A family of algorithms
for
approximate Bayesian inference

Thomas P. Minka

MIT Media Lab

tpminka@media.mit.edu

www.media.mit.edu/~tpminka/

Supervised by Rosalind Picard, Affective Computing Group

Low−level language

Optimize free parameters

Minimize training set error

Regularize

Occam’s razor

Cross−validation

Maximize margins

Boosting

⋮

High−level language

Model the domain
with a stochastic process

Condition on data

Sum over possibilities

Automatic regularization,
margins, voting,
complexity control

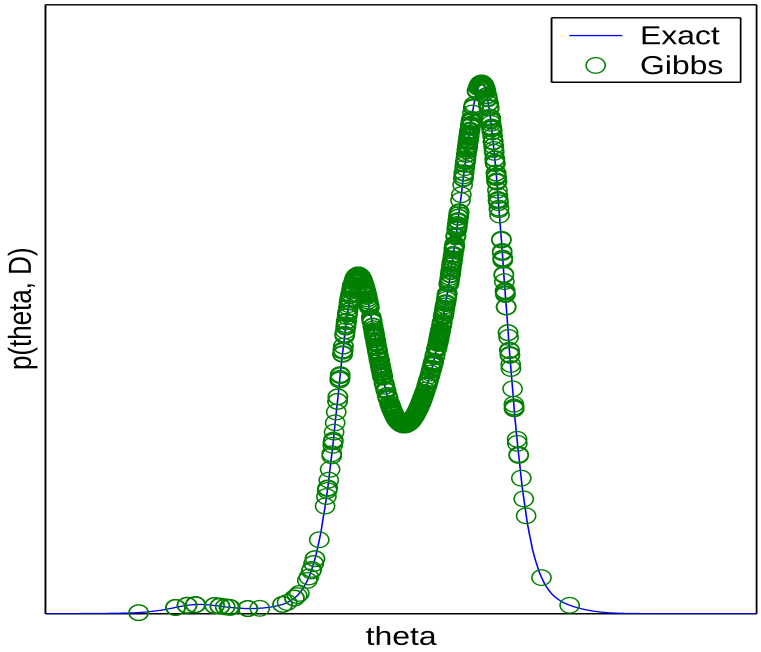
Exposes logic behind inferences

Bayesian quantities

Unnormalized posterior	$p(x, Data)$
Evidence (normalizing term)	$p(Data) = \int_x p(x, Data) dx$
Posterior	$E[x \vert Data] = \int_x x \; p(x \vert Data) \; dx$
Posterior mean	$E[x \vert Data] = \int_x x \; p(x \vert Data) \; dx$
Predictive density	$p(y \vert Data) = \int_x p(y \vert x) \; p(x \vert Data) \; dx$

