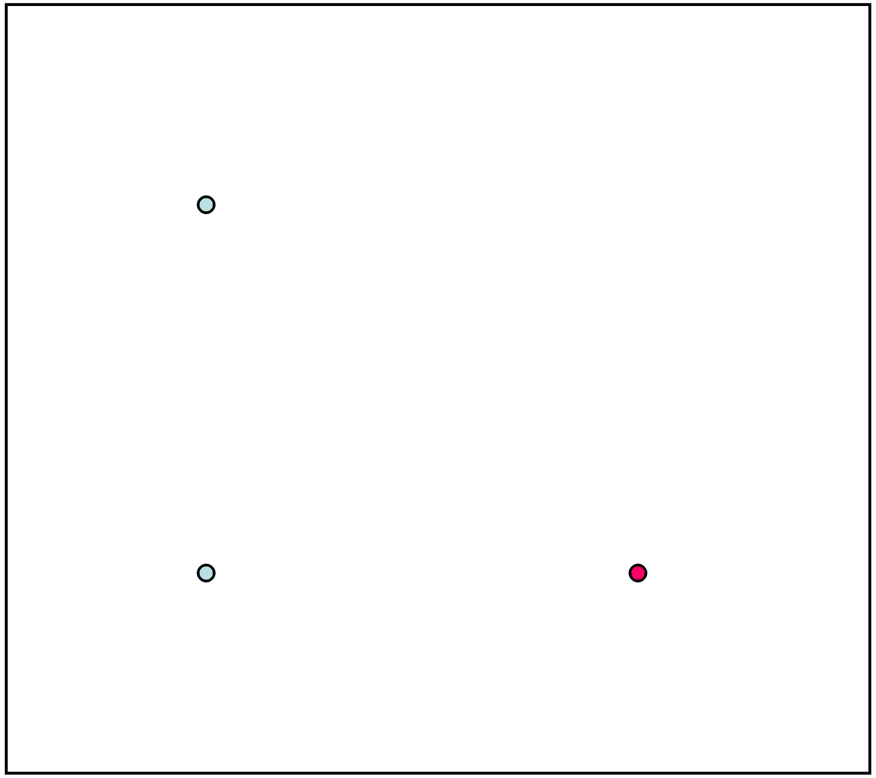
Expectation Propagation

Tom Minka
Microsoft Research, Cambridge, UK

2006 Advanced Tutorial
Lecture Series, CUED

A typical machine learning problem

Spam filtering by linear separation

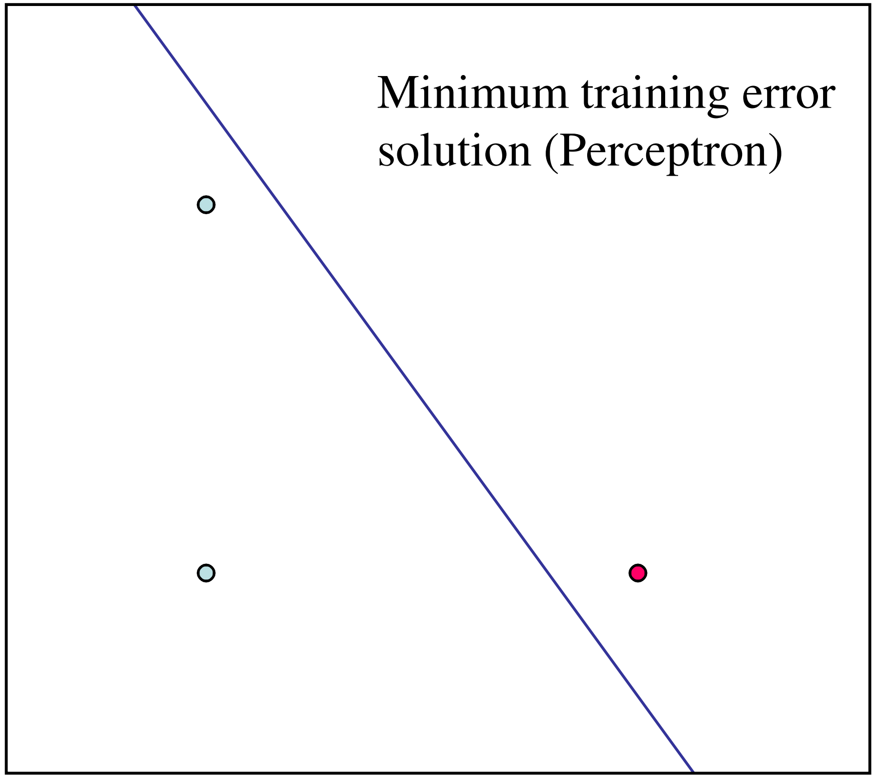


Not spam

Spam

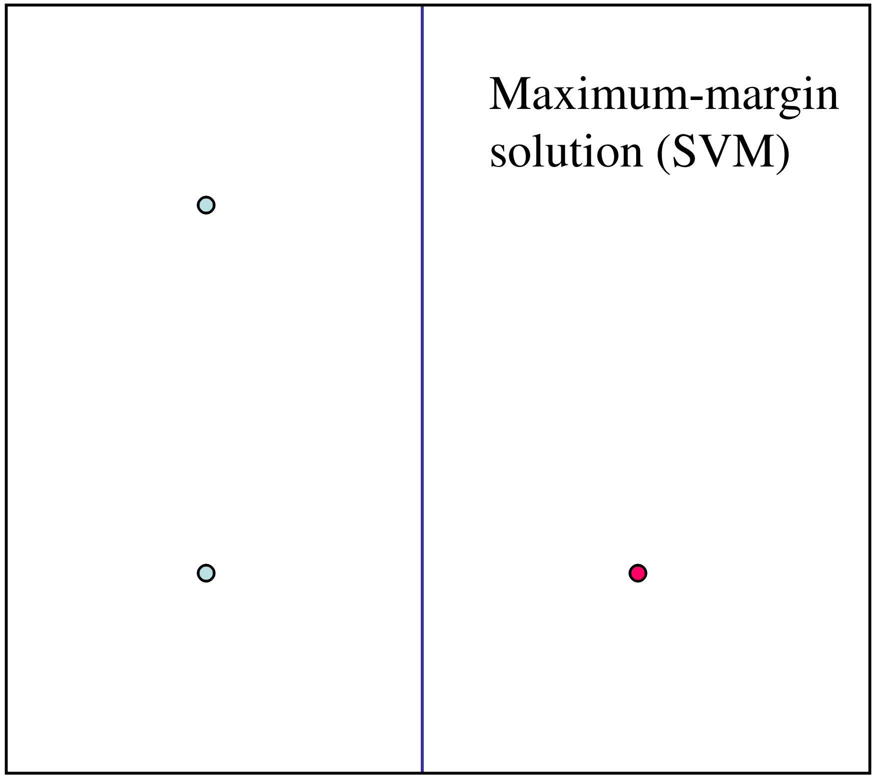
Choose a boundary that will generalize to new data

Linear separation



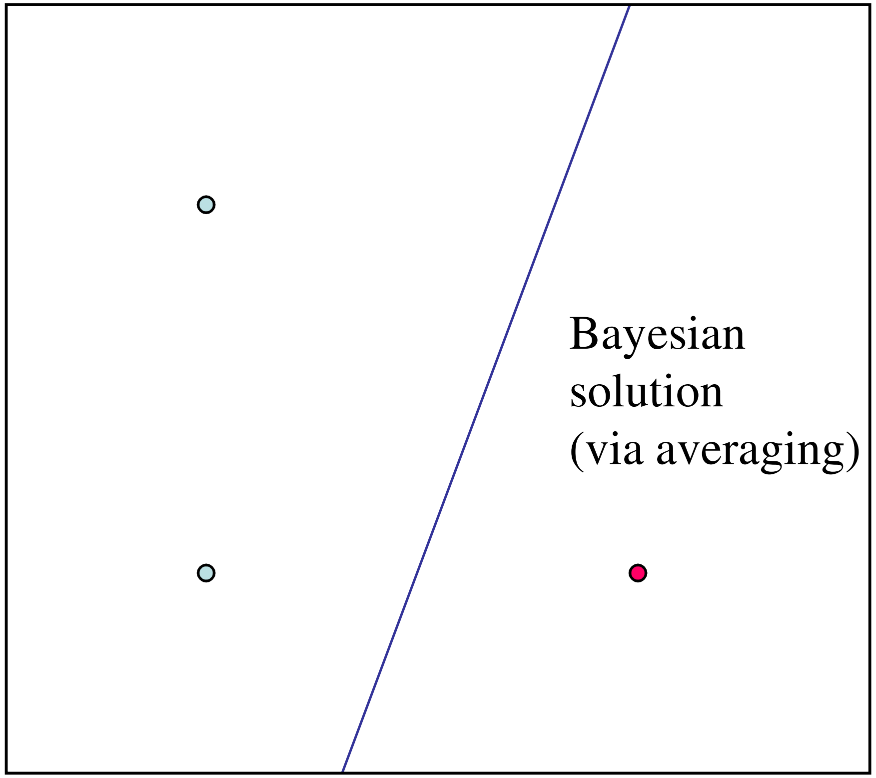
Too close to data – won't generalize well

Linear separation



Ignores information in the vertical direction

Linear separation



Has a margin, and uses information in all dimensions

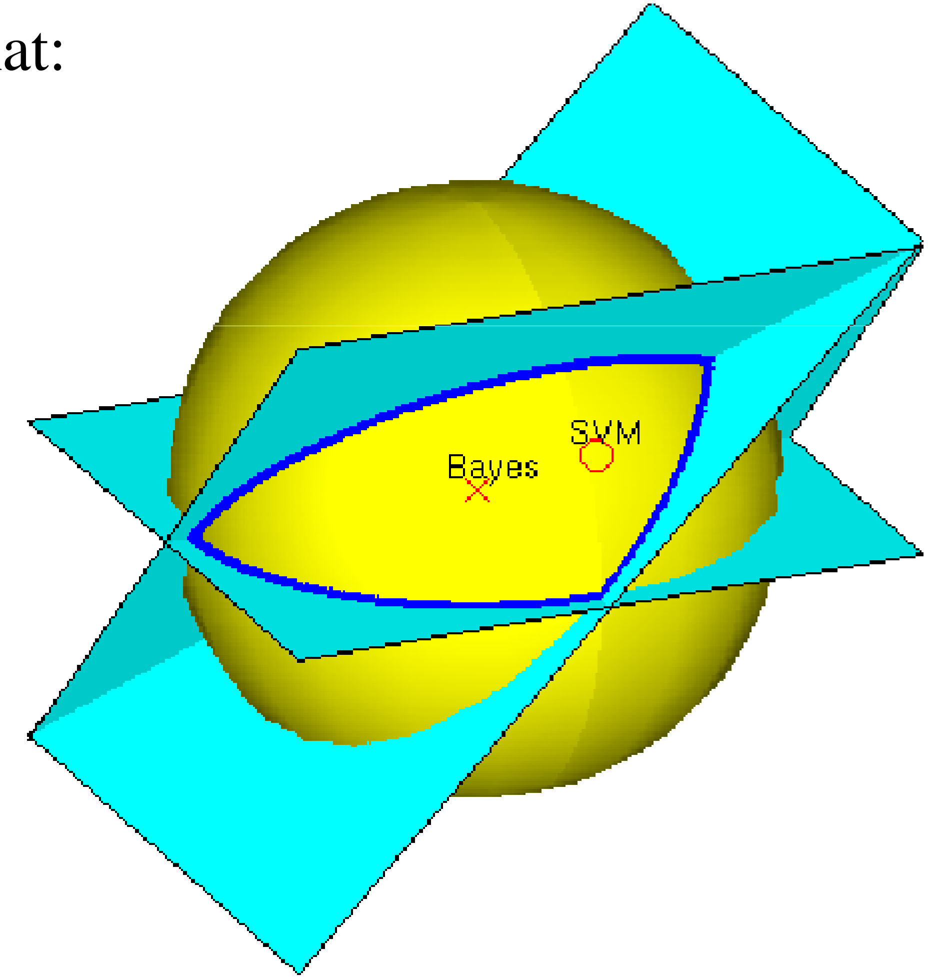
Geometry of linear separation

Separator is any vector w such that:

$$\mathbf{w}^T \mathbf{x}_i > 0 \qquad \text{(class 1)}$$

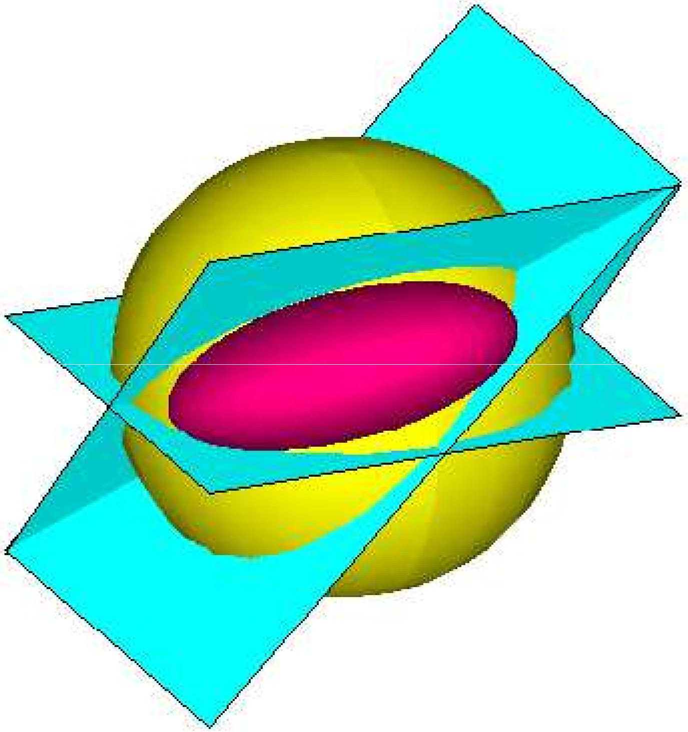
$$\mathbf{w}^T \mathbf{x}_i < 0 \qquad \text{(class 2)}$$

$$\|\mathbf{w}\| = 1 \qquad \text{(sphere)}$$



This set has an unusual shape
SVM: Optimize over it
Bayes: Average over it

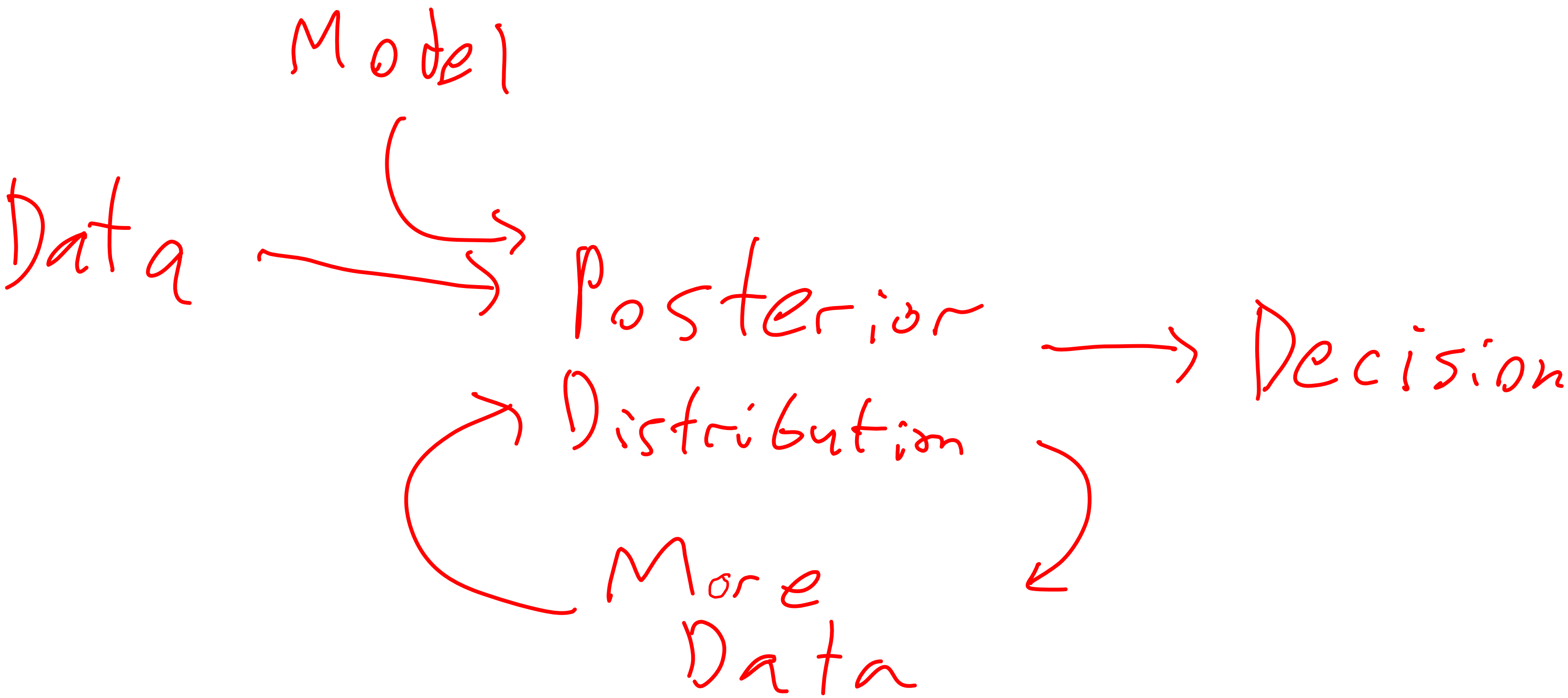
Performance on linear separation



EP Gaussian approximation to posterior

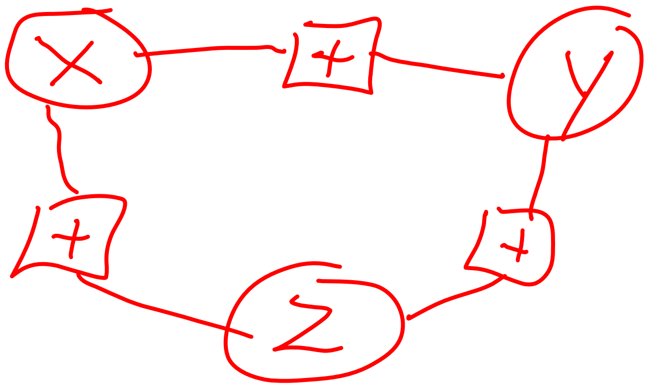
Bayesian paradigm

- Consistent use of probability theory for representing unknowns (parameters, latent variables, missing data)

