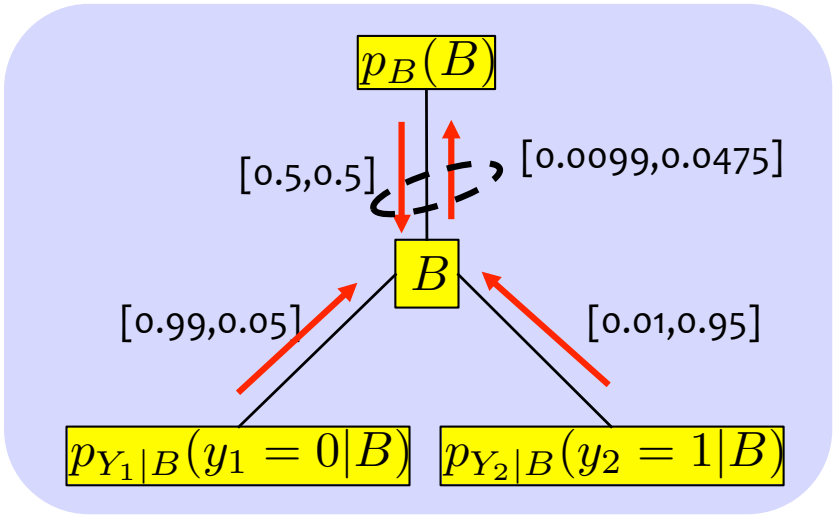
Example: normalization

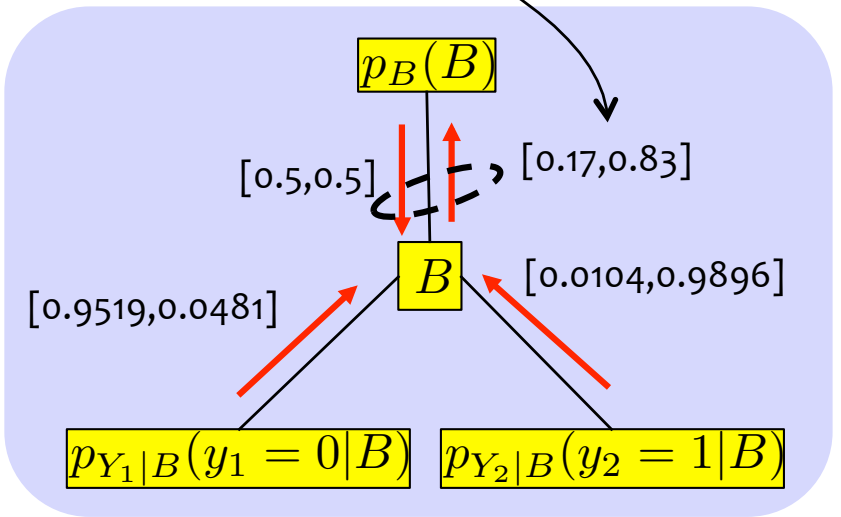
- Forgetting normalization constant is OK

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 ???





Example: log-domain

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

Probability domain:

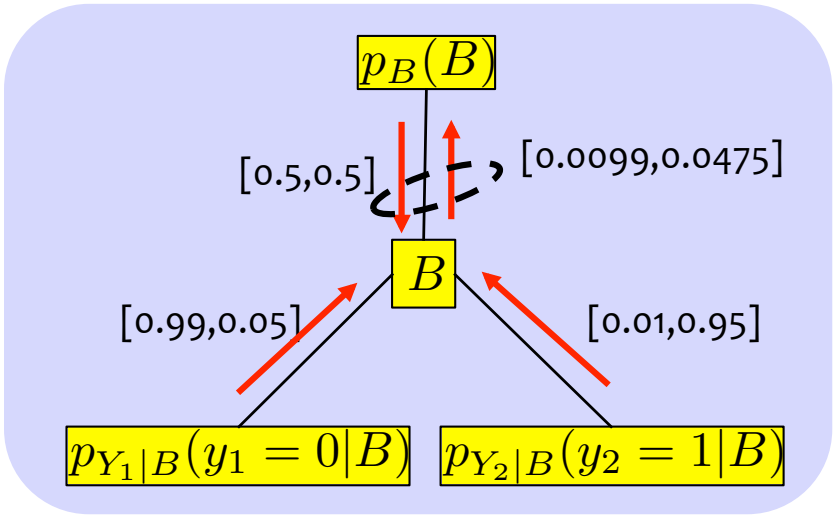
$p_{B|Y}(0|[0,1])$= 0.0099 /(0.0099+0.0475)=0.17
$p_{B|Y}(1|[0,1])$= 0.0475 /(0.0099+0.0475)=0.83