Approximating posterior distributions

Sampling

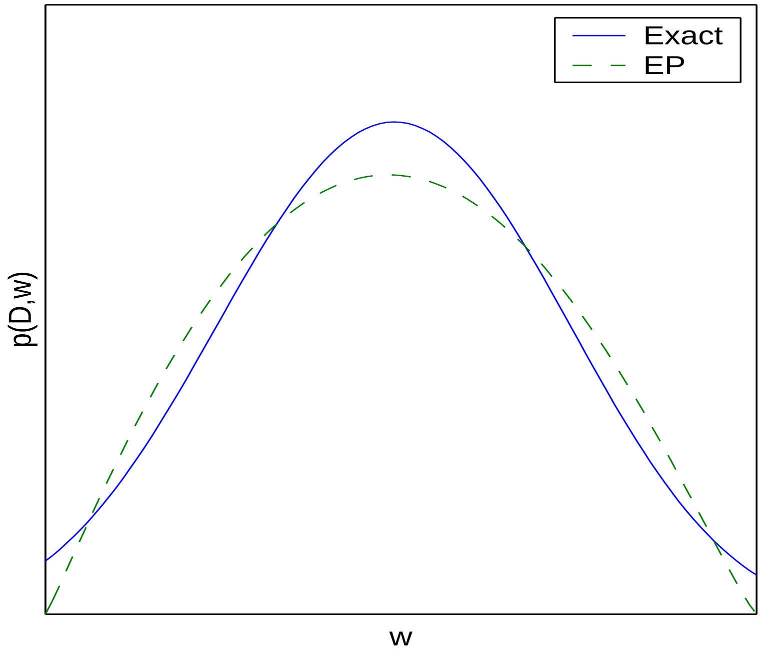


good for complex,

multimodal posteriors

slow, predictable

Deterministic approximation



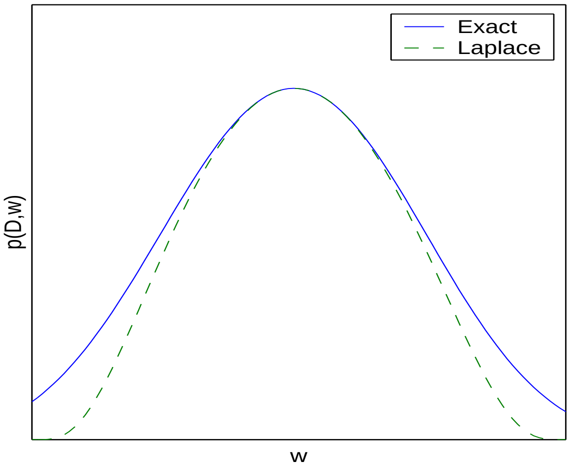
good for simple,

smooth posteriors

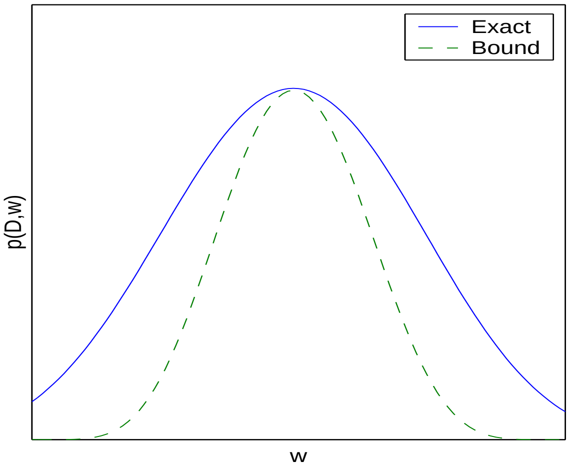
fast, unpredictable

Deterministic approximation

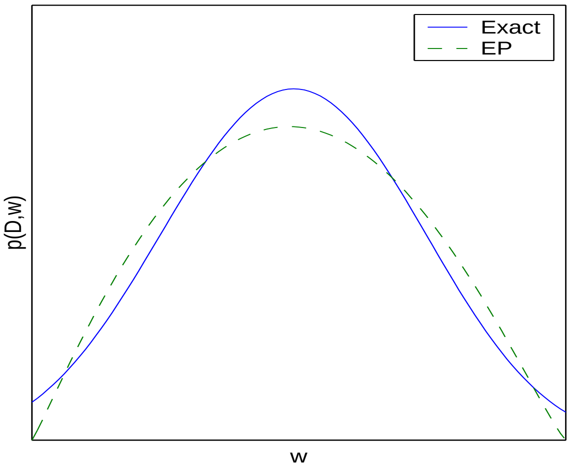
Laplace's method



Variational bound



Minimize
KL−divergence



- Assumed−density filtering
- Belief propagation
- Expectation propagation

Outline

- Assumed−density filtering (ADF)

  = Sequential KL−projection

  (Boyen&Koller, Opper&Winther, Barber&Sollich, Lauritzen, ...)

- Expectation propagation (EP)

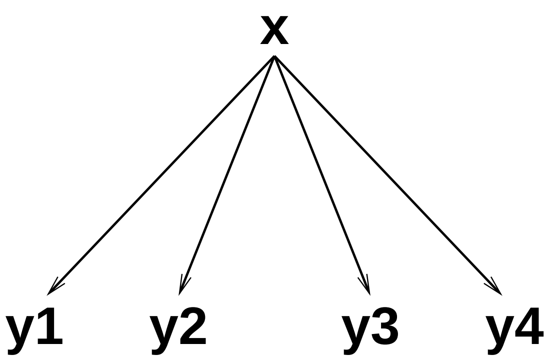
  = ADF + iterative refinement

- Belief propagation

  = EP for factorized posterior

- Application of EP: Bayes point machine
  - Voting all linear classifiers
  - Choosing feature space (kernel)

Recursive estimation



$$p(x|y_1, ..., y_4) = p(x)p(y_1|x)p(y_2|x)p(y_3|x)p(y_4|x)$$

If x,y are jointly Gaussian (or in exp family),
use recursive estimation (Kalman filter):

$$\begin{aligned} p(x|y_1, ..., y_t) &\propto p(y_t|x)p(x|y_1, ..., y_{t-1}) \\ q^{new}(x) &\propto p(y_t|x)q^{old}(x) \end{aligned}$$

q(x) is Gaussian each time −− only propagate mean, var of x

(Kalman updates)

Sequential, but independent of ordering

What if p(y|x) is not linear or not Gaussian?