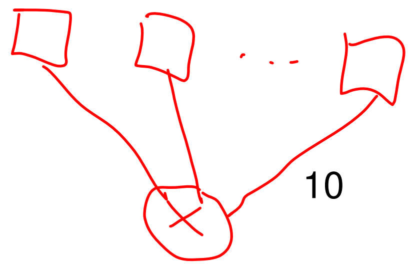


Factor graphs

- Shows how a function of several variables can be factored into a product of simpler functions
- $f(x,y,z) = (x+y)(y+z)(x+z)$
- Very useful for representing posteriors



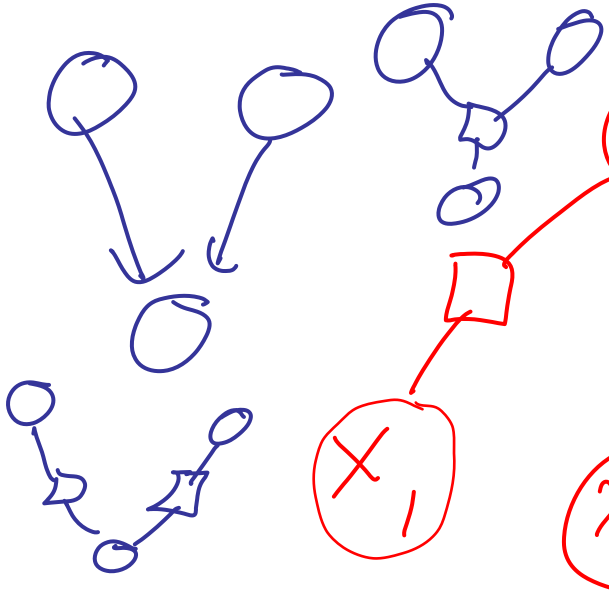
$$f(x) = x^n$$
$$= x \cdot x \cdots x$$

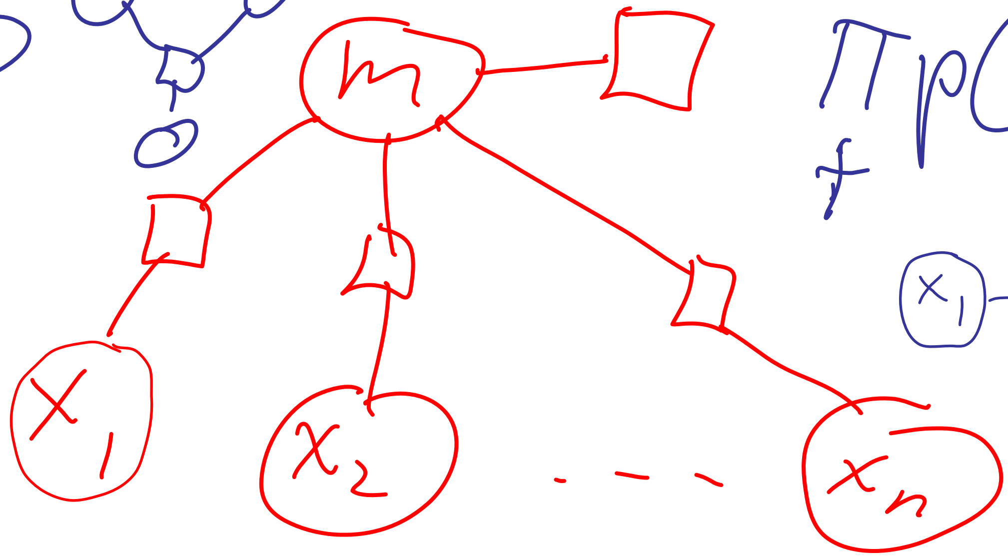


Example factor graphs

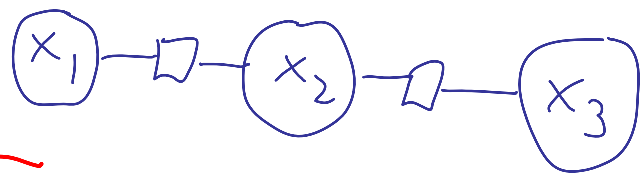
$$p(x_i \mid m) = N(x_i ; m, 1)$$

$$p(m \mid x_1 \ldots x_n) \propto p(m)\, p(x_1 \mid m) \ldots p(x_n \mid m)$$





$$\prod_t p(x_t \mid x_{t-1})$$



Two tasks

- Modeling
  – What graph should I use for this data?
- Inference
  – Given the graph and data, what is the mean of x (for example)?
  – Algorithms:
    - Sampling
    - Variable elimination
    - Message-passing (Expectation Propagation, Variational Bayes, …)

Division of labor

- Model construction
  - Domain specific (computer vision, biology, text)
- Inference computation
  - Generic, mechanical
  - Further divided into:
    - Fitting an approximate posterior
    - Computing properties of the approx posterior

Benefits of the division

- Algorithmic knowledge is consolidated into general graph-based algorithms (like EP)
- Applied research has more freedom in choosing models
- Algorithm research has much wider impact

Take-home message

- Applied researcher:
  - express your model as factor graph
  - use graph-based inference algorithms
- Algorithm researcher:
  - present your algorithm in terms of graphs

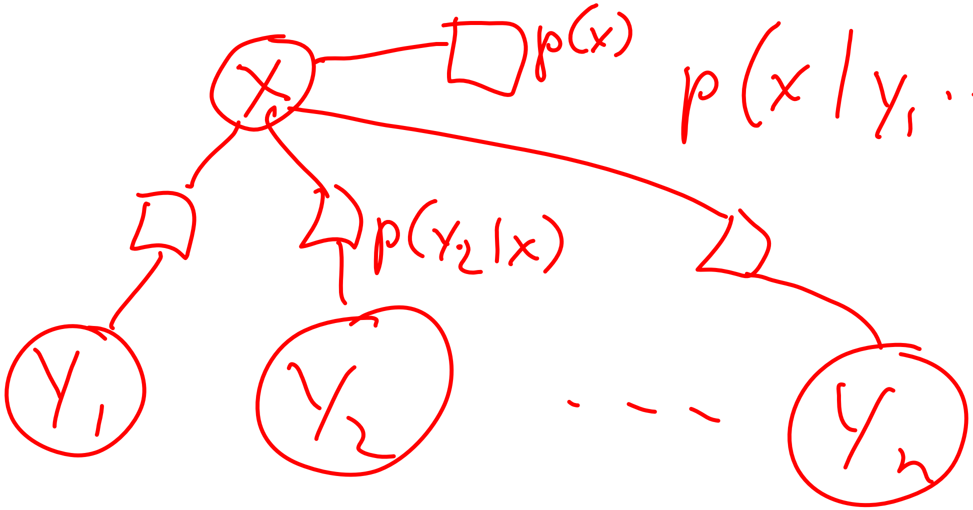
A (seemingly) intractable problem

Clutter problem

- Want to estimate x given multiple y's

$$p(x) = \mathcal{N}(x; 0, 100)$$

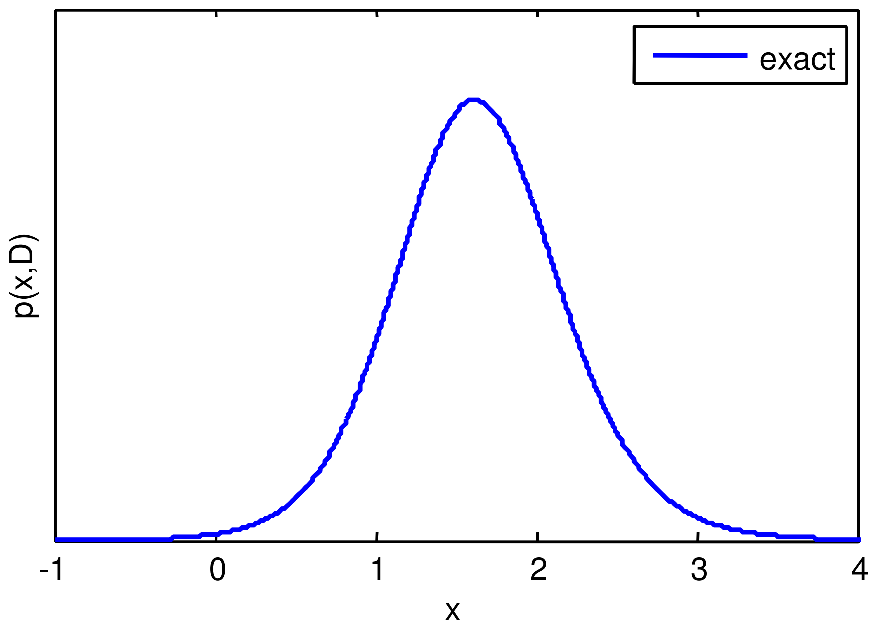
$$p(y_i|x) = (0.5)\mathcal{N}(y_i; x, 1) + (0.5)\mathcal{N}(y_i; 0, 10)$$



$p(x|y_1 \cdots y_n) \propto p(x) \prod_i p(y_i|x)$

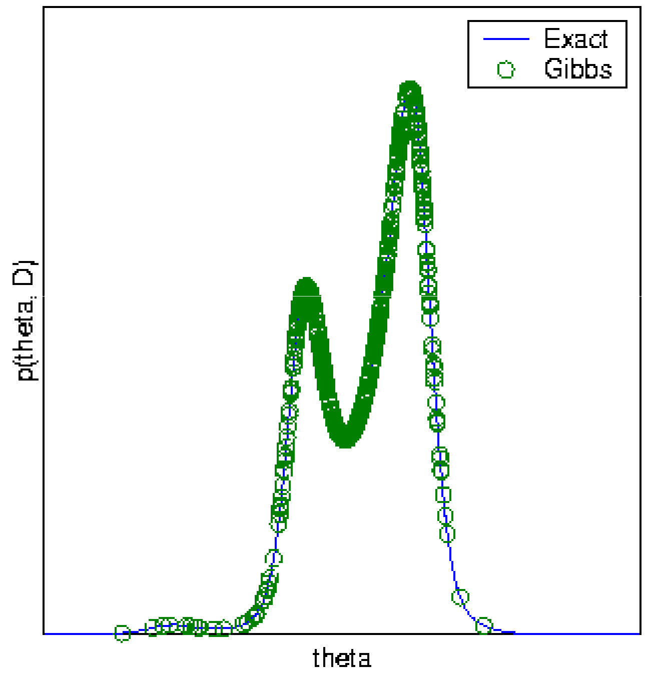
2^n Gaussians

Exact posterior



Representing posterior distributions

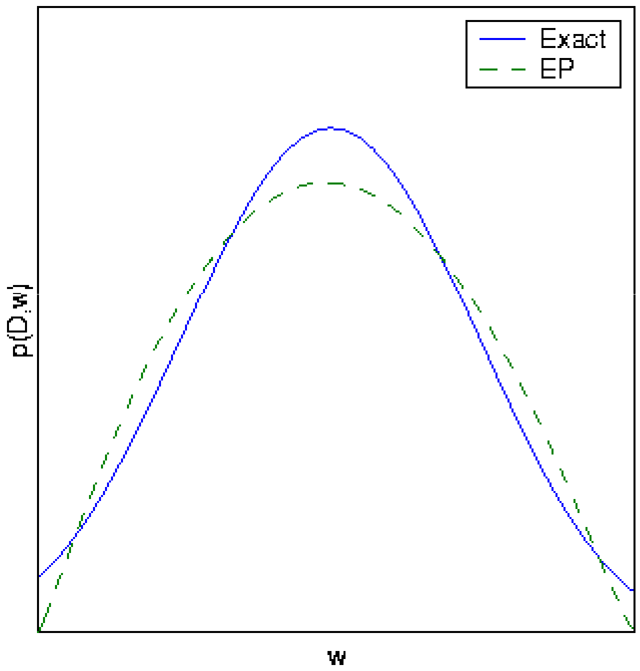
Sampling



Good for complex,
multi-modal distributions

Slow, but predictable accuracy

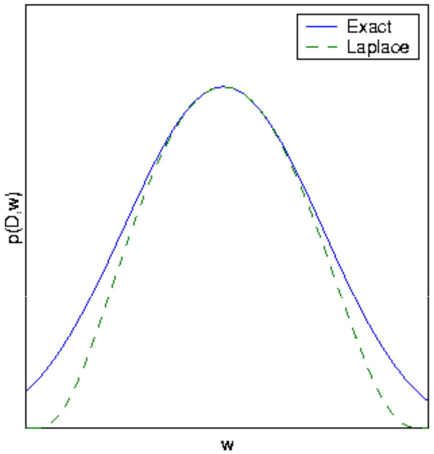
Deterministic approximation



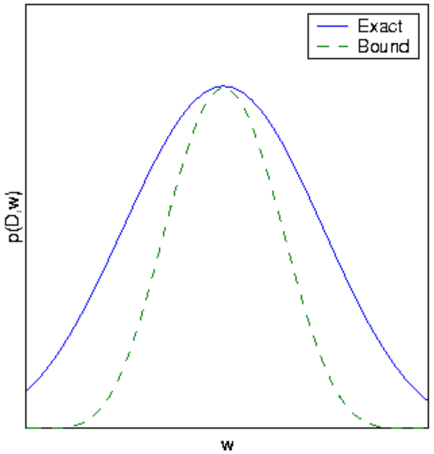
Good for simple,
smooth distributions

Fast, but unpredictable accuracy

Deterministic approximation

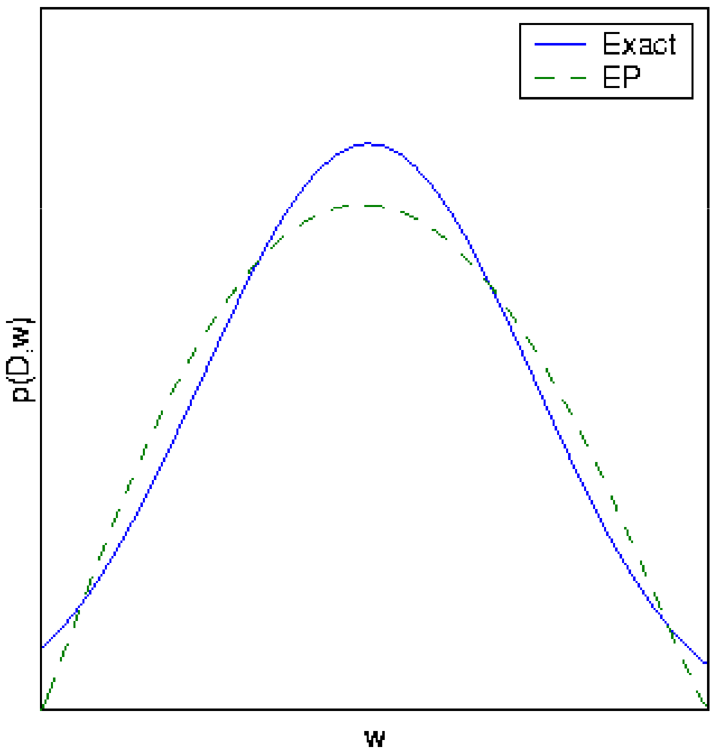


Laplace's method
• Bayesian curve fitting, neural networks (MacKay)
• Bayesian PCA (Minka)



Variational bounds
• Bayesian mixture of experts (Waterhouse)
• Mixtures of PCA (Tipping, Bishop)
• Factorial/coupled Markov models (Ghahramani, Jordan, Williams)

Moment matching



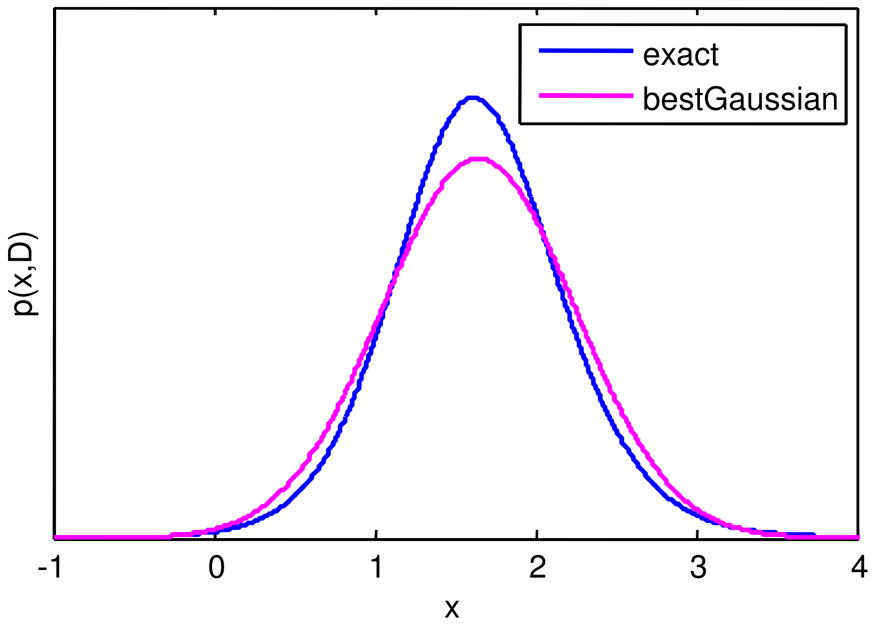
Another way to perform deterministic approximation
• Much higher accuracy on some problems

Assumed-density filtering
(1984)

Loopy belief propagation
(1997)

Expectation Propagation
(2001)

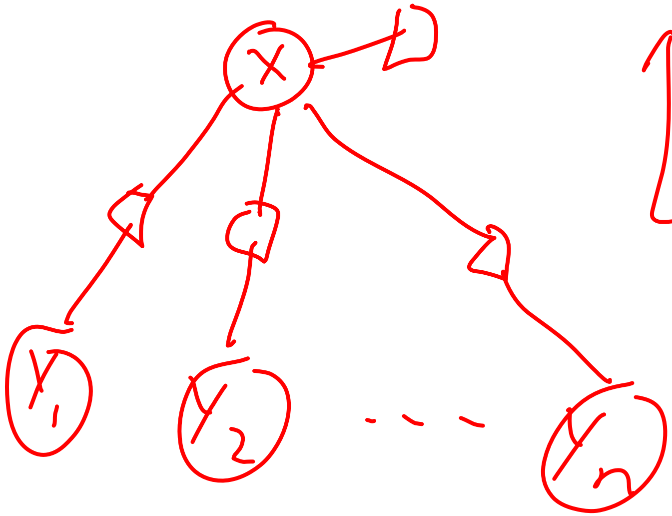
Best Gaussian by moment matching

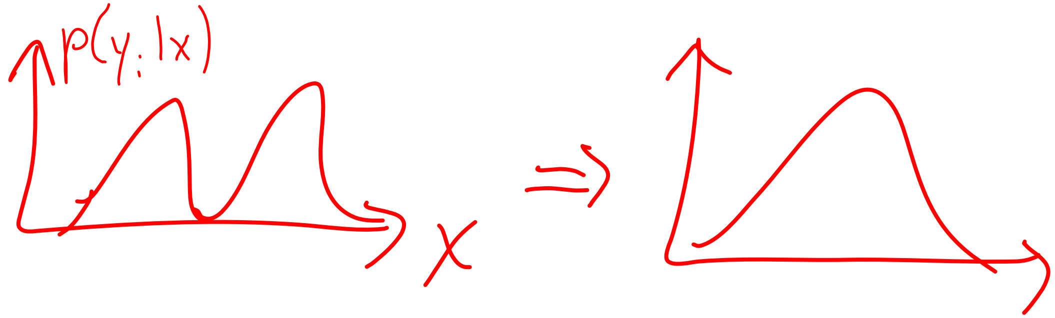


Strategy

- Approximate *each* factor by a Gaussian in x

$$
\begin{aligned}
p(y_i|x) &= (0.5)\mathcal{N}(y_i; x, 1) + (0.5)\mathcal{N}(y_i; 0, 10) \\
&\approx \mathcal{N}(x; m_i, v_i)
\end{aligned}
$$