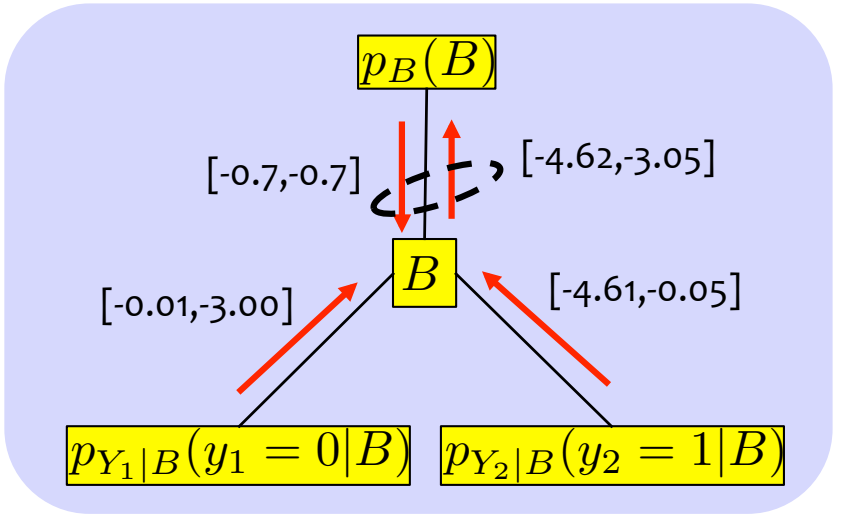
$p_Y([0,1])$= 0.0099+0.0475=0.0575

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(-4.622)+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

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

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

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

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

  - Defined recursively

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

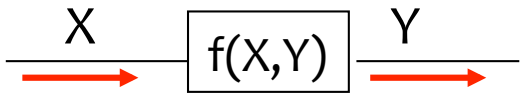
  - With core operation: 1 max + 1 table look-up
  - Very efficient

- 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

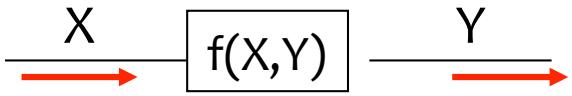
- Interpret messages a probability mass functions
- 3 approaches
  - Make them discrete, and use known techniques
  - Parametric representation (e.g., Gaussian mixture); message = vector of parameters
  - Non-parametric representation; message = vector of (weighted) samples
- We focus on non-parametric 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 $p_X(x)$



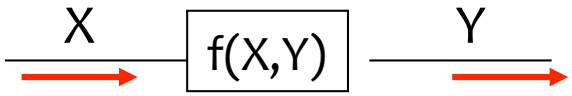
$$
\begin{aligned}
p_Y(y) &= \gamma \int f(x,y) p_X(x) dx \\
&= \int p_{Y|X}(y|x) p_X(x) dx \\
&\approx \int p_{Y|X}(y|x) \sum_{k=1}^{N} w_k \delta(x - x_k) dx \\
&= \sum_{k=1}^{N} w_k p_{Y|X}(y|x_k)
\end{aligned}
$$