Extended Kalman filtering

p(y|x) is Gaussian in y but not linear in x, e.g.

$$p(y|x) = \frac{1}{\sqrt{2\pi v}} \exp(-\frac{(y - f(x))^2}{2v})$$

Approximate p(y|x) with a linearization:

$$\begin{aligned} \tilde{p}(y|x) &= \frac{1}{\sqrt{2\pi v}} \exp(-\frac{(y - \tilde{f}(x))^2}{2v}) \\ \tilde{f}(x) &= f(x_0) + f'(x_0)(x - x_0) \\ x_0 &= E_q[x] \text{ (based on current } q(x)) \end{aligned}$$

This makes the posterior Gaussian:

$$q^{new}(x) \propto \tilde{p}(y_t|x) q^{old}(x)$$

yields the EKF updates

Sequential, but sensitive to ordering

Batch relinearization: Change all approximations to new x0

Assumed−density filtering

Works for any p(y|x):

$$\begin{aligned} p(x, D) &= p(x)p(y_1|x)p(y_2|x)p(y_3|x)p(y_4|x) \\ &= p(x)t_1(x)t_2(x)t_3(x)t_4(x) \end{aligned}$$

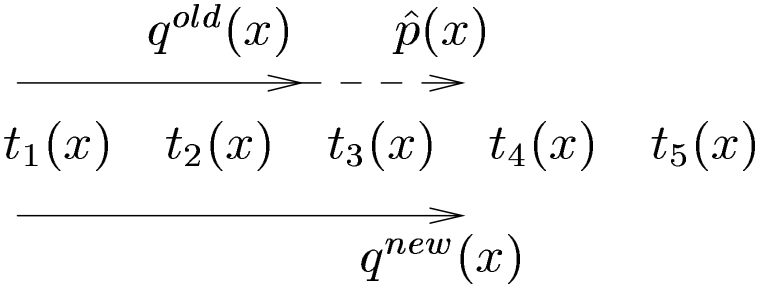
1. Initialize $q(x) = p(x)$

2. Loop i:

$$\begin{aligned} \hat{p}(x) &= \frac{t_i(x)q^{old}(x)}{\int t_i(x)q^{old}(x)dx} \\ q^{new}(x) &= \operatorname{argmin}_q \, D(\hat{p} \,||\, q) \end{aligned}$$

where q(x) has constrained parametric form

Graphically:



Gaussian filtering

At each step, assume posterior is Gaussian:

$$\mathrm{argmin}_q D(\hat{p} \mid\mid q) \qquad\qquad q(x) \sim \mathcal{N}(x; m, v)$$

$$\Downarrow$$

$$\begin{aligned} E_q[x] &= E_{\hat{p}}[x] \\ m &= \int x\hat{p}(x)dx \\ E_q[x^2] &= E_{\hat{p}}[x^2] \\ v + m^2 &= \int x^2\hat{p}(x)dx \end{aligned}$$

$$\mathcal{N}(x; m^{old}, v^{old})p(y_i|x) \Rightarrow \mathcal{N}(x; m^{new}, v^{new})$$

q preserves certain expectations of p

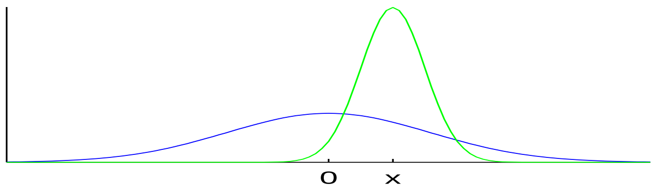
→ true for any assumed density in exponential family

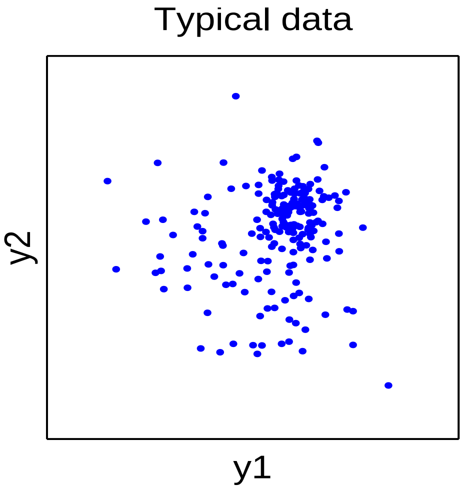
(Gamma, Dirichlet, multinomial, ...)