Approximating a single factor

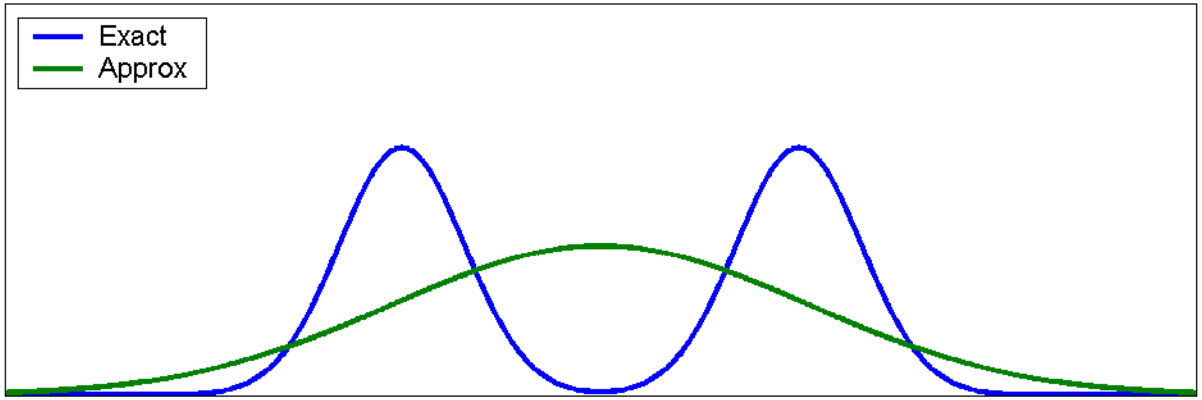
$p(y_i \mid x)$

$$E[x] = \frac{\int x \, p(y_i \mid x) \, dx}{\int p(y_i \mid x) \, dx}$$

$\Rightarrow$

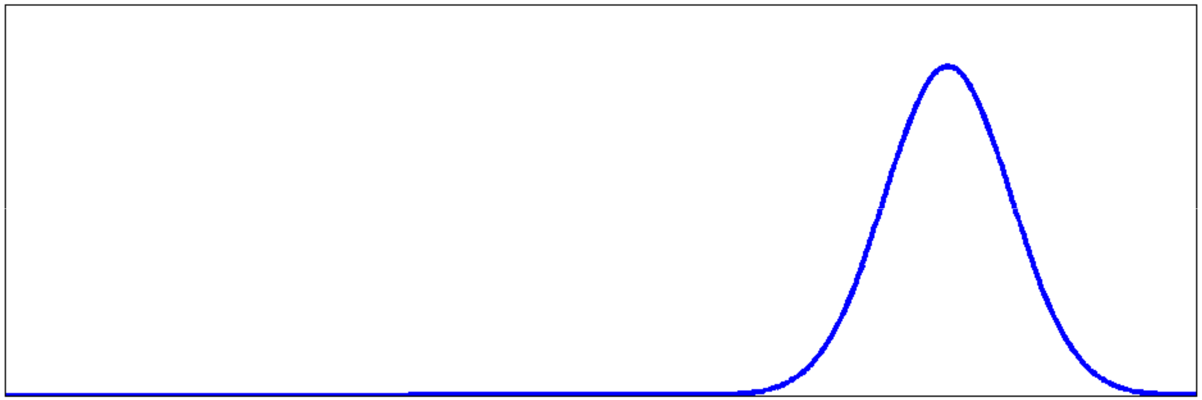
$N(x; m_i, v_i)$

(naïve)



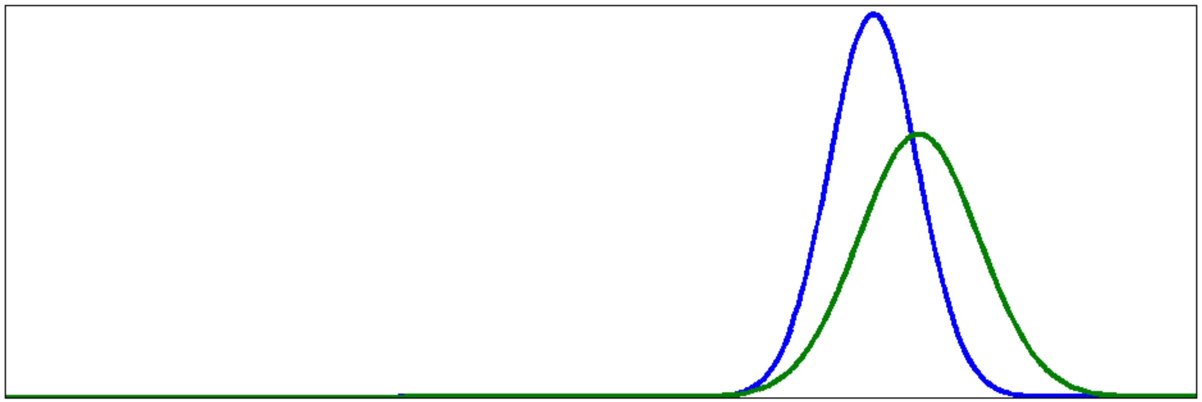
$f_i(x)$

$\times$



$q^{\setminus i}(x)$

$=$



$p(x)$

(informed)

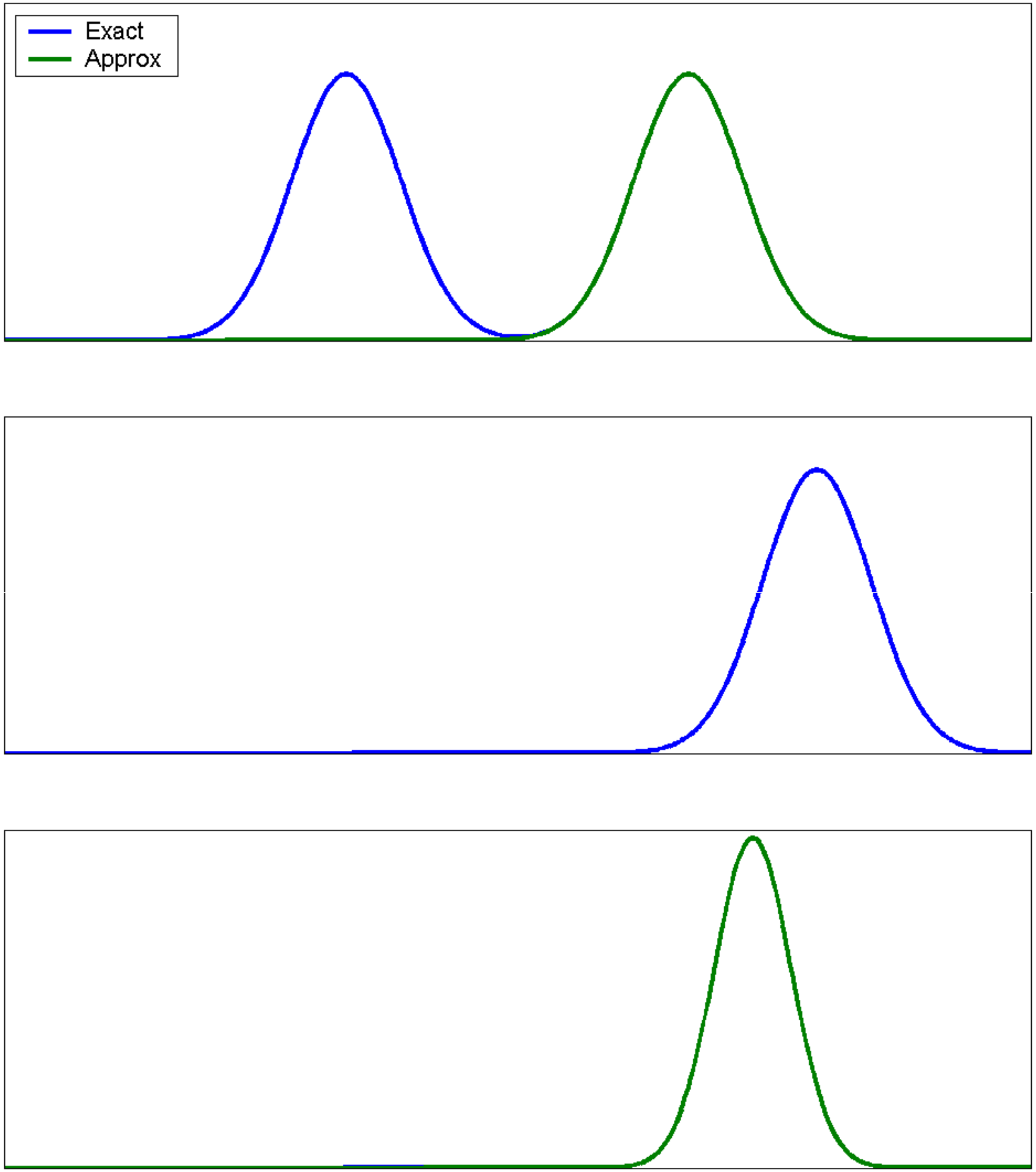
$f_i(x)$

$\times$

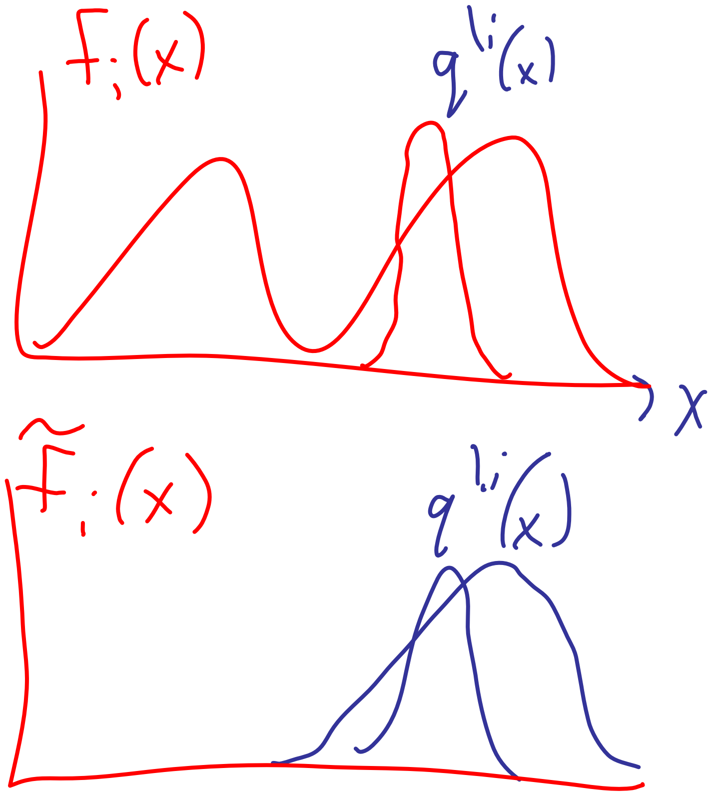
$q^{\setminus i}(x)$

$=$

$p(x)$



Single factor with Gaussian context



$$f_i(x)\, q^{\backslash i}(x) \approx \tilde{f}_i(x)\, q^{\backslash i}(x)$$

$$\mathrm{proj}[f_i(x)\, q^{\backslash i}(x)] = \tilde{f}_i(x)\, q^{\backslash i}(x)$$

$$\mathrm{proj}[p(x)] = \tilde{p}(x)$$

Same moments

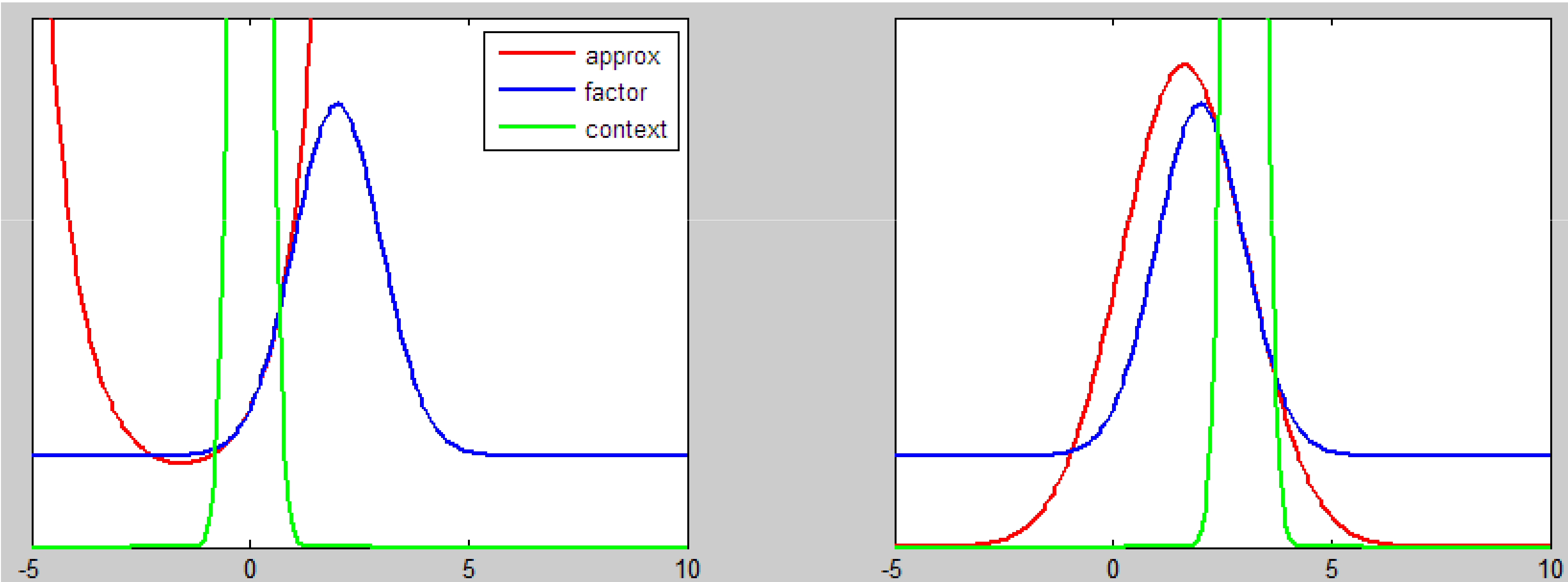
$$\tilde{f}_i(x) = \frac{\mathrm{proj}[f_i(x)\, q^{\backslash i}(x)]}{q^{\backslash i}(x)}$$

Gaussian multiplication formula

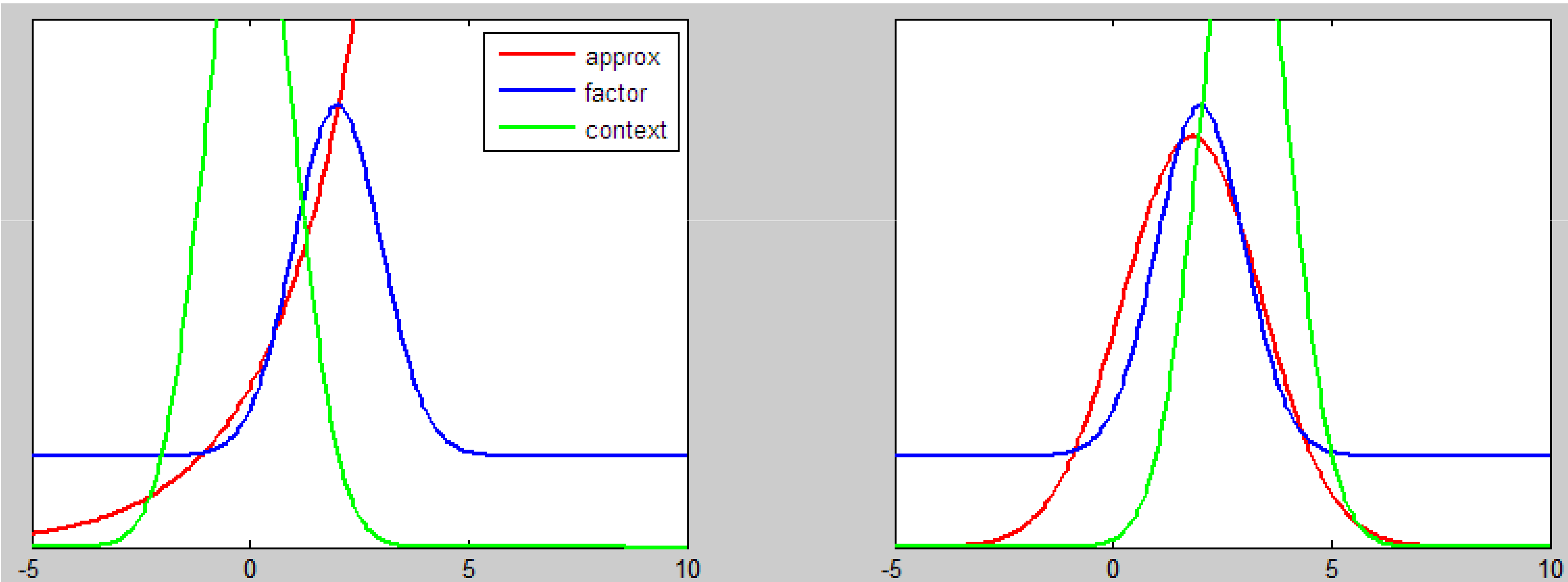
$$\begin{aligned}
\mathcal{N}(x; m_1, v_1)\mathcal{N}(x; m_2, v_2) &= \mathcal{N}(m_1; m_2, v_1 + v_2)\mathcal{N}(x; m, v) \\
\text{where } v &= \frac{1}{\frac{1}{v_1} + \frac{1}{v_2}} \\
m &= v\left(\frac{m_1}{v_1} + \frac{m_2}{v_2}\right)
\end{aligned}$$

$$\begin{aligned}
\mathcal{N}(x; m_1, v_1)/\mathcal{N}(x; m_2, v_2) &= \frac{v_2\mathcal{N}(x; m, v)}{(v_2 - v_1)\mathcal{N}(m_1; m_2, v_2 - v_1)} \\
\text{where } v &= \frac{1}{\frac{1}{v_1} - \frac{1}{v_2}} \\
m &= v\left(\frac{m_1}{v_1} - \frac{m_2}{v_2}\right)
\end{aligned}$$