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

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

Non-parametric representation: method 2

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



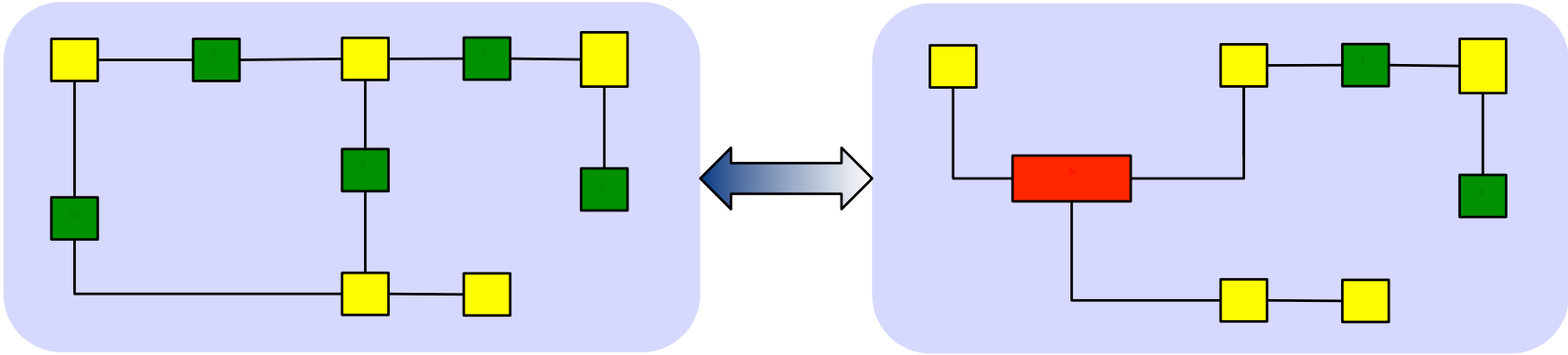
$$
\begin{aligned}
p_Y(y) &= \gamma \int f(x,y) p_X(x) dx \\
&= \int p_{Y|X}(y|x) p_X(x) dx \\
&\approx \int p_{Y|X}(y|x) \sum_{k=1}^{N} w_k \delta(x - x_k) dx \\
&= \sum_{k=1}^{N} w_k p_{Y|X}(y|x_k)
\end{aligned}
$$

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

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

Graphs with cycles

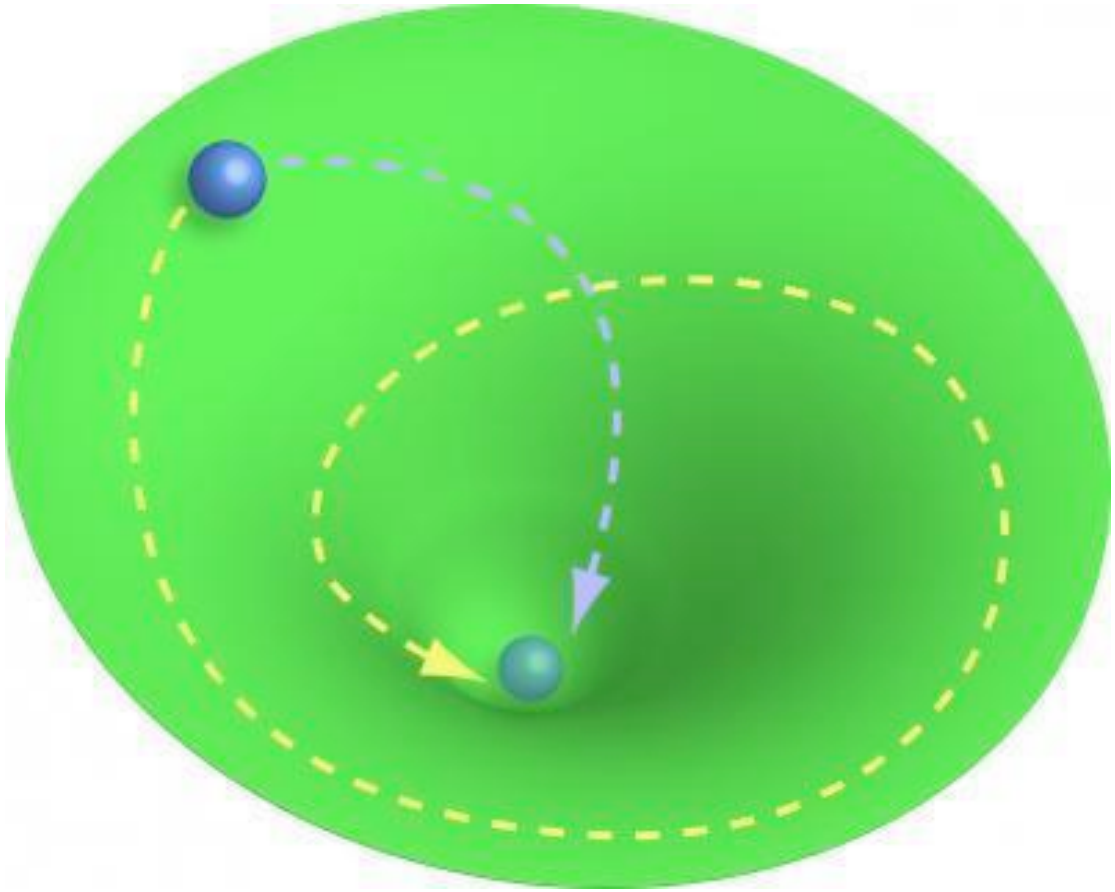
- Traditional approach
  - Convert to tree using junction tree methods