Example

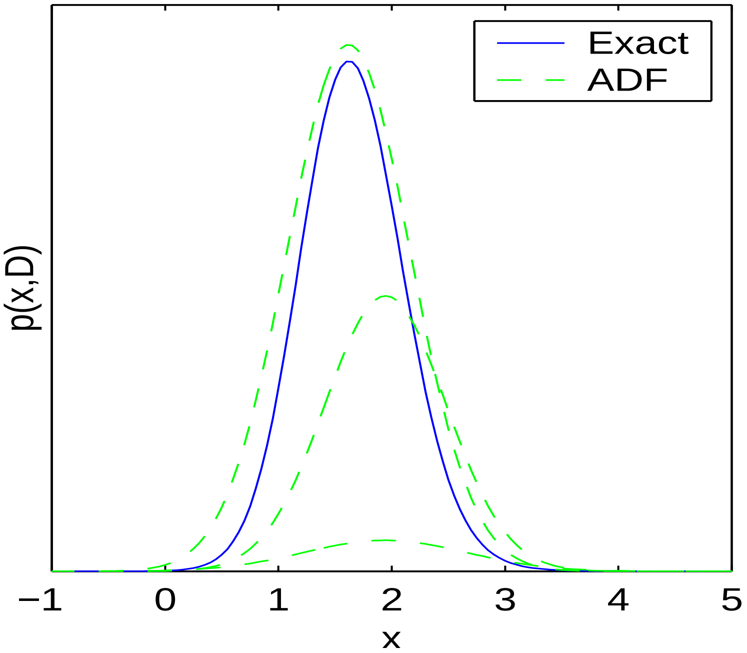
Data model

$$p(y|x) = \frac{1}{2}\mathcal{N}(y; x, 1) + \frac{1}{2}\mathcal{N}(y; 0, 10)$$





ADF posterior for three orderings of same data:



True x = 2

20 data points

ADF is sensitive to ordering

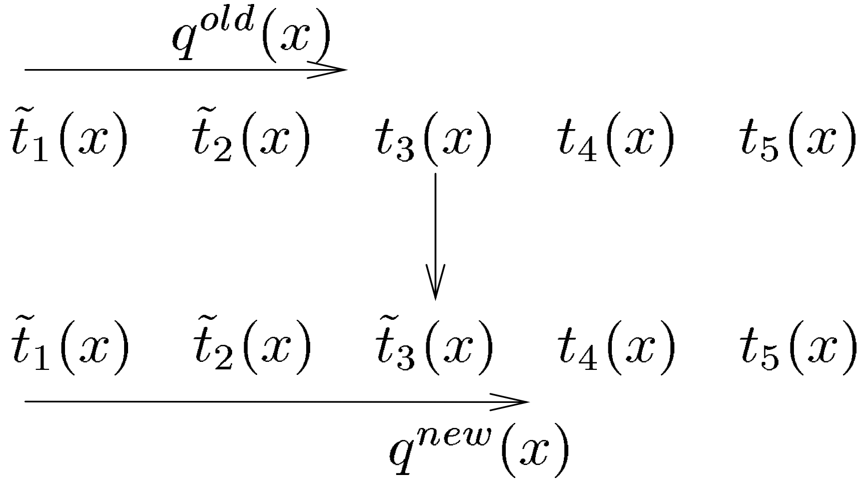
Can we make ADF independent of ordering?

Another view of ADF

ADF can interpreted as making an EKF−type update:

$$q^{new}(x) \propto \tilde{t}_i(x) q^{old}(x)$$

Graphically:



$$q^{old}(x)$$

$$\tilde{t}_1(x) \quad \tilde{t}_2(x) \quad t_3(x) \quad t_4(x) \quad t_5(x)$$

$$\tilde{t}_1(x) \quad \tilde{t}_2(x) \quad \tilde{t}_3(x) \quad t_4(x) \quad t_5(x)$$

$$q^{new}(x)$$

as long as we define

$$\tilde{t}_3(x) = \frac{q^{new}(x)}{q^{old}(x)}$$

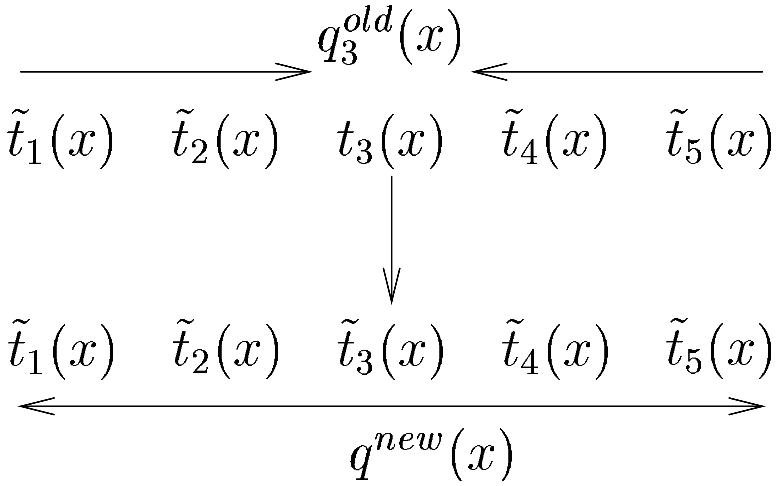
$$q \text{ in exp family} \Rightarrow \tilde{t} \text{ has same form}$$