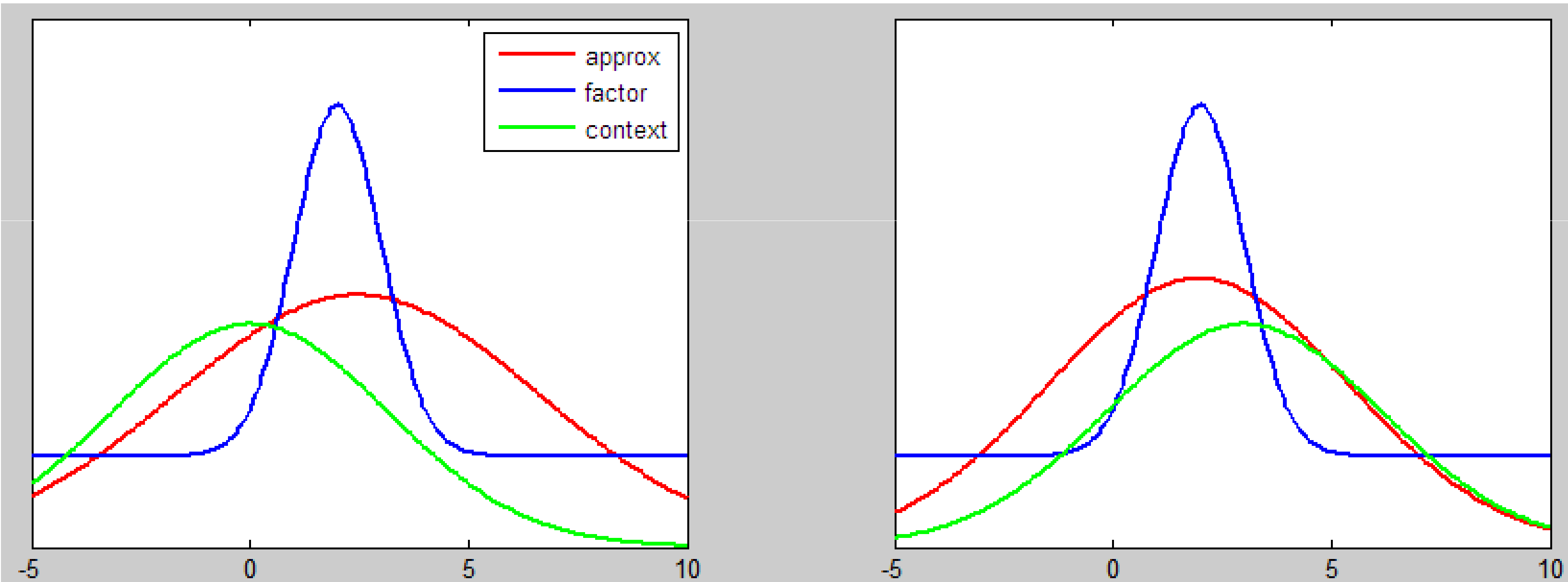
Approximation with narrow context



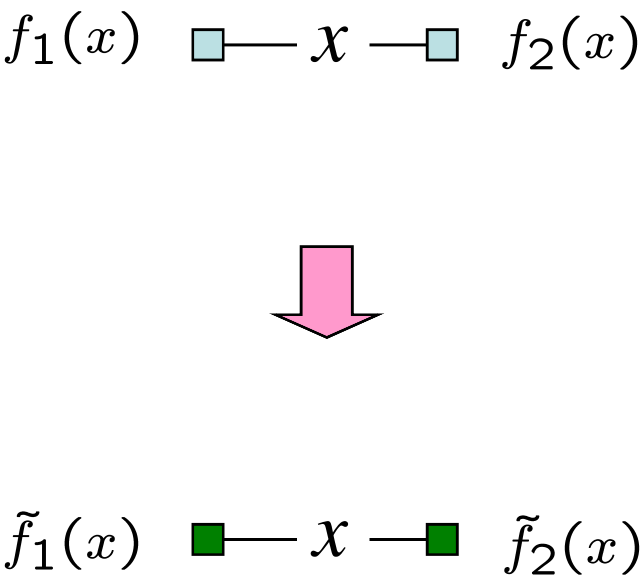
Approximation with medium context



Approximation with wide context



Two factors



$$\tilde{f}_1(x) = \frac{\text{proj}[f_1(x)\tilde{f}_2(x)]}{\tilde{f}_2(x)}$$

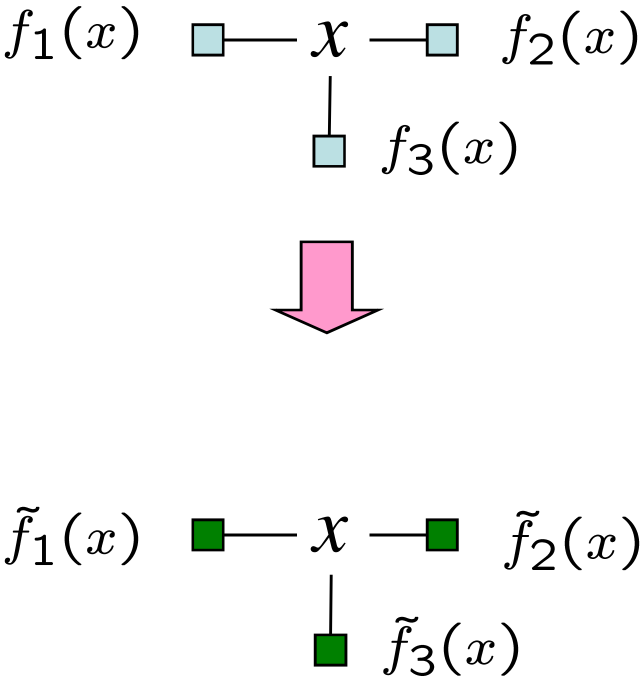
$$\tilde{f}_2(x) = \frac{\text{proj}[f_2(x)\tilde{f}_1(x)]}{\tilde{f}_1(x)}$$

$$\tilde{f}_1(x)\tilde{f}_2(x) = \text{proj}[f_2(x)\tilde{f}_1(x)]$$

Message passing

$$\tilde{f}_1(x)\tilde{f}_2(x) = \text{proj}[f_1(x)f_2(x)]$$

Three factors



$$\tilde{f}_1(x) = \frac{\mathrm{proj}[f_1(x)\tilde{f}_2(x)\tilde{f}_3(x)]}{\tilde{f}_2(x)\tilde{f}_3(x)}$$

$$q^{\backslash 1}(x) = \tilde{f}_2(x)\tilde{f}_3(x)$$

$$\tilde{f}_1(x) = \frac{\mathrm{proj}[f_1(x)q^{\backslash 1}(x)]}{q^{\backslash 1}(x)}$$

Message passing

Message Passing = Distributed Optimization

$q(x) = \tilde{f}_1(x)\,\tilde{f}_2(x)\,\tilde{f}_3(x)$

- Messages represent a simpler distribution $q(x)$ that approximates $p(x)$
  - A *distributed* representation
- Message passing = optimizing q to fit p
  - q stands in for p when answering queries
- Choices:
  - What type of distribution to construct (approximating family)
  - What cost to minimize (divergence measure)

Distributed divergence minimization

- Write p as product of factors:

$$p(x) = \prod_a f_a(x)$$

- Approximate factors one by one:

$$f_a(x) \to \tilde{f}_a(x)$$

- Multiply to get the approximation:

$$q(x) = \prod_a \tilde{f}_a(x)$$

Global divergence to local divergence

- Global divergence:

$$D(p(x) \mid\mid q(x)) =$$

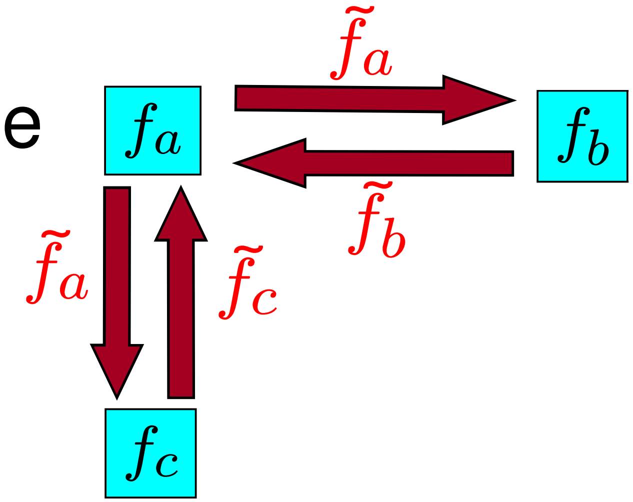
$$D(f_a(x) \prod_{b \neq a} f_b(x) \mid\mid \tilde{f}_a(x) \prod_{b \neq a} \tilde{f}_b(x))$$

- Local divergence:

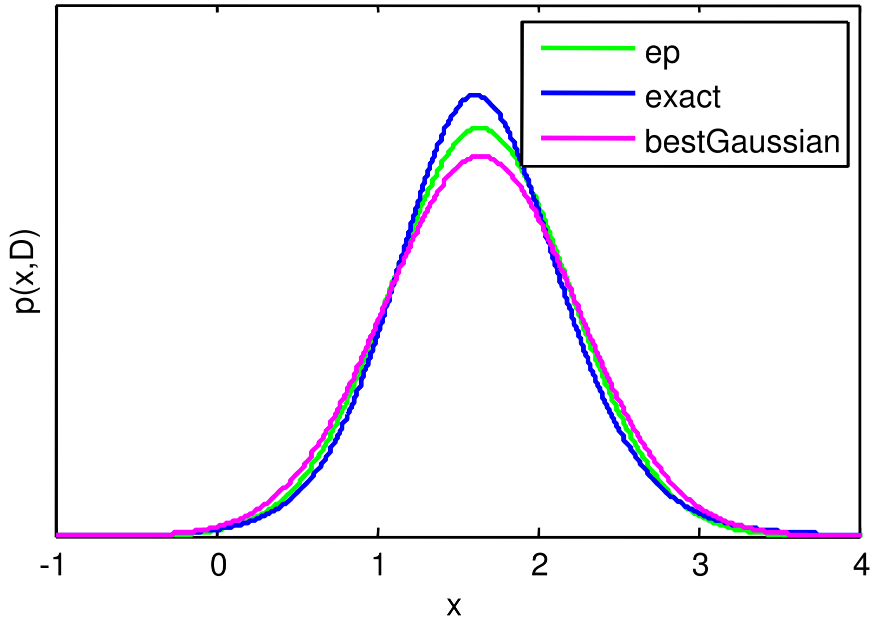
$$D(f_a(x) \prod_{b \neq a} \tilde{f}_b(x) \mid\mid \tilde{f}_a(x) \prod_{b \neq a} \tilde{f}_b(x))$$

Message passing

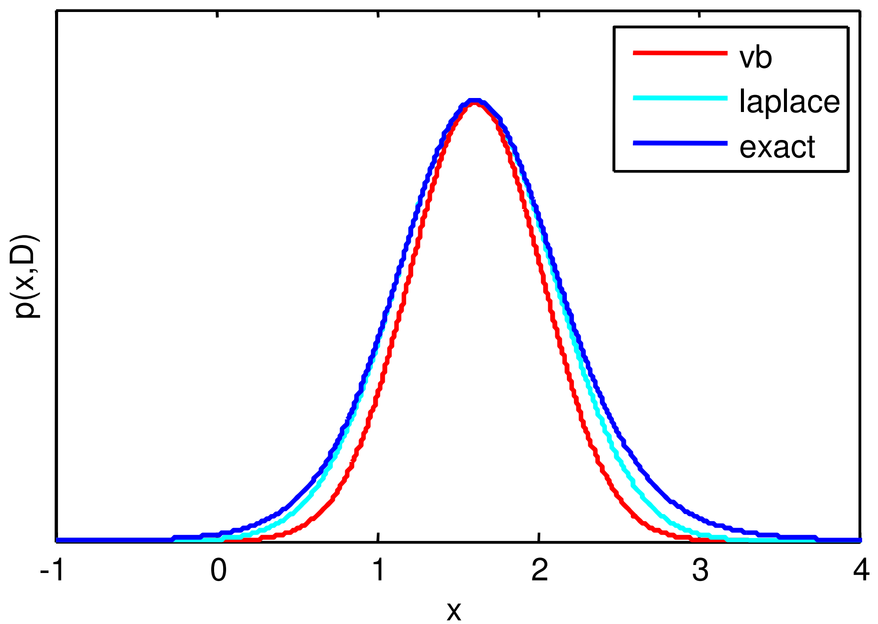
- Messages are passed between *factors*
- Messages are factor approximations: $\tilde{f}_a(x)$
- Factor a receives $\tilde{f}_b(x), b \neq a$
  - Minimize local divergence to get $\tilde{f}_a(x)$
  - Send to other factors
  - Repeat until convergence



Gaussian found by EP



Other methods



Accuracy

Posterior mean:

exact = 1.64864

ep = 1.64514

laplace = 1.61946

vb = 1.61834

Posterior variance:

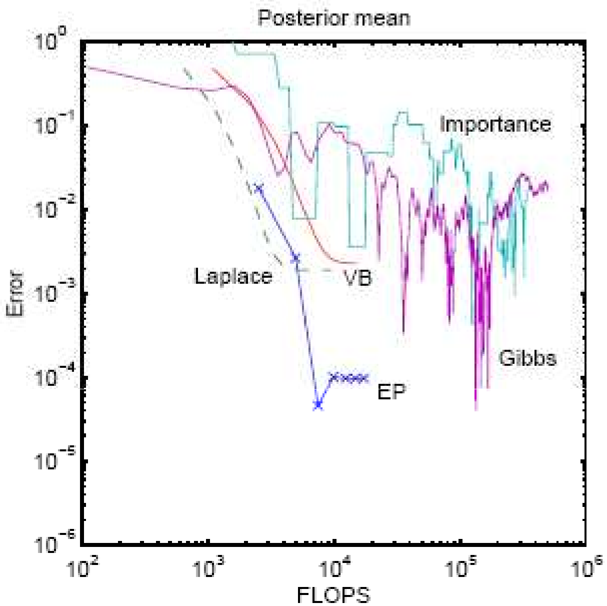
exact = 0.359673

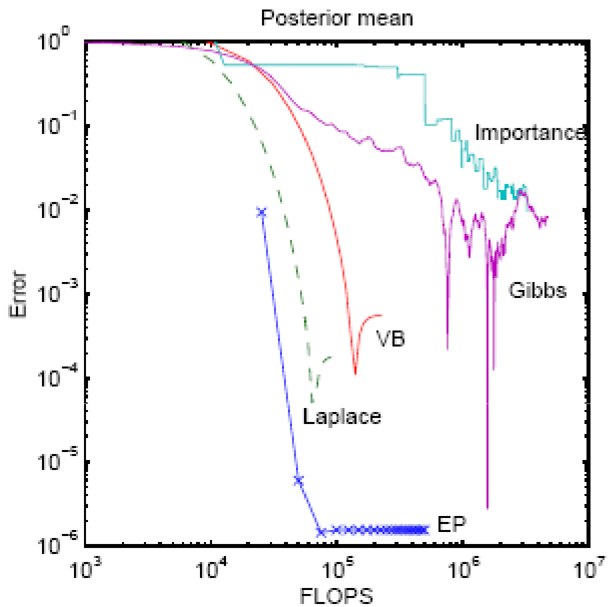
ep = 0.311474

laplace = 0.234616

vb = 0.171155

Cost vs. accuracy





20 points

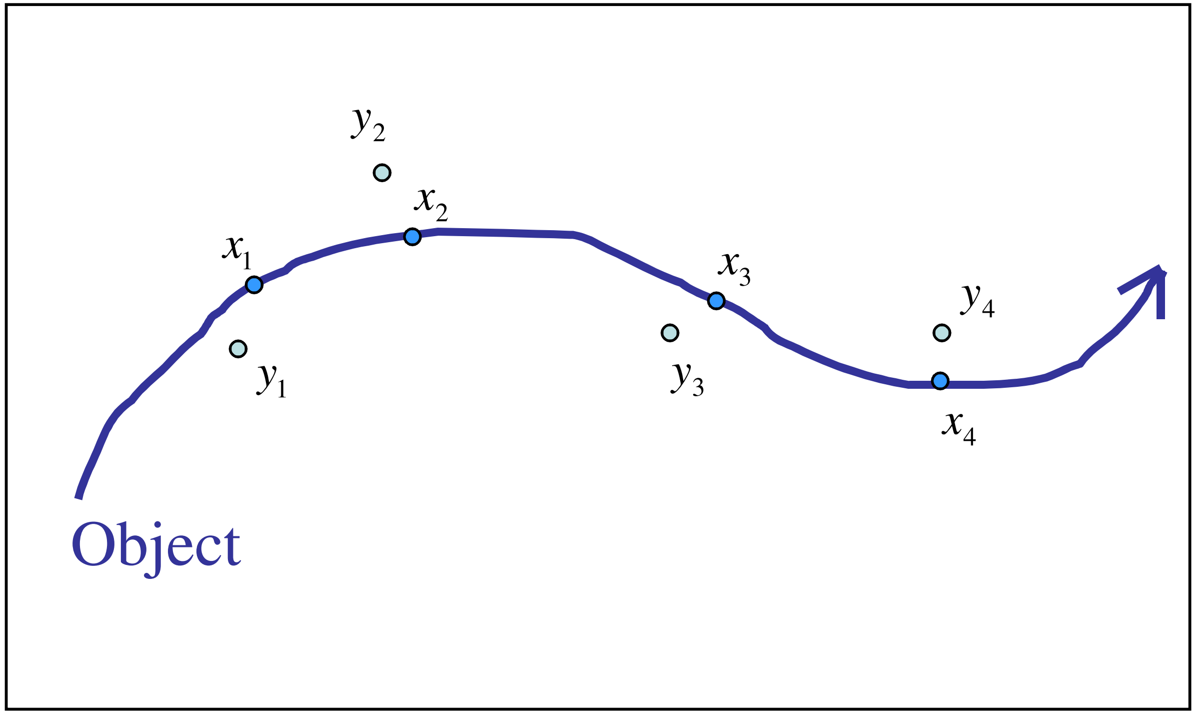
200 points

Deterministic methods improve with more data (posterior is more Gaussian)
Sampling methods do not