- 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

Convergence behavior



Turbo decoding example

- Typical behavior

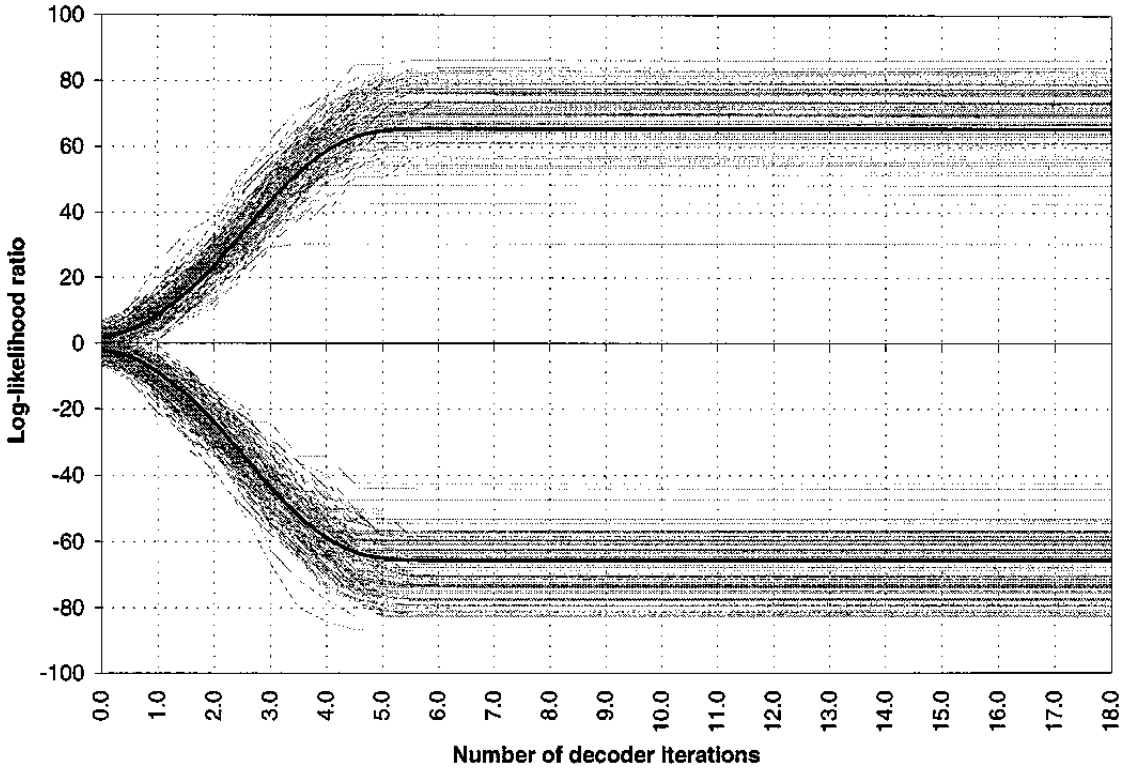


Fig. 1. Bit convergence of a typical frame in which all errors are corrected. (SNR = 1.77 dB, random interleaver).