Approximate each term as Gaussian, then multiply

Now we can repeat and refine the Gaussians

Expectation Propagation

Use the approximations to refine each term:



$q_3^{old}(x)$

$\tilde{t}_1(x)$ $\tilde{t}_2(x)$ $t_3(x)$ $\tilde{t}_4(x)$ $\tilde{t}_5(x)$

$\tilde{t}_1(x)$ $\tilde{t}_2(x)$ $\tilde{t}_3(x)$ $\tilde{t}_4(x)$ $\tilde{t}_5(x)$

$q^{new}(x)$

To refine a term:

1. Remove $\tilde{t}_i$:

$$q_i^{old}(x) \propto \frac{q^{new}(x)}{\tilde{t}_i(x)}$$

2. Recompute $q^{new}(x)$ as in ADF:

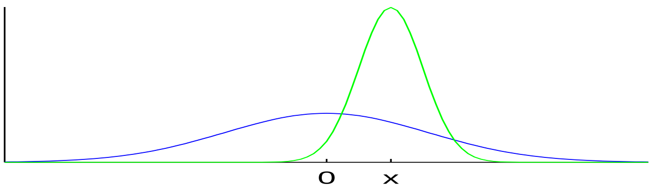
$$\begin{aligned} \hat{p}(x) &\propto t_i(x) q_i^{old}(x) \\ q^{new}(x) &= \operatorname{argmin}_q D(\hat{p} \,||\, q) \end{aligned}$$

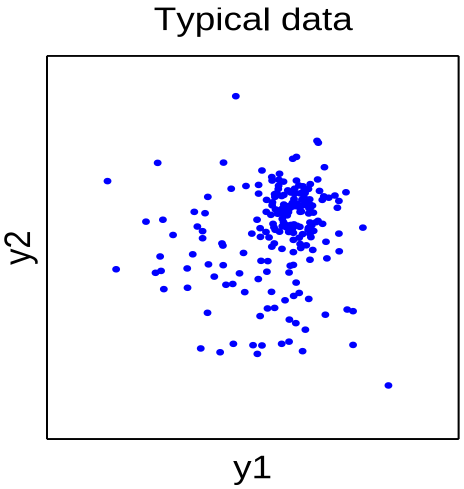
3. $\tilde{t}_i(x) = \frac{q^{new}(x)}{q_i^{old}(x)}$