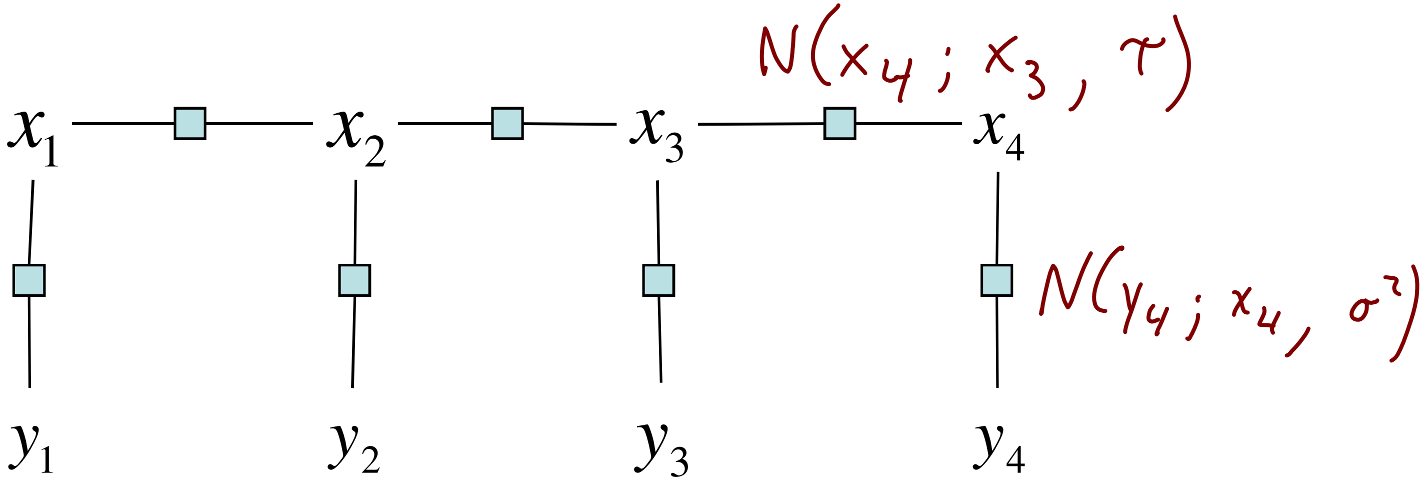
Time series problems

Example: Tracking

Guess the position of an object given noisy measurements



Factor graph

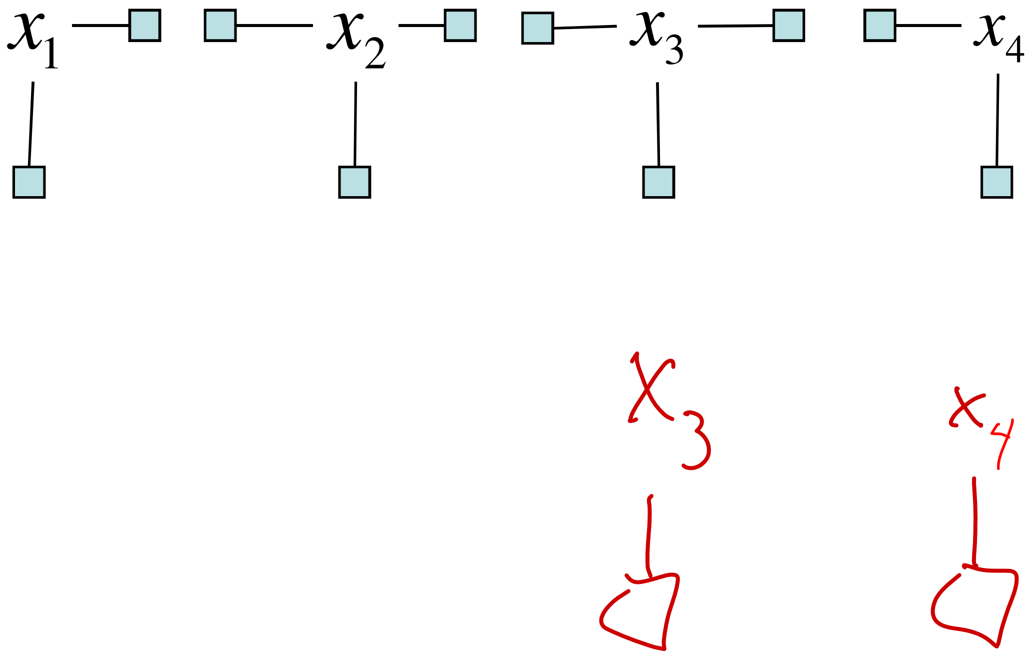


e.g. $\quad x_t = x_{t-1} + v_t \quad$ (random walk)

$$y_t = x_t + \text{noise}$$

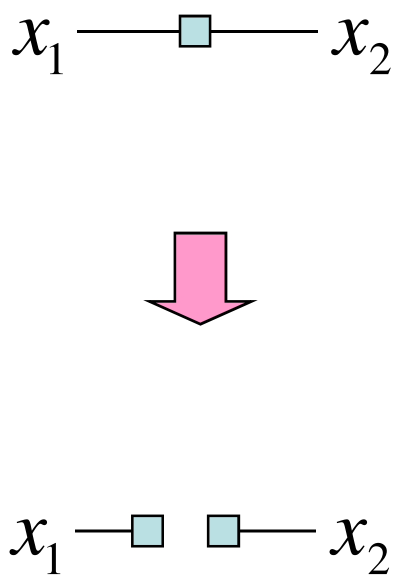
want distribution of x's given y's

Approximate factor graph



Splitting a pairwise factor

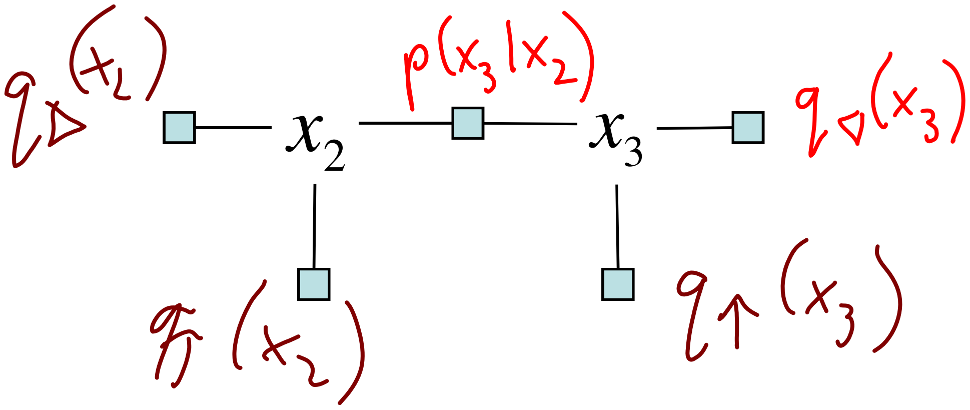
$N(x_2; x_1, \tau)$



$\mathbb{E}[x_1]$ $E[x_2]$

$\mathrm{var}(x_1)$ $\mathrm{var}(x_2)$

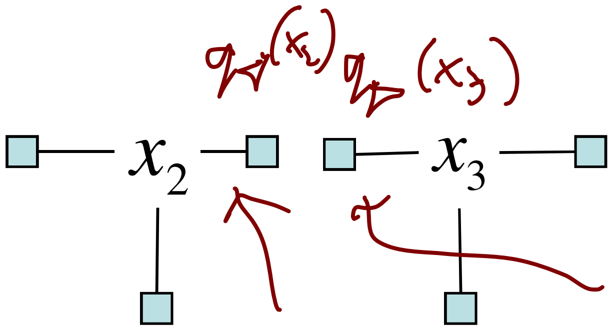
Splitting in context



$$q_\triangleleft(x_2) = \frac{\mathrm{proj}\left[q_\triangleright(x_2)\, q_\uparrow(x_2) \cdot \int p(x_3|x_2)\, q_\triangleleft(x_3)\, q_\uparrow(x_3)\, dx_3 \right]}{q_\triangleright(x_2)\, q_\uparrow(x_2)}$$

$$= \int p(x_3|x_2)\, q^{\setminus i}(x_3)$$

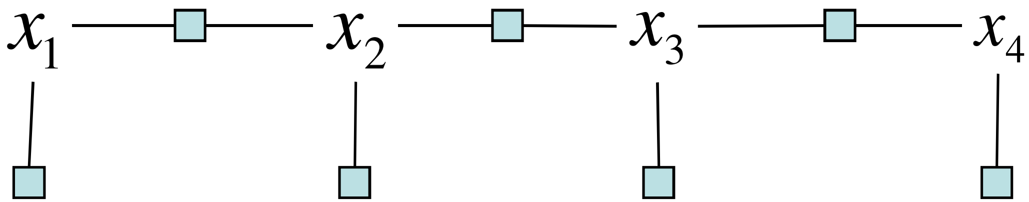
$$q^{\setminus i}(x_3) = q_\triangleleft(x_3)\, q_\uparrow(x_3)$$



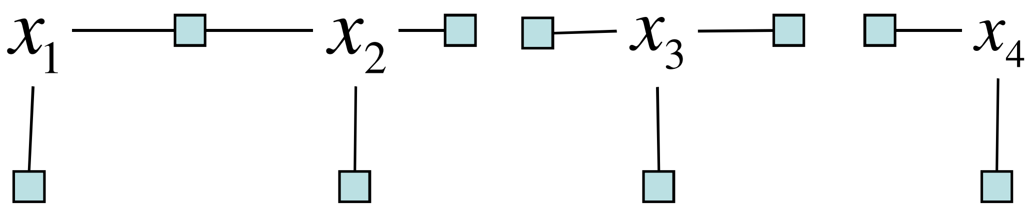
$\int_{x_2} p(x_3|x_2)\, q^{\setminus i}(x_2)$ forward message

backward message $\int_{x_3} p(x_3|x_2)\, q^{\setminus i}(x_3)$

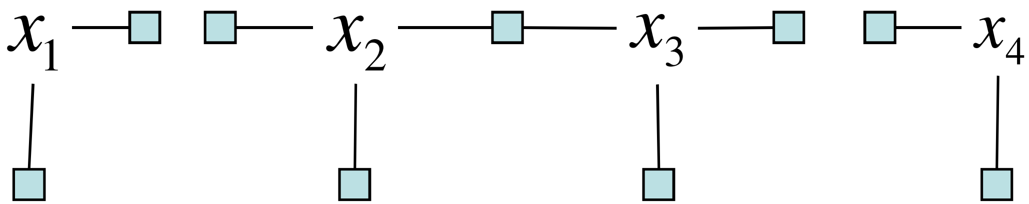
Sweeping through the graph



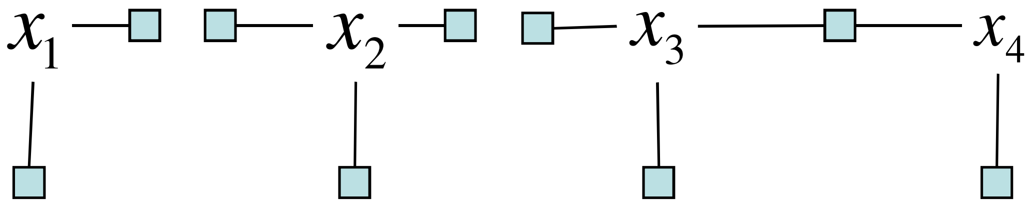
Sweeping through the graph



Sweeping through the graph

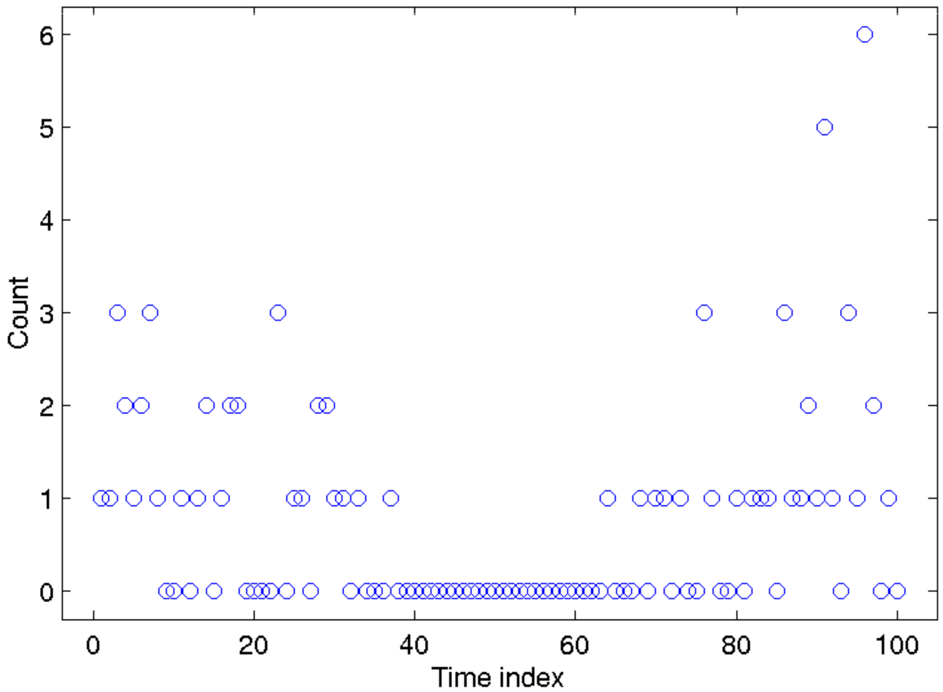


Sweeping through the graph



Example: Poisson tracking

- y_t is a Poisson-distributed integer with mean $\exp(x_t)$



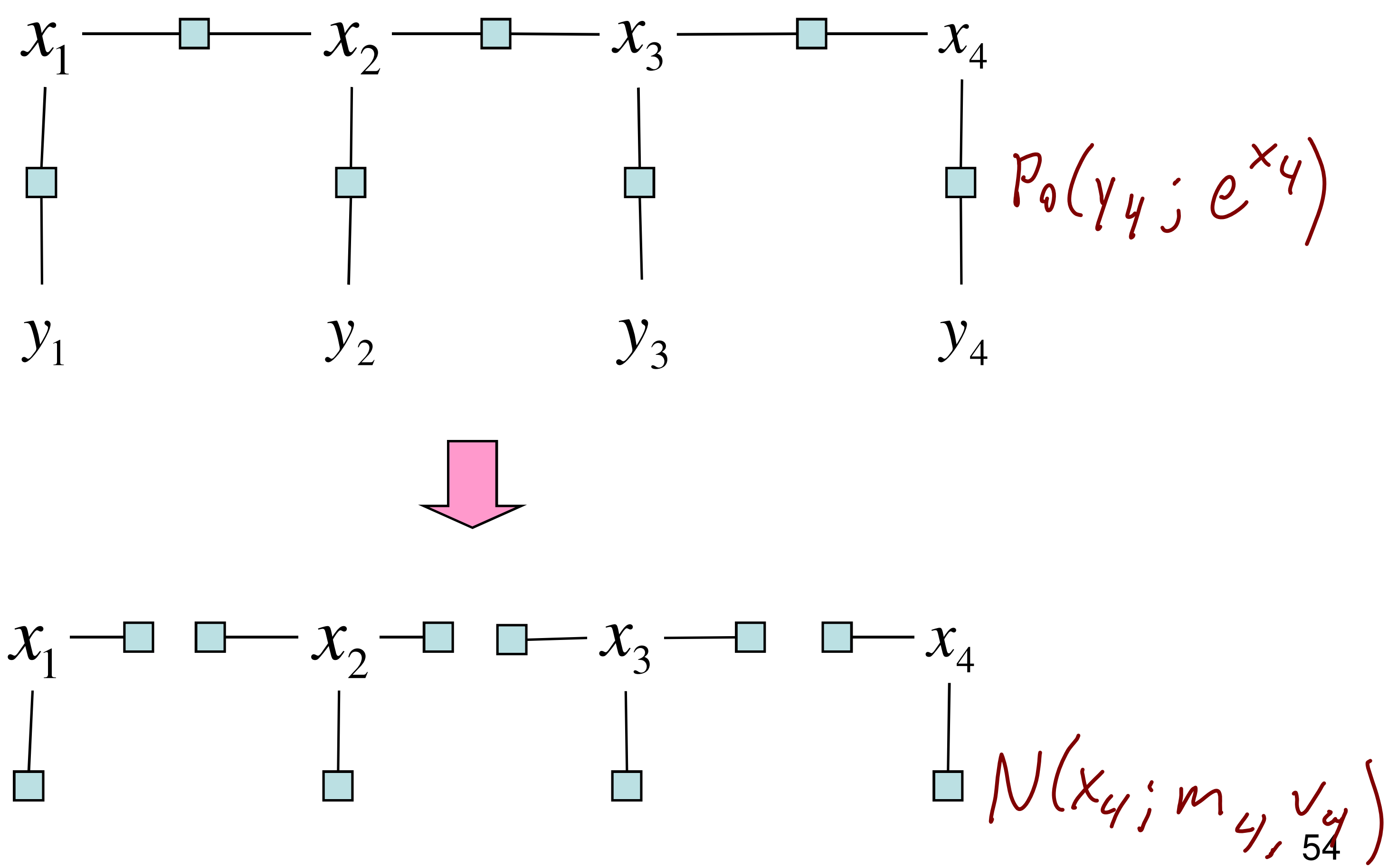
Poisson tracking model

$$p(x_1) \sim N(0,100)$$

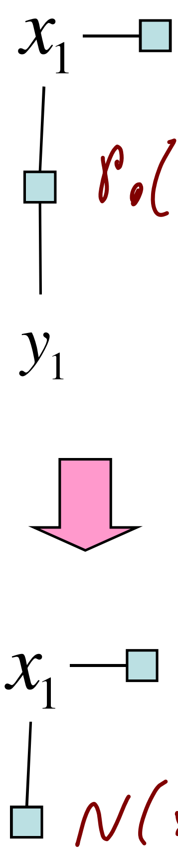
$$p(x_t \mid x_{t-1}) \sim N(x_{t-1}, 0.01)$$