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

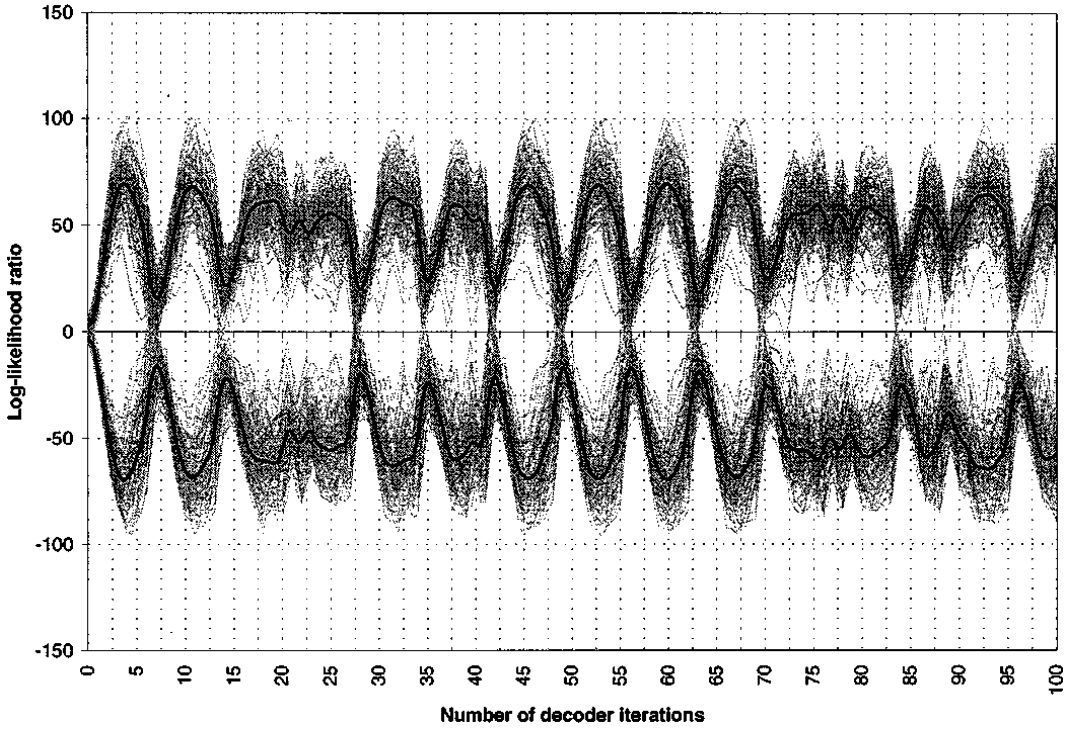


Fig. 11. Bit convergence of a frame which shows large oscillations to 100 iterations. (SNR= 3.27 dB, constrained interleaver).

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 $p(\mathbf{x}) \propto \exp\left(-\mathbf{x}^T\mathbf{W}\mathbf{x} + \mathbf{b}^T\mathbf{x}\right)$