Example continued

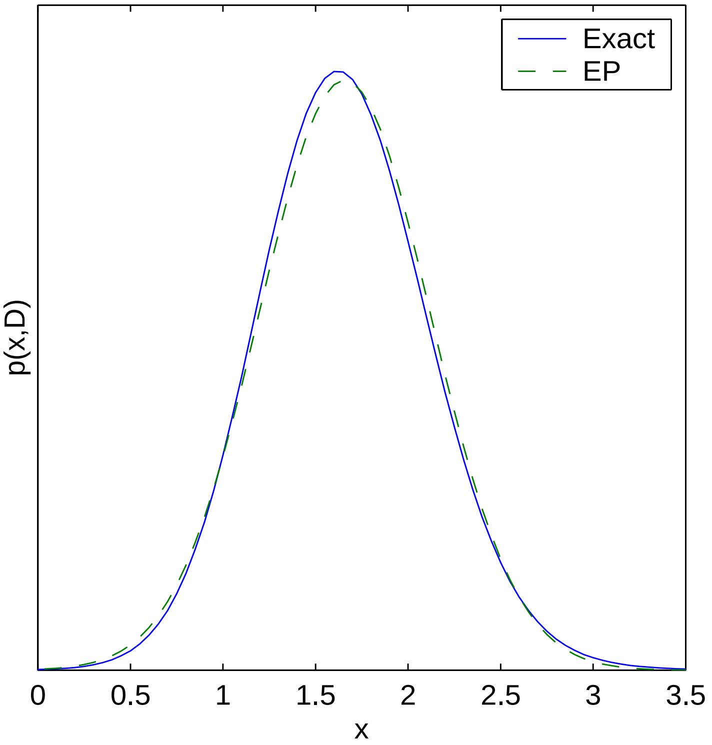
Data model

$$p(y|x) = \frac{1}{2}\mathcal{N}(y;x,1) + \frac{1}{2}\mathcal{N}(y;0,10)$$

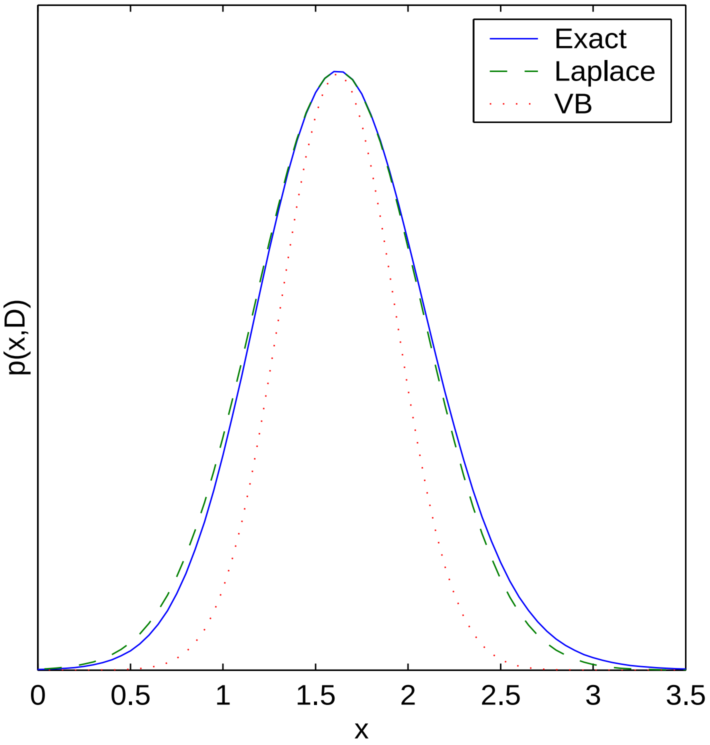




EP posterior at convergence

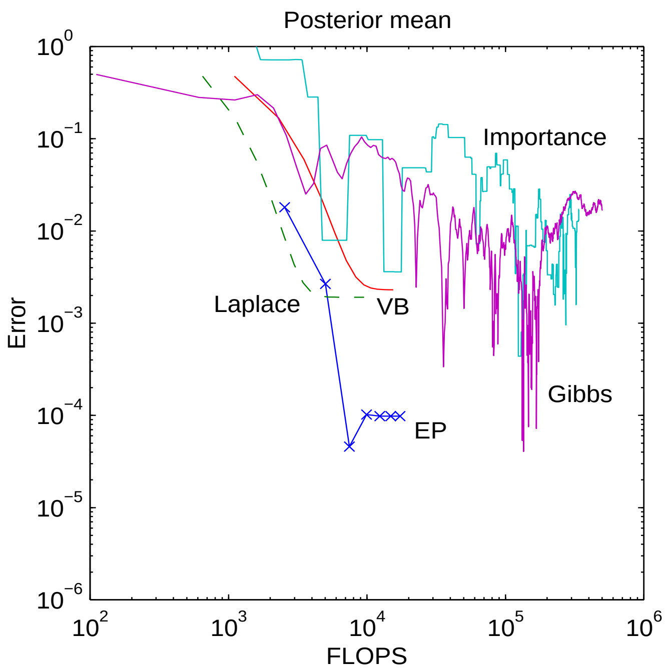


Other methods

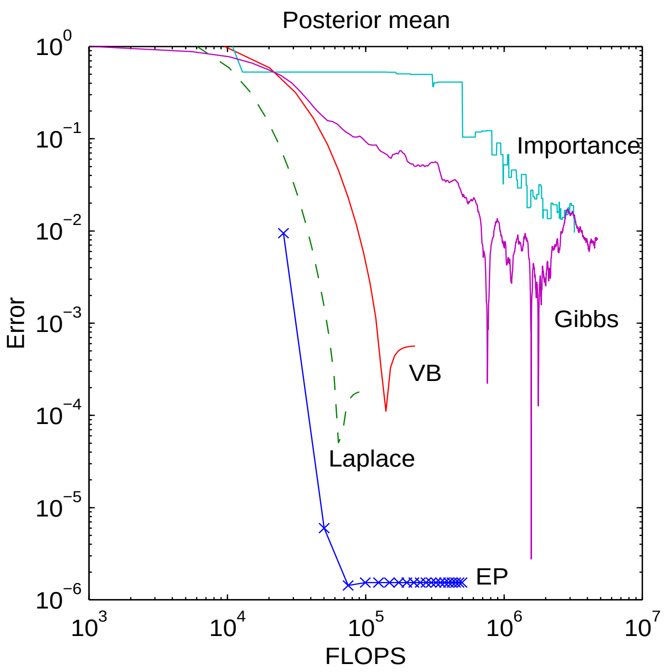


All independent of data ordering

Performance



Data size n=20



n=200

ADF = first ‘x’ of EP

VB = variational bound

Deterministic methods improve with more data

(posterior is more Gaussian)