$$p(y_t \mid x_t) = \exp(y_t x_t - e^{x_t}) / y_t!$$

Factor graph



Approximating a measurement factor



$q_{\backslash}(x_1)$

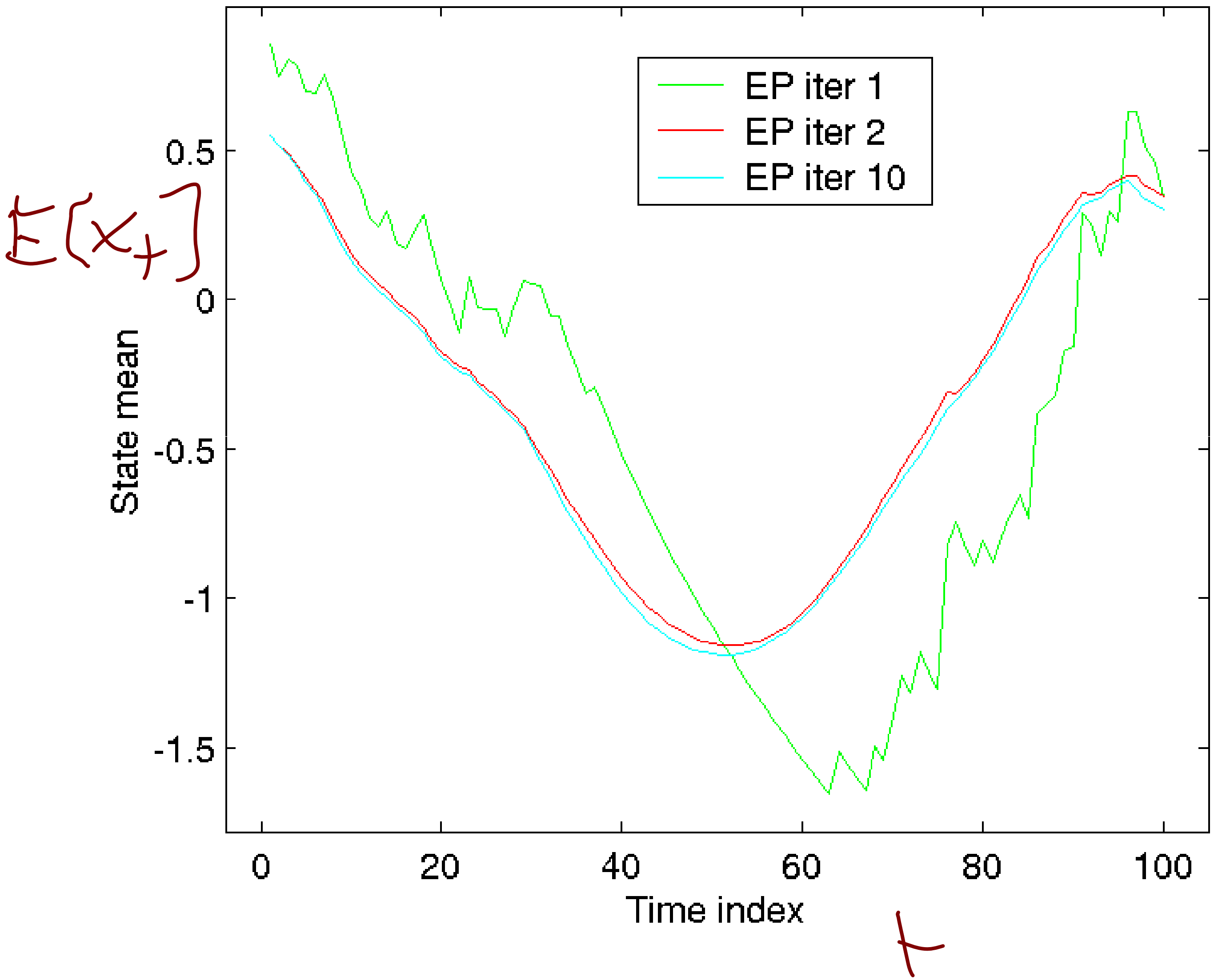
$p_0(y_1; e^{x_1})$

$$N(x_1; m_1, v_1) = \frac{\mathrm{proj}\left[p(y_1 | x_1)\, q_{\backslash}(x_1)\right]}{q_{\backslash}(x_1)}$$

$q_{\backslash}(x_1)$

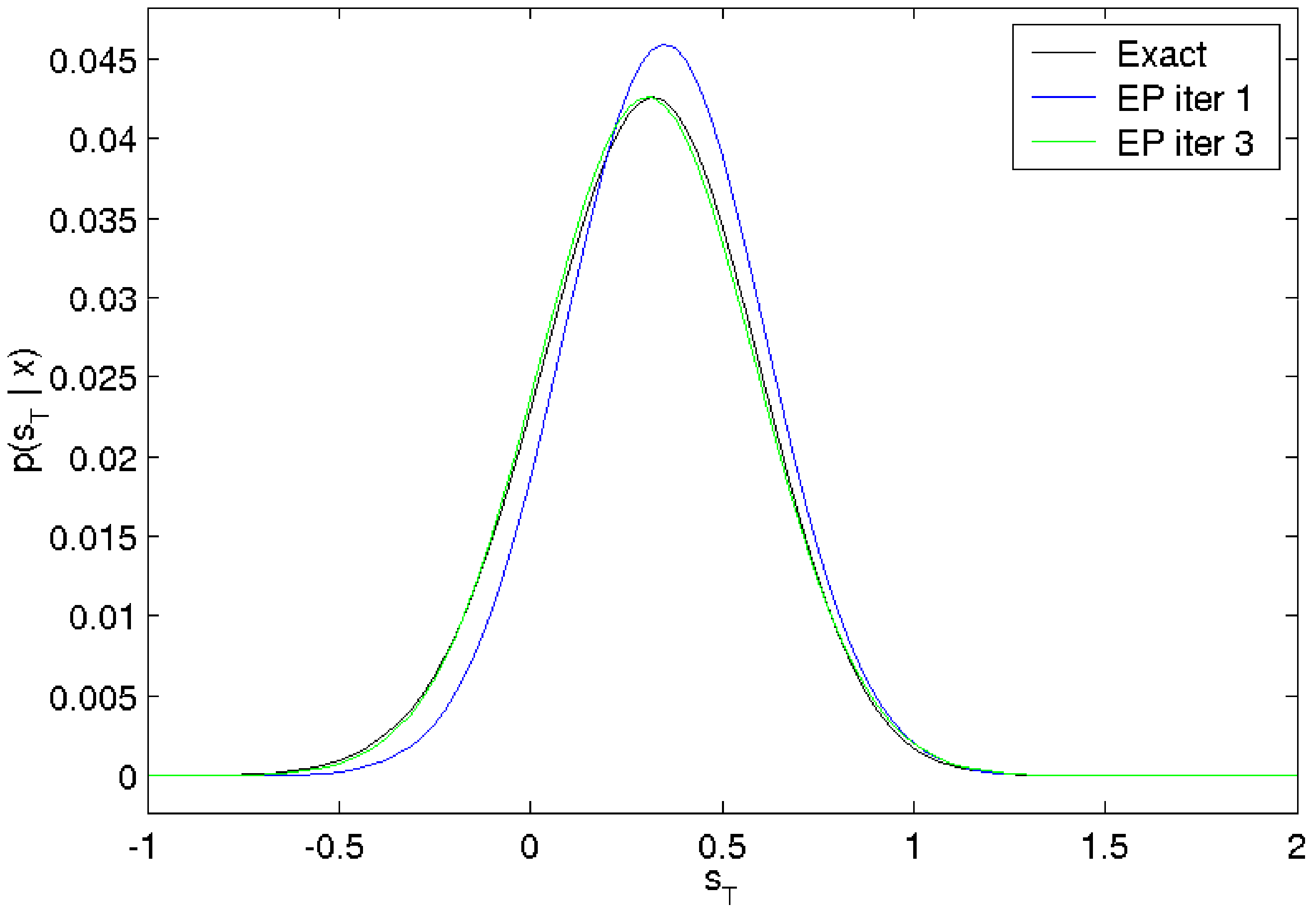
$N(x_1; m_1, v_1)$

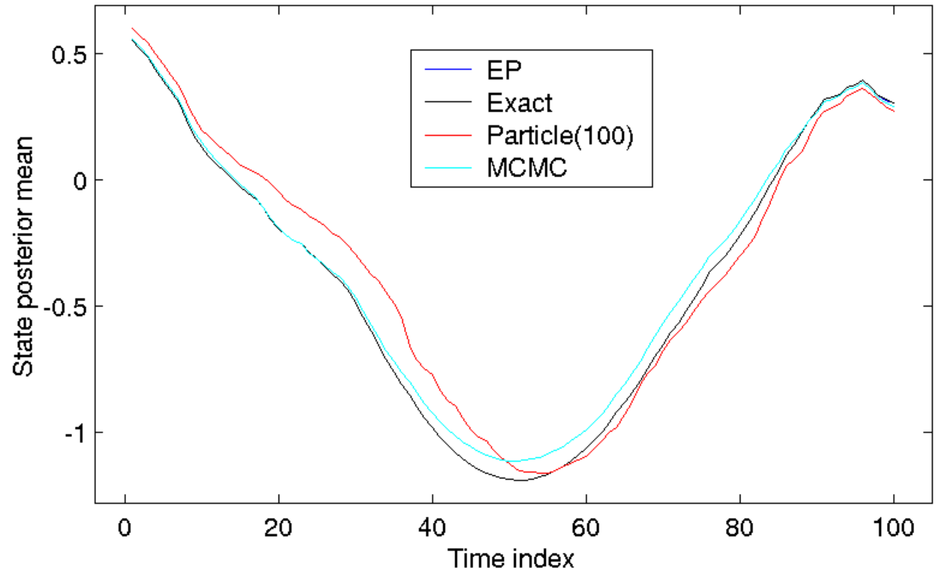
$$N(x_1; m_1, v_1)\, q_{\backslash}(x_1) = \mathrm{proj}\left[p_0(y_1; e^{x_1})\, q_{\backslash}(x_1)\right]$$


EP iter 1
EP iter 2
EP iter 10
State mean
Time index
0.5
0
-0.5
-1
-1.5
0
20
40
60
80
100
E[x_t]
t

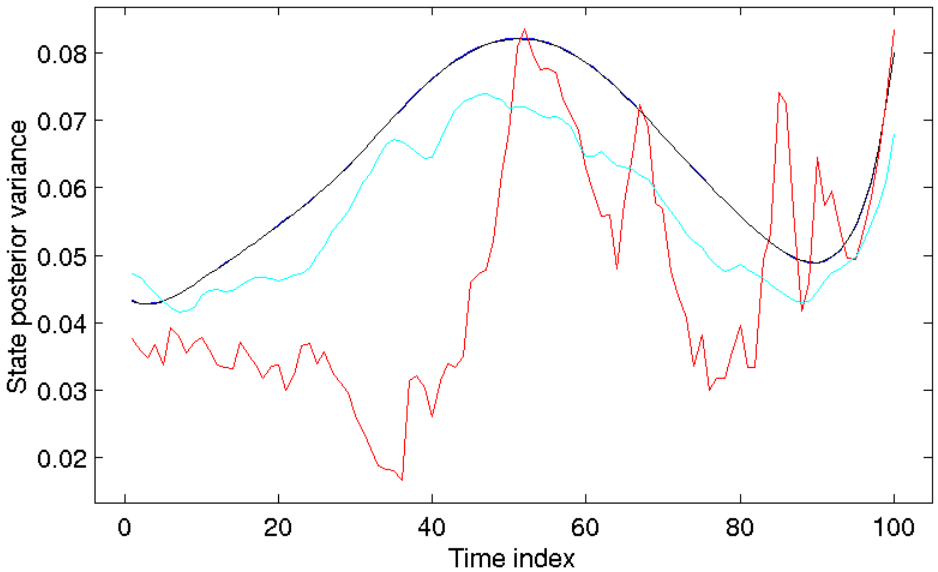
Posterior for the last state

x_T

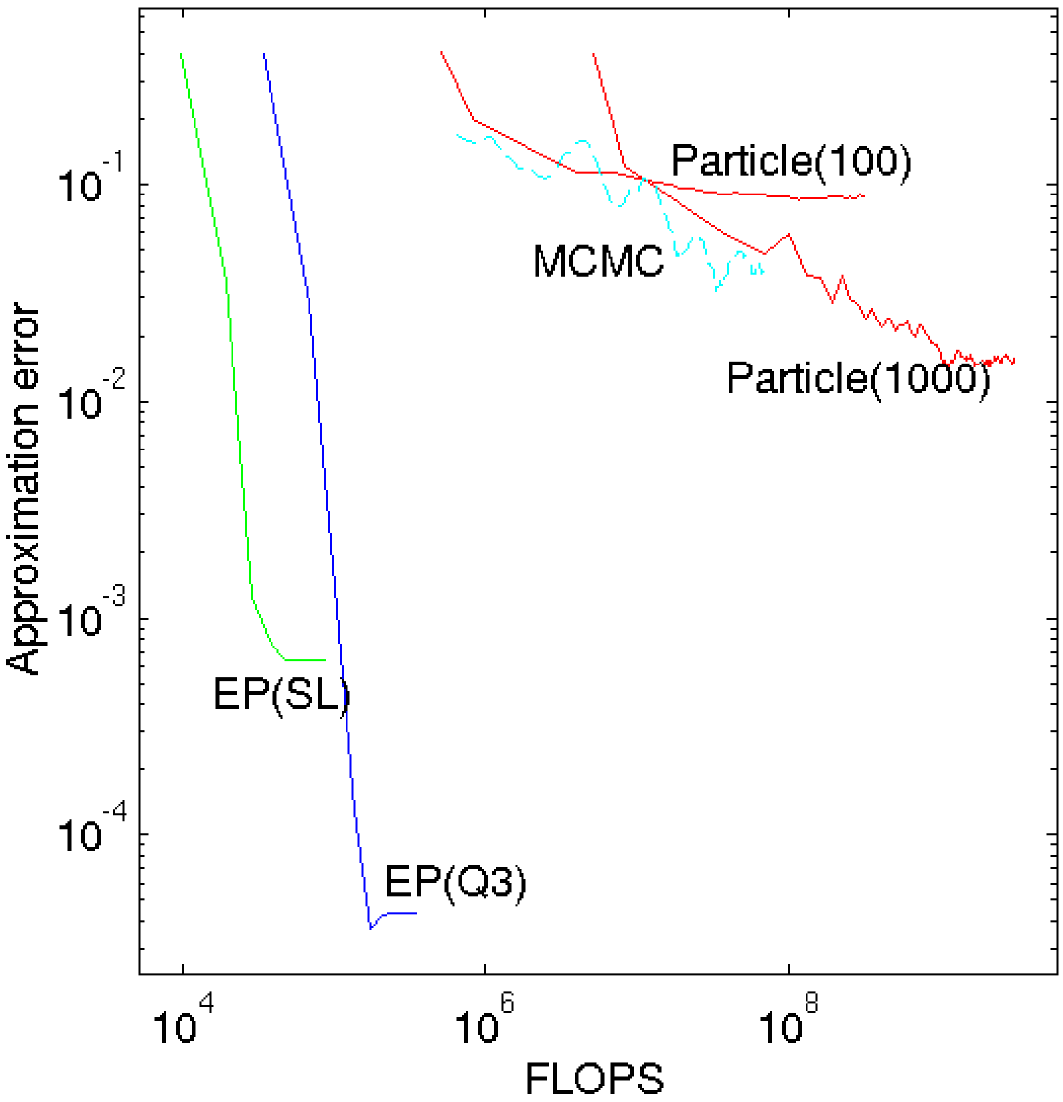




State posterior mean
Time index
EP
Exact
Particle(100)
MCMC
0.5
0
-0.5
-1
0
20
40
60
80
100

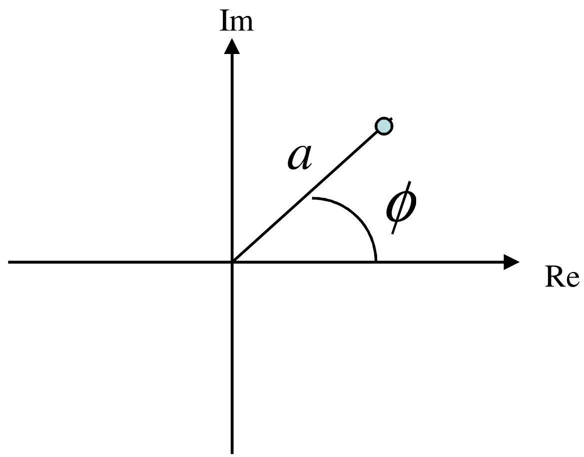


State posterior variance
Time index
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0
20
40
60
80
100


Particle(100)
MCMC
Particle(1000)
EP(SL)
EP(Q3)
Approximation error
FLOPS

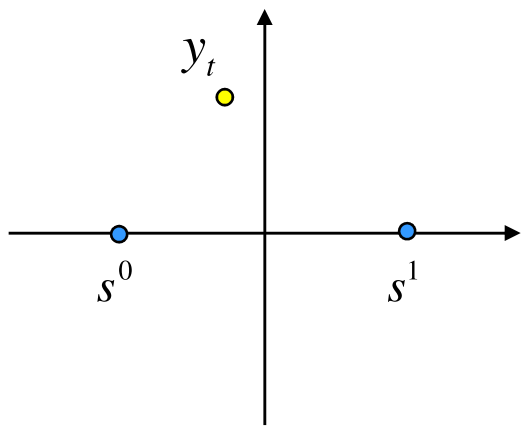
EP for signal detection

- Wireless communication problem
- Transmitted signal = $a \sin(\omega t + \phi)$
- (a, ϕ) vary to encode each symbol
- In complex numbers: $ae^{i\phi}$



Binary symbols, Gaussian noise

- Symbols are 1 and –1 (in complex plane)
- Received signal = $a \sin(\omega t + \phi) + \text{noise}$
- Recovered $\hat{a} e^{\hat{\phi}} = a e^{\phi} + \text{noise} = y_t$
- Optimal detection is easy in this case

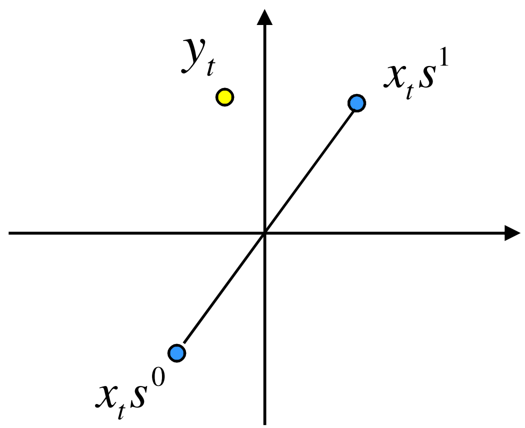


Fading channel

- Channel systematically changes amplitude and phase:

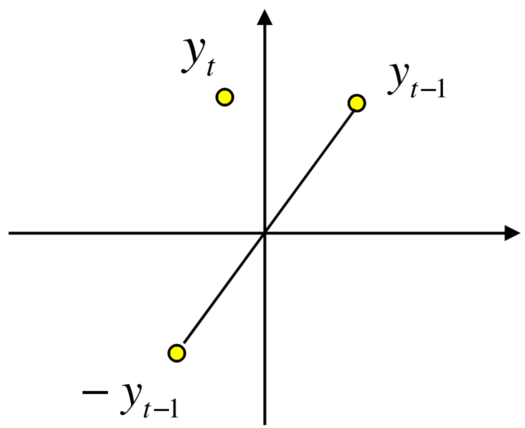
$$y_t = x_t s + \text{noise}$$

- x_t changes over time



Differential detection

- Use last measurement to estimate state
- Binary symbols only
- No smoothing of state = noisy