$$= \prod_{i,j} \exp(-w_{ij}x_i x_j) \times \prod_i \exp(b_i x_i)$$

1. Convergence is guaranteed when W is diagonal dominant (Weiss, Freeman, 2001)

$$|w_{ii}| > \sum_{j \neq i} |w_{ij}|, \forall i$$

2. Convergence is guaranteed when W=I-R, for which ρ(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

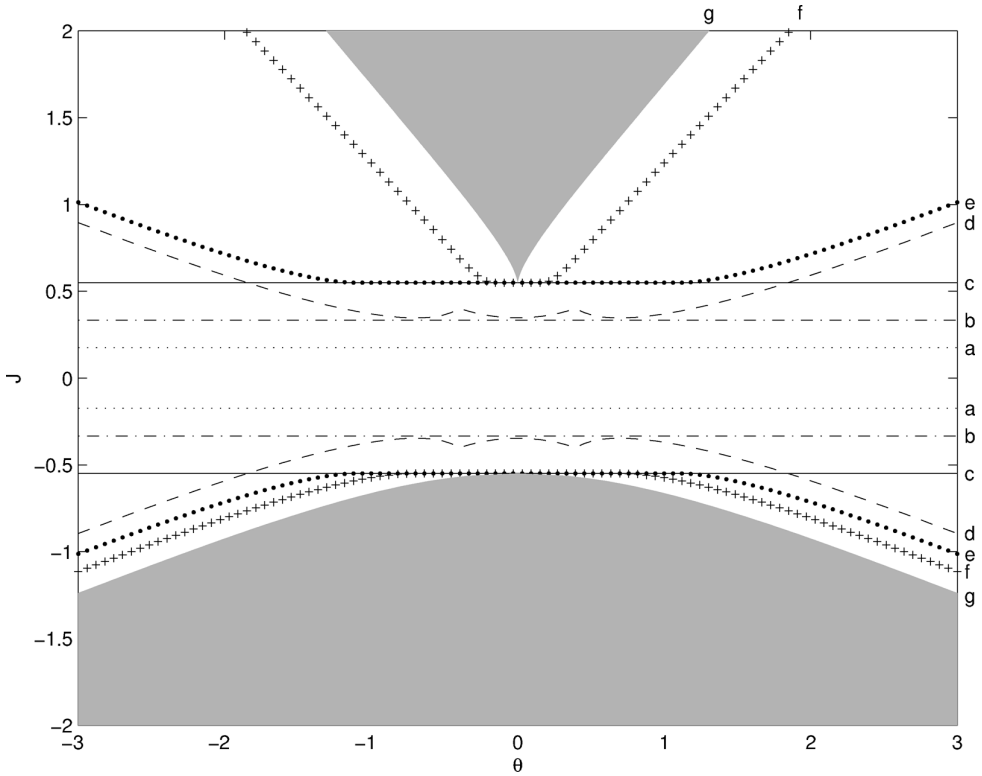
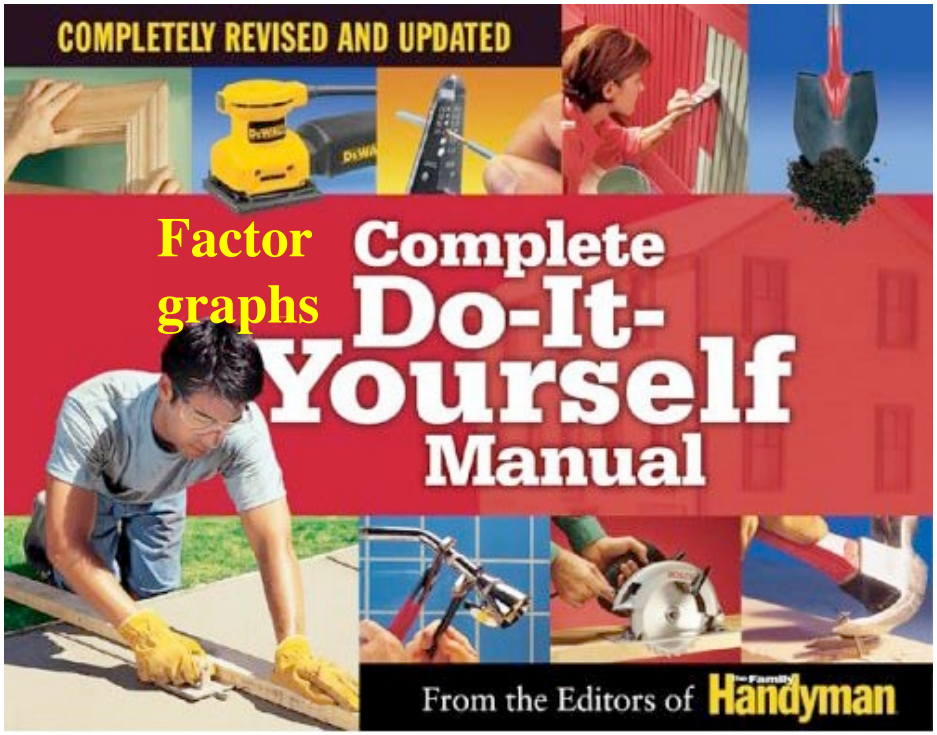