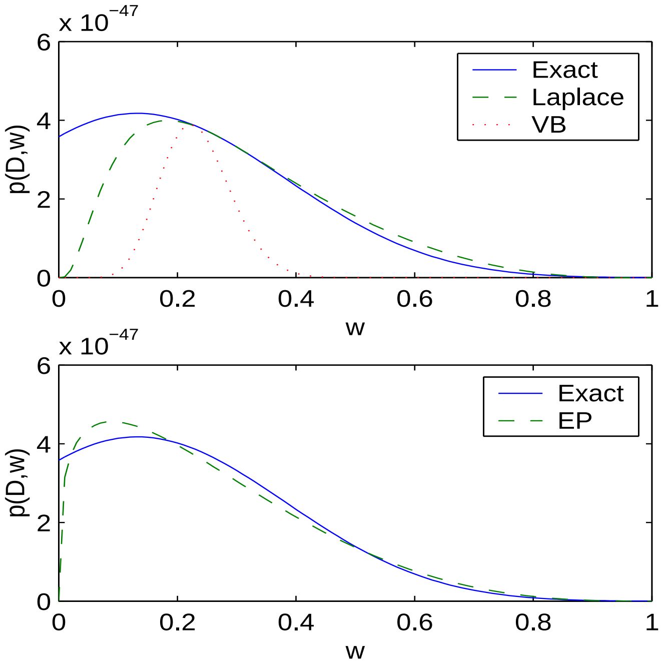
Sampling methods do not care

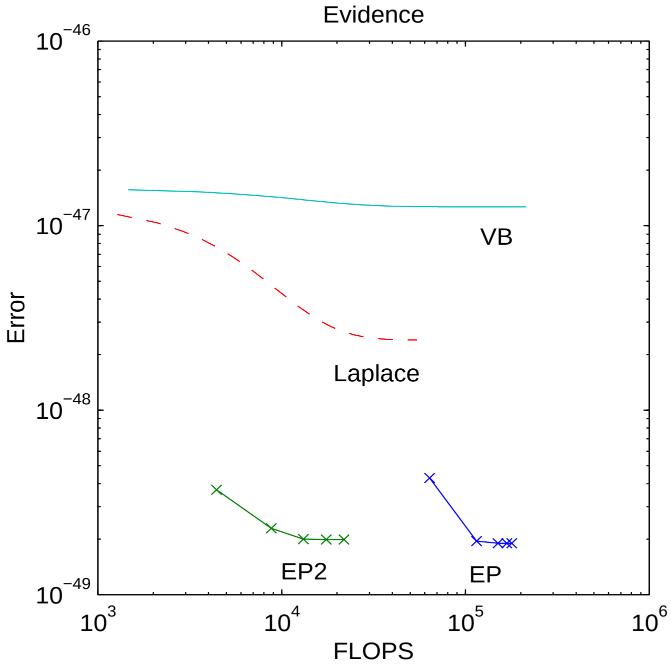
Mixture weights example

$$
\begin{aligned}
p(w, D) &= p(w) \prod_i p(y_i|w) \\
p(y|w) &= w\mathcal{N}(y; 0, 3) + (1-w)\mathcal{N}(y; 1, 3) \\
p(w) &= 1 \\
q(w) &\sim \text{Dirichlet}(a_1, a_2)
\end{aligned}
$$