Factor graph

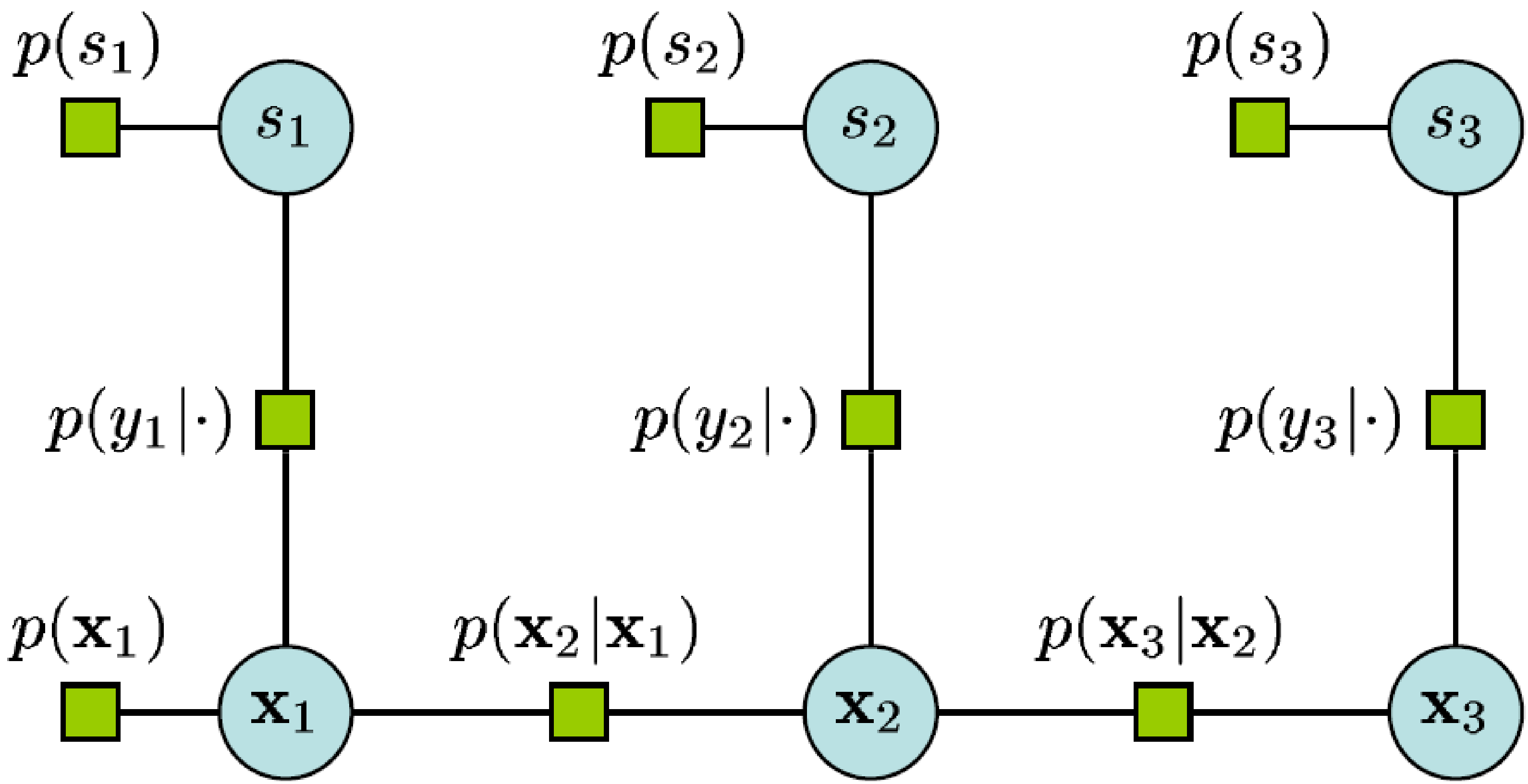
s_1 s_2 s_3 s_4

y_1 y_2 y_3 y_4

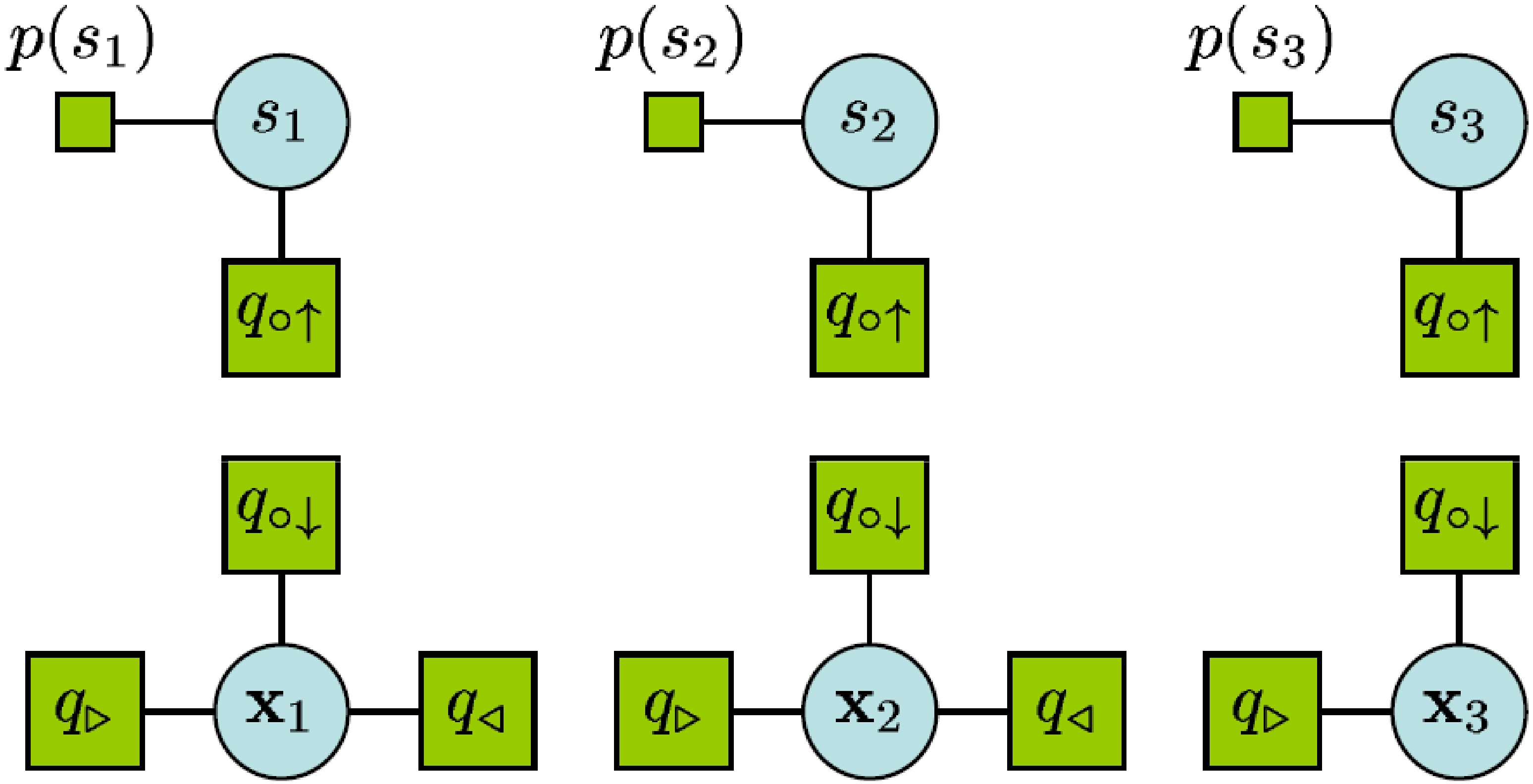
x_1 x_2 x_3 x_4

Symbols can also be correlated (e.g. error-correcting code)

Dynamics are learned from training data (all 1's)

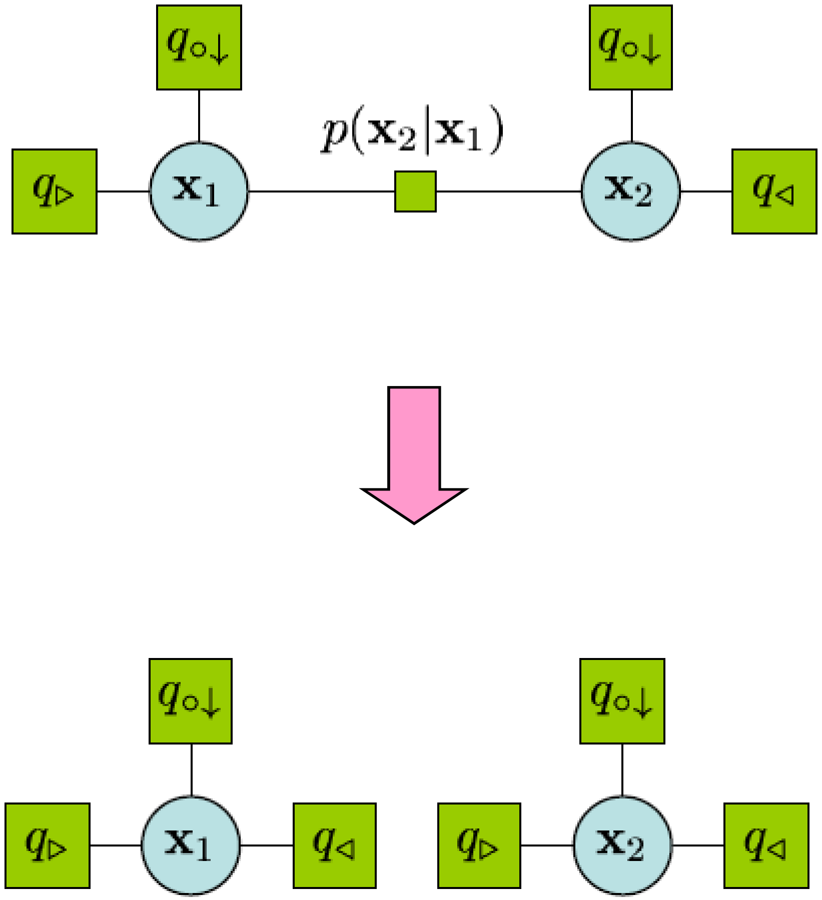


(a) Exact posterior $p(s_1, s_2, s_3, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$

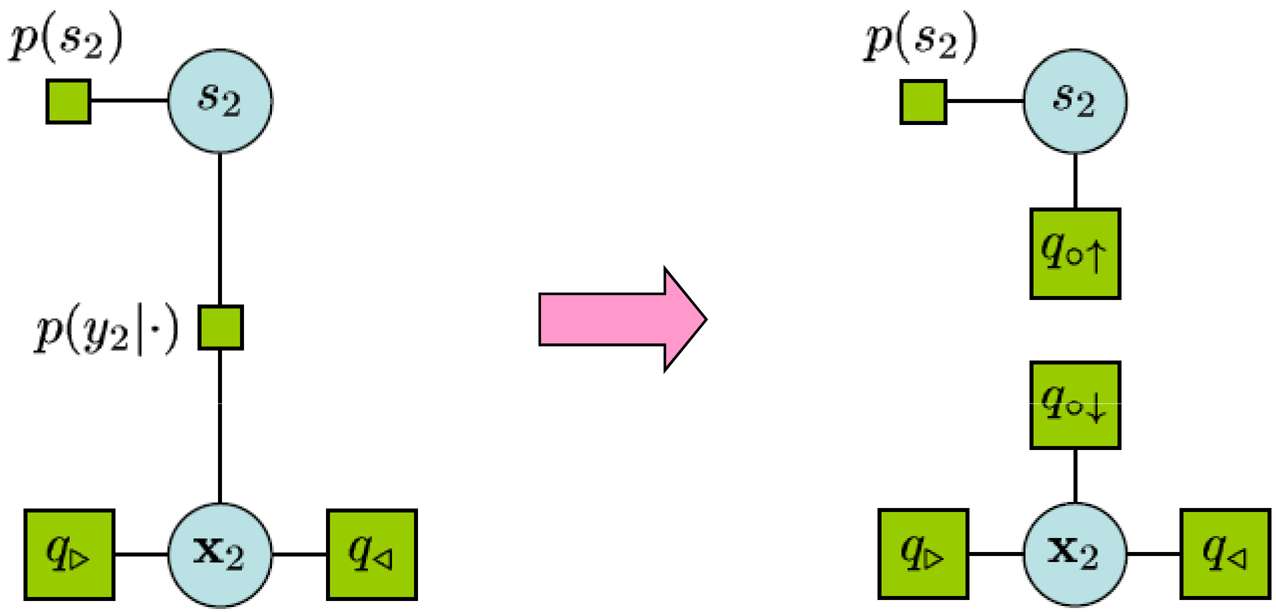


(b) Approximate posterior $\prod_i q(s_i)q(\mathbf{x}_i)$

Splitting a transition factor



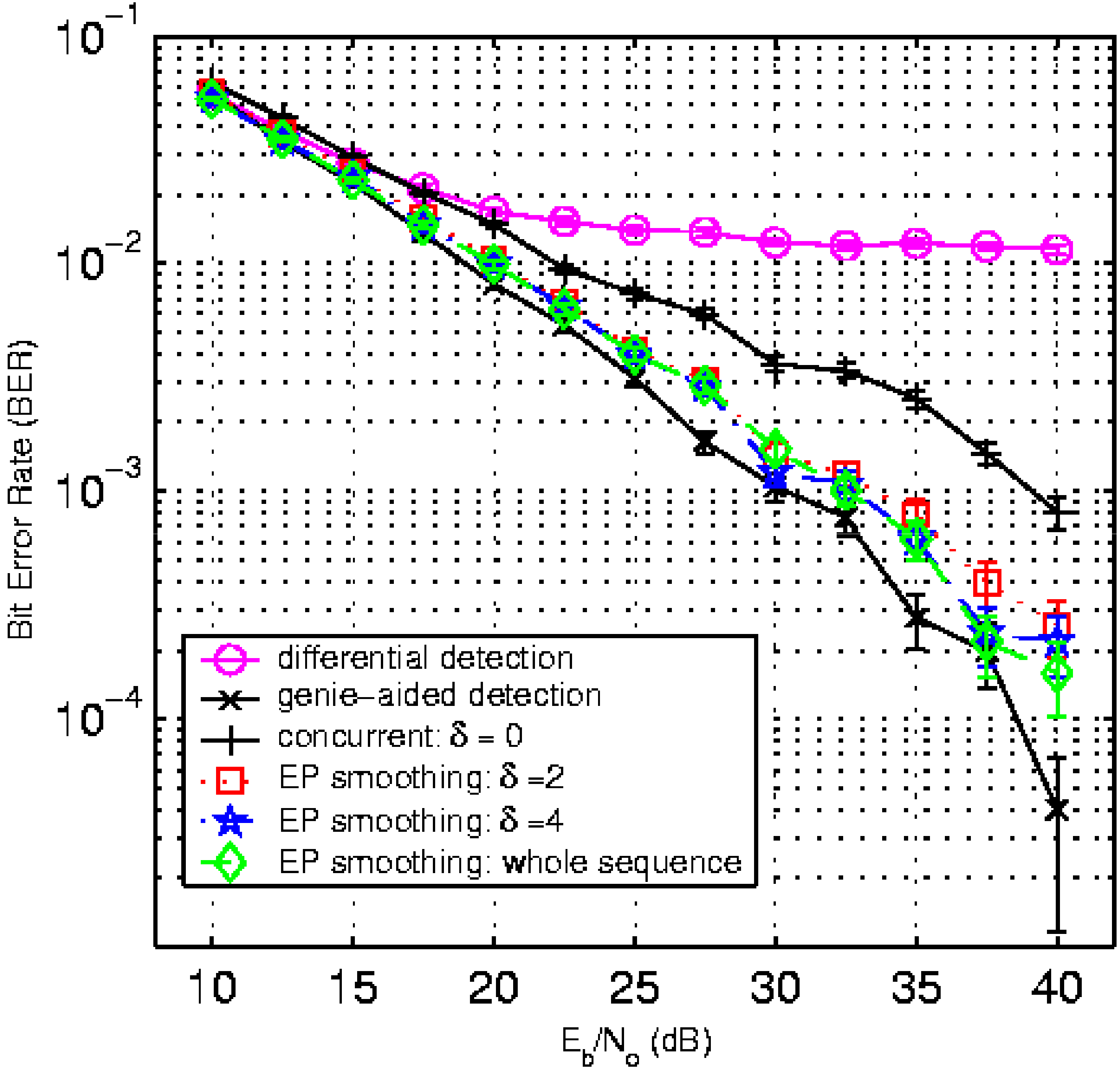
Splitting a measurement factor



On-line implementation

- Iterate over the last δ measurements
- Previous measurements act as prior

- Results comparable to particle filtering, but much faster



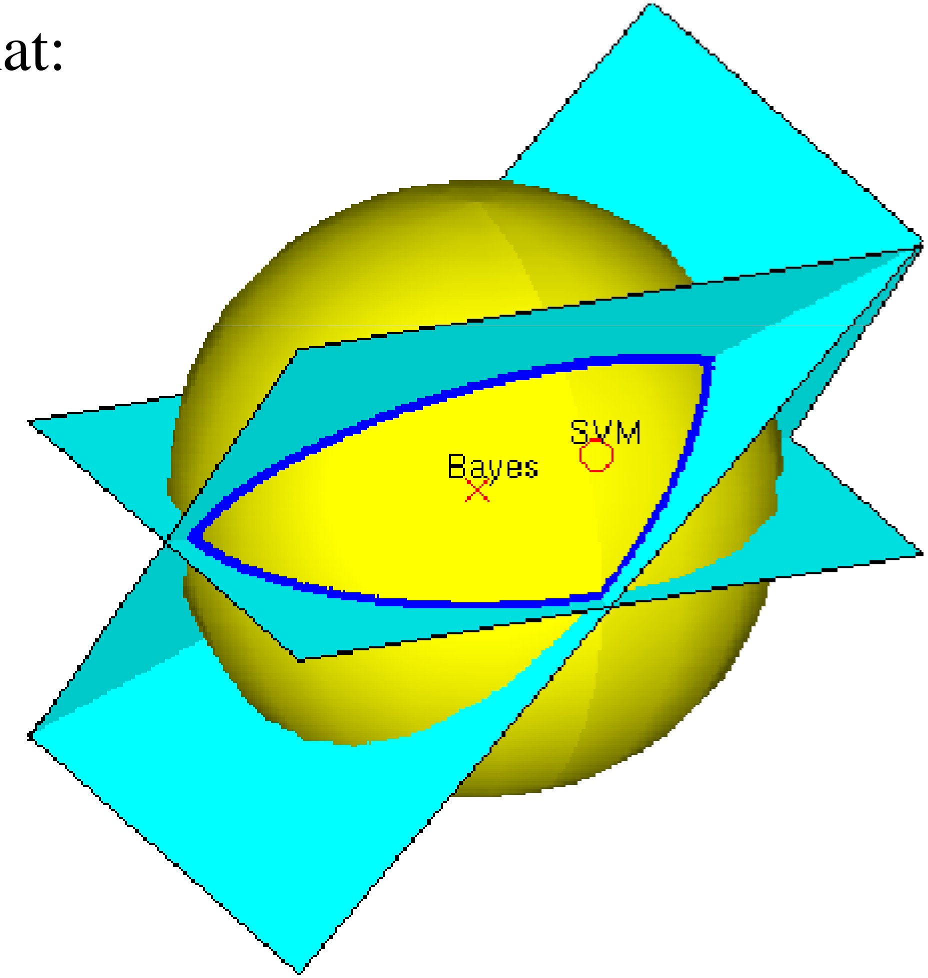
10^{-1}
10^{-2}
10^{-3}
10^{-4}
Bit Error Rate (BER)
10
15
20
25
30
35
40
E_b/N_o (dB)
differential detection
genie–aided detection
concurrent: δ = 0
EP smoothing: δ =2
EP smoothing: δ =4
EP smoothing: whole sequence