Fig. 4. Comparison of various BP convergence bounds for the fully connected $N = 4$ binary Ising model with uniform coupling J and uniform local field θ. a: Heskes' condition, b: Simon's condition, c: spectral radius condition, d: Dobrushin's condition, e: improved spectral radius condition for $m =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

Outline

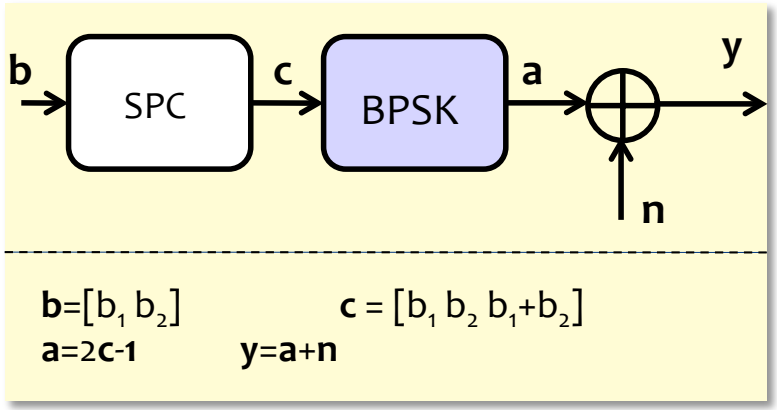
- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

Example



Build your own factor graph receiver

- Transmitter



- Develop optimal bit-by-bit detector

$$\hat{b}_k = \arg\max_{b_k \in \mathbb{B}} p(b_k|\mathbf{y})$$

- Implement in MATLAB (less than 20 lines)

Now: MATLAB implementation

- In LLR domain

$$\lambda = \log \frac{\mu(1)}{\mu(0)}$$

$$= \log \mu(1) - \log \mu(0)$$

- Recall that

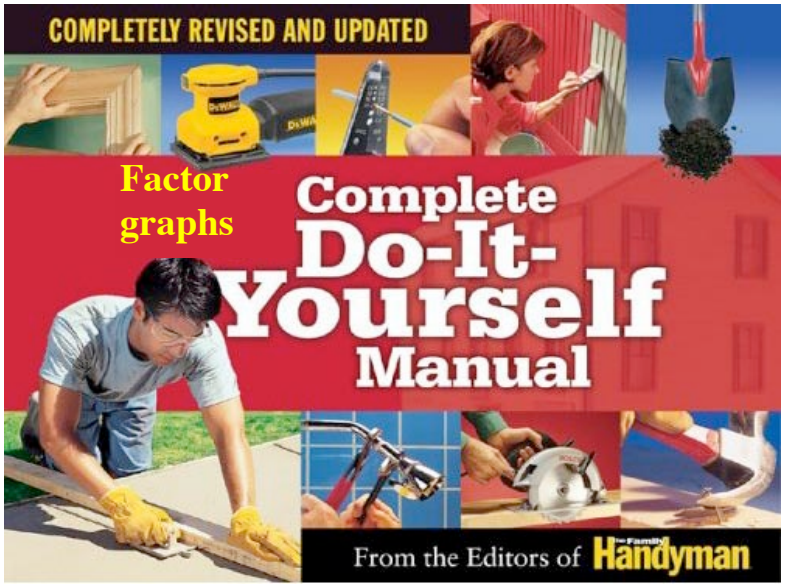
$$\mu_a \times \mu_b \rightarrow \log \mu_a + \log \mu_b$$

$$\mu_a + \mu_b \rightarrow \max^*(\log \mu_a, \log \mu_b)$$

- Other conversion rules

$$\log \mu_a(1) = +\lambda_a/2$$

$$\log \mu_a(0) = -\lambda_a/2$$

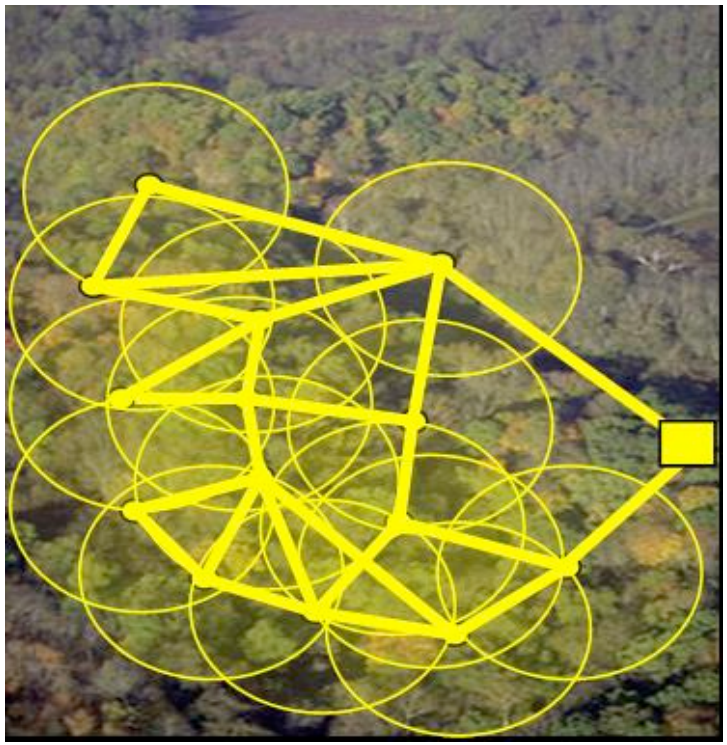


Outline

- Applications
- Background and terminology
- Bayesian detection
- Tool 1: Bayesian graphical models
  - Basics
  - Recipe for Bayesian inference
  - Practicalities
  - A worked example: a digital receiver
- Tool 2: Belief consensus
  - Basics
  - Convergence
- Applications revisited
- Variational interpretation
- Summary and conclusions

Motivation

- Many individual nodes/observers, form network
- How to perform distributed inference, optimization, or control without centralized processing?
- Connection to flocking behavior



Problem formulation

Model

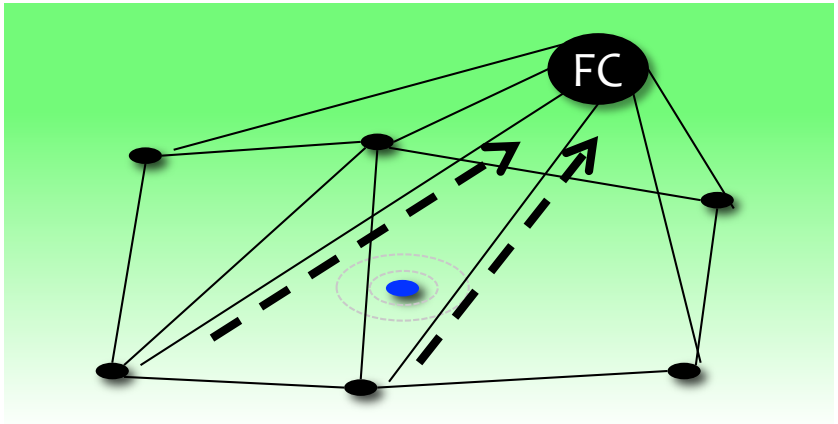
- N sensors, form a network, $\mathcal{N}_i$ = neighbors of node i
- Independent observation at sensor i: $y_i \in R^M$
- Prior p(x), $x \in \{0,1\}$, known to **all** sensors
- Likelihood $p(y_i|x)$ known **only** to sensor i
- No packet collisions

Goal

- Compute $p(x|\mathbf{y})$ at every sensor, without centralized processing

Centralized approach

- Version 1: broadcast to fusion center



- Version 2: routing to fusion center