Linear separation revisited

Geometry of linear separation

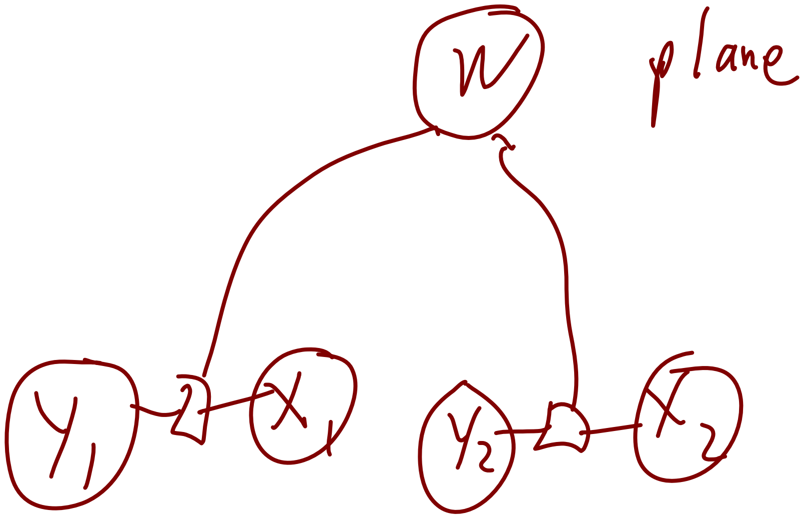
Separator is any vector w such that:

$$\mathbf{w}^T \mathbf{x}_i > 0 \quad \text{(class 1)}$$
$$\mathbf{w}^T \mathbf{x}_i < 0 \quad \text{(class 2)}$$
$$\|\mathbf{w}\| = 1 \quad \text{(sphere)}$$



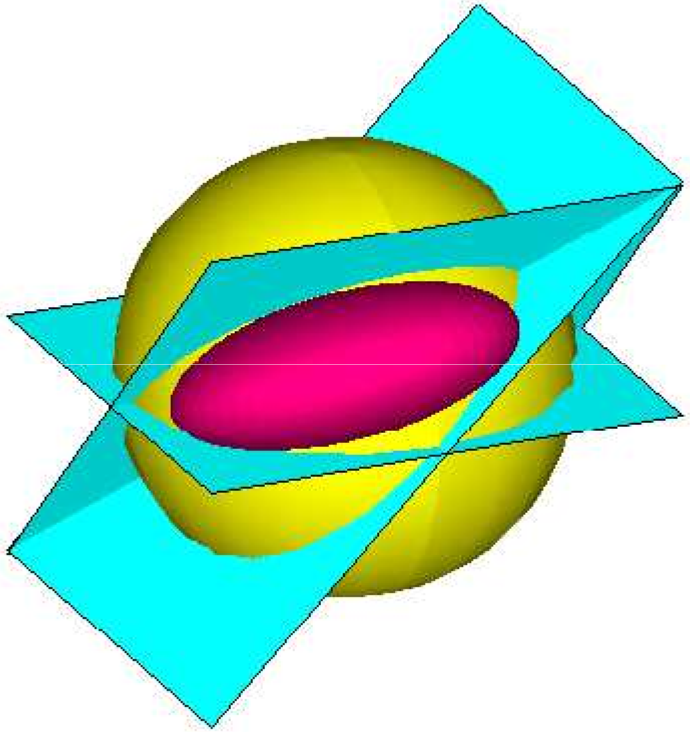
This set has an unusual shape
SVM: Optimize over it
Bayes: Average over it

Factor graph



$$p(y_i = \pm 1 \mid \mathbf{x}_i, \mathbf{w}) = I(y_i \mathbf{x}_i^\top \mathbf{w} > 0)$$

Performance on linear separation



EP Gaussian approximation to posterior

Time vs. accuracy

A typical run on the 3-point problem

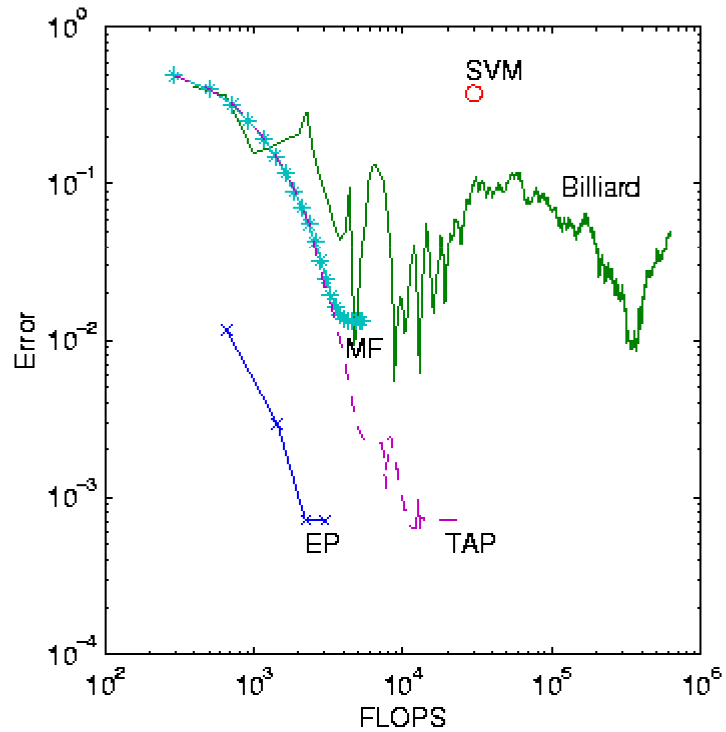
Error = distance to true mean of w

Billiard = Monte Carlo sampling
(Herbrich et al, 2001)

Opper&Winther's algorithms:

MF = mean-field theory

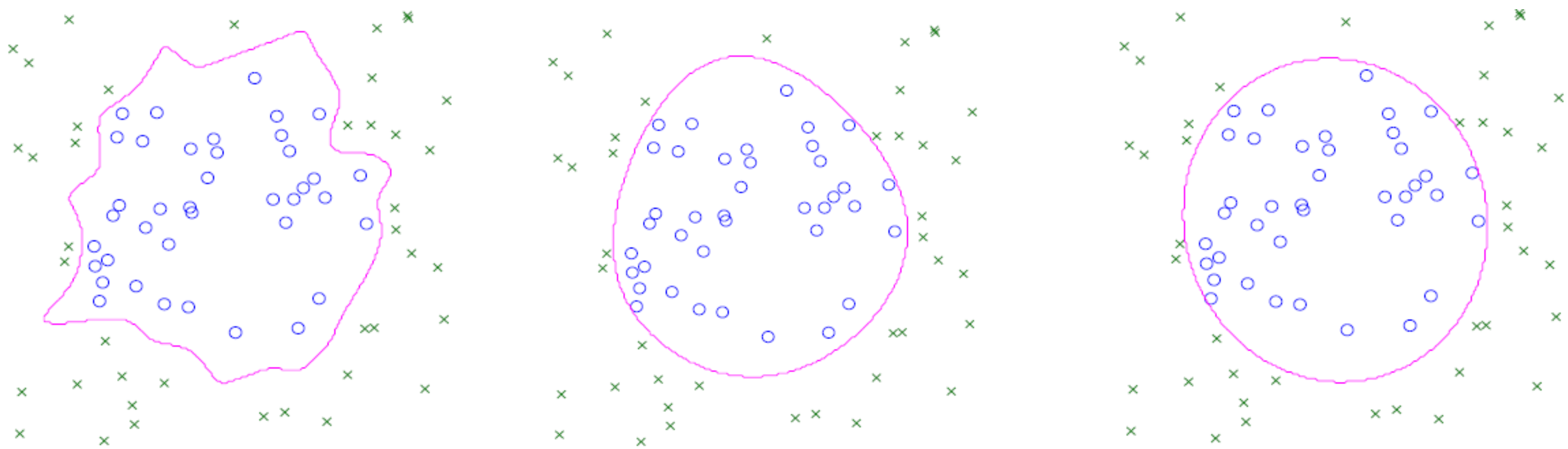
TAP = cavity method (equiv to Gaussian EP for this problem)



Gaussian kernels

- Map data into high-dimensional space so that

$$\phi(\mathbf{x}_i)^{\mathrm{T}}\phi(\mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$



Bayesian model comparison

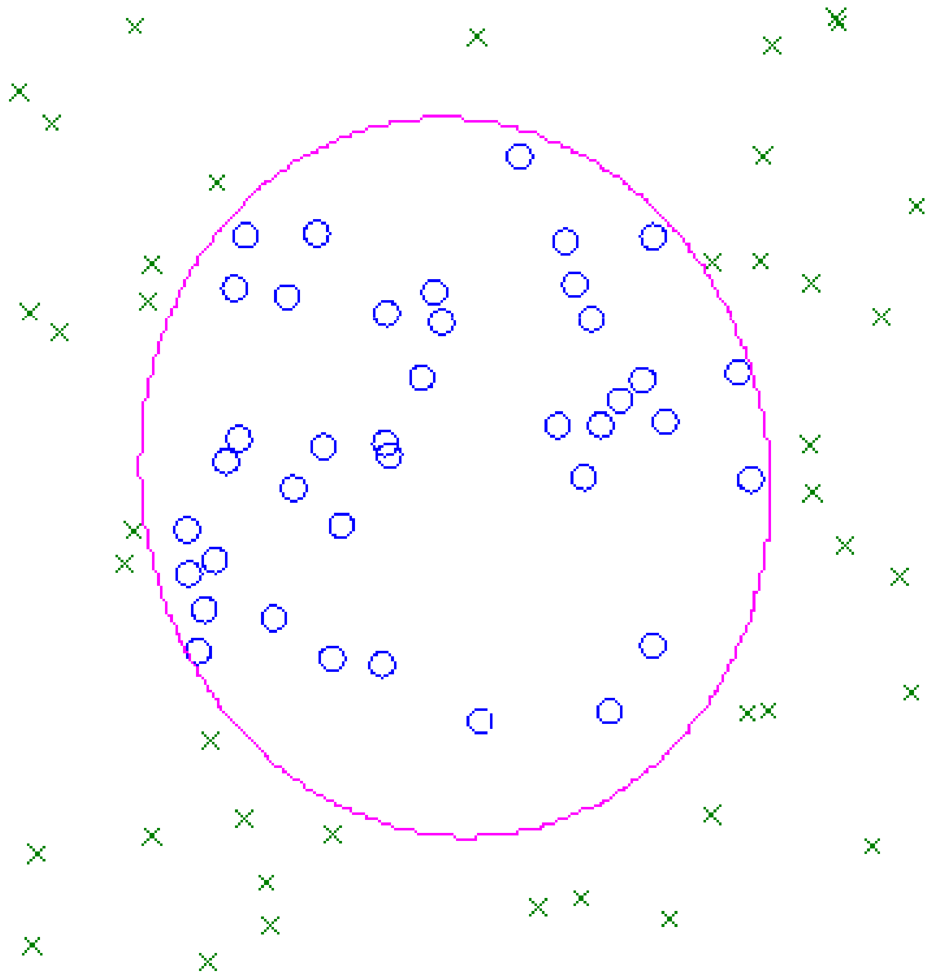
- Multiple models M_i with prior probabilities $p(M_i)$
- Posterior probabilities:

$$p(M_i|D) \propto p(D|M_i)p(M_i)$$

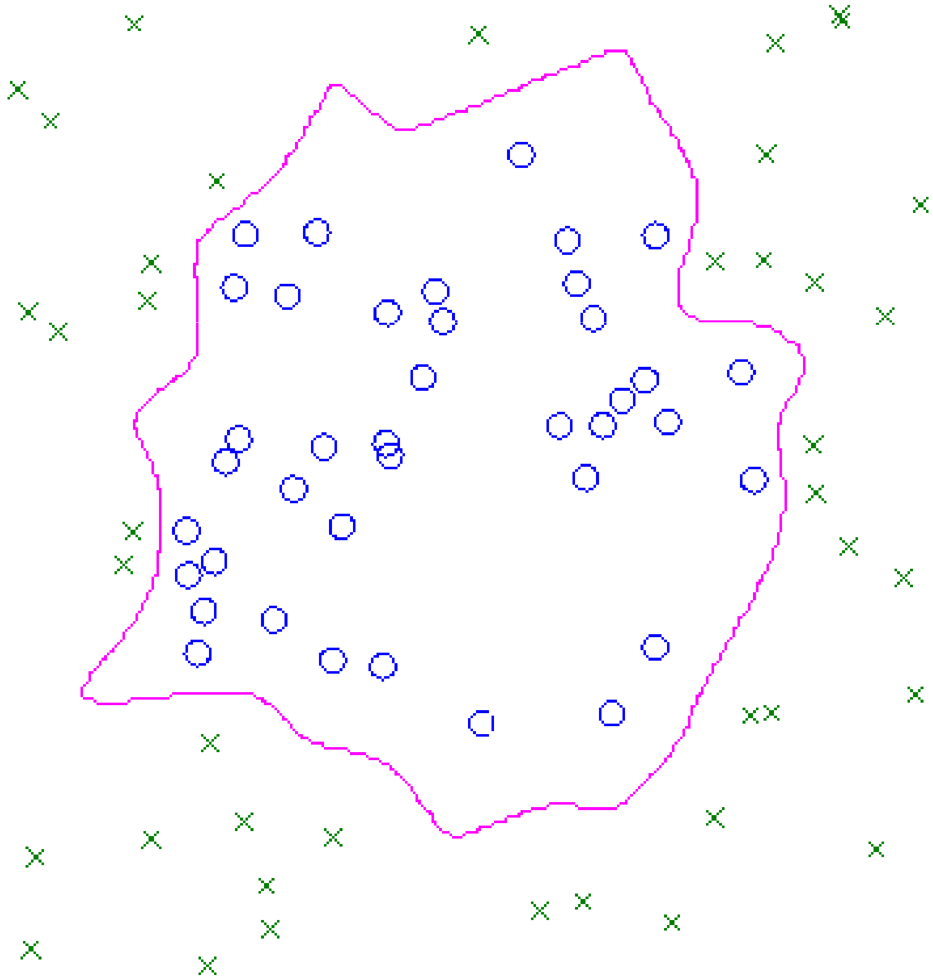
- For equal priors, models are compared using model evidence:

$$p(D|M_i) = \int_\theta p(D, \theta|M_i)d\theta$$

Highest-probability kernel



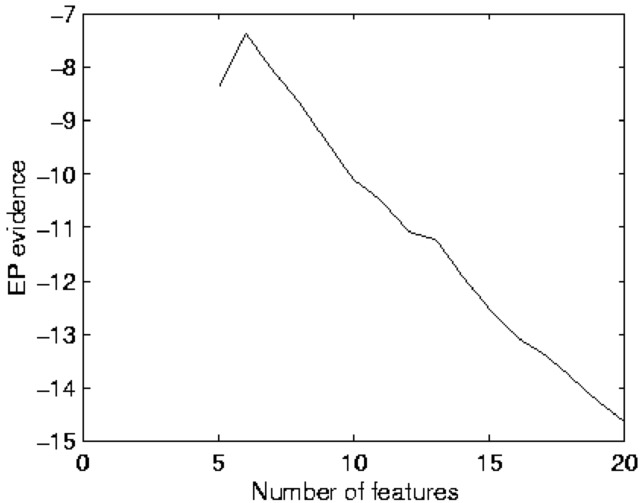
Margin-maximizing kernel



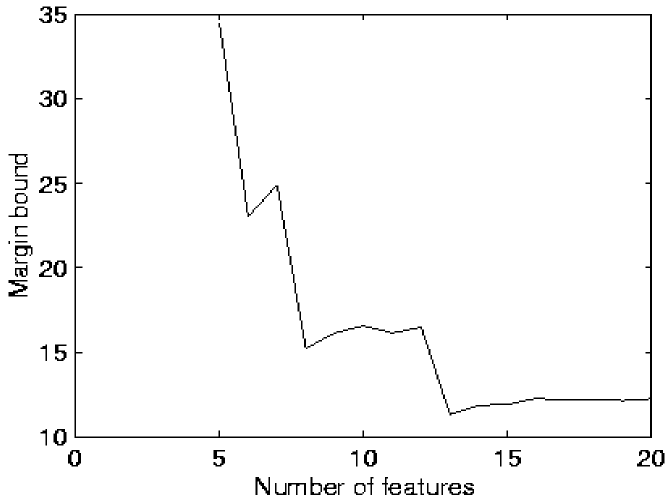
Bayesian feature selection

Synthetic data where 6 features are relevant (out of 20)

Bayes picks 6



Margin picks 13



EP versus Monte Carlo

- Monte Carlo is general but expensive
  - A sledgehammer
- EP exploits underlying simplicity of the problem (if it exists)
- Monte Carlo is still needed for complex problems (e.g. large isolated peaks)
- Trick is to know what problem you have

Further reading

- Bayes Point Machine toolbox
  http://research.microsoft.com/~minka/papers/ep/bpm/
- EP bibliography
  http://research.microsoft.com/~minka/papers/ep/roadmap.html
- EP quick reference
  http://research.microsoft.com/~minka/papers/ep/minka-ep-quickref.pdf