KL-minimization preserves the expectations $E[\log(w)]$, $E[\log(1-w)]$

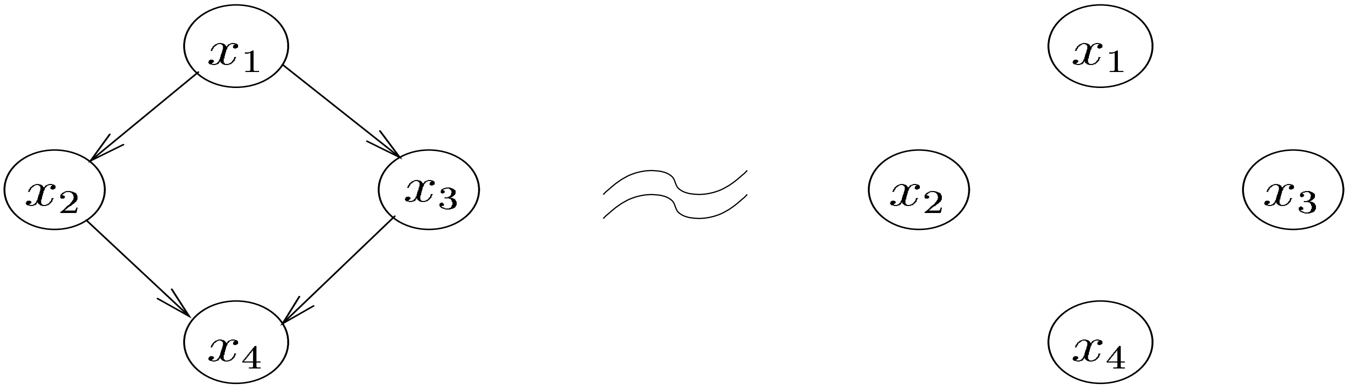
Typical result





EP2 preserves the expectations $E[w]$, $E[w^2]$ instead

Factorized approximation of belief networks



$p(x_1)p(x_2|x_1)p(x_3|x_1)p(x_4|x_2,x_3)$ $\qquad$ $q_1(x_1)q_2(x_2)q_3(x_3)q_4(x_4)$

$$\begin{aligned} \mathrm{argmin}_q D(\hat{p} \,||\, q) \quad &\Rightarrow \quad q_k(x_k) = \hat{p}(x_k) \\ \text{where } q(x) = \prod_k q_k(x_k) \quad &\qquad \text{(marginals)} \end{aligned}$$

Factorized ADF (Boyen&Koller):

$$p(\mathbf{x}) = \prod_i p(x_i|\mathrm{pa}(x_i))$$

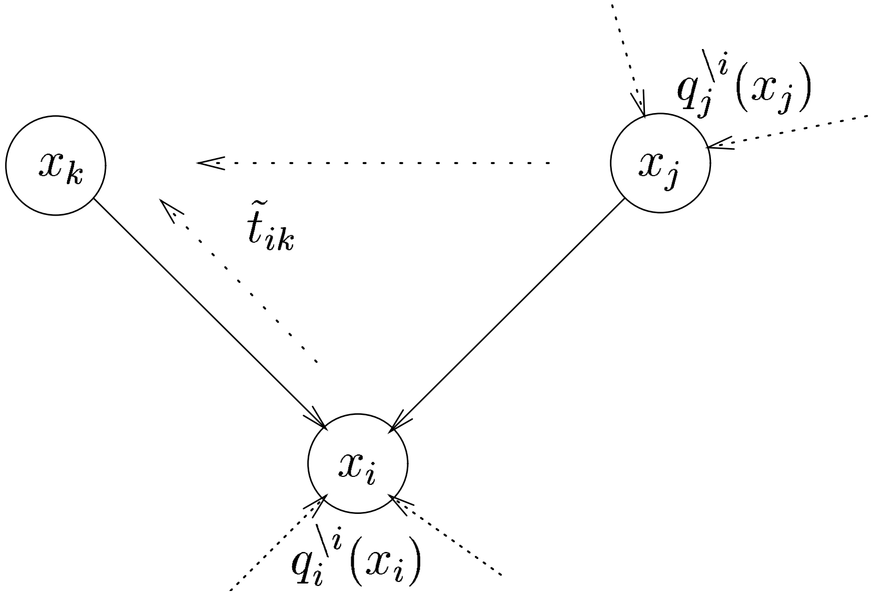
1. Init $q(\mathbf{x}) = 1$
2. Loop i:

$$\begin{aligned} \hat{p}(\mathbf{x}) &\propto p(x_i|\mathrm{pa}(x_i))q^{old}(\mathbf{x}) \\ q_k^{new}(x_k) &= \hat{p}(x_k) \end{aligned}$$

Factorized EP

$$
\begin{aligned}
\tilde{t}_i(\mathbf{x}) &= \prod_k \tilde{t}_{ik}(x_k) && \text{(messages)} \\
q^{new}(\mathbf{x}) &\propto \prod_j \tilde{t}_j(\mathbf{x}) && \text{(belief state)} \\
q_i^{old}(\mathbf{x}) &\propto \prod_{j \neq i} \tilde{t}_j(\mathbf{x}) && \text{(partial belief state)} \\
\tilde{t}_{ik}(x_k) &= \frac{q^{new}(x_k)}{q_i^{old}(x_k)} \\
&= \sum_{\mathbf{x} \backslash x_k} p(x_i | \mathrm{pa}(x_i)) \prod_{j \neq k} q_i^{old}(x_j) d\mathbf{x} \\
&\quad \text{(message } i \to k\text{)}
\end{aligned}
$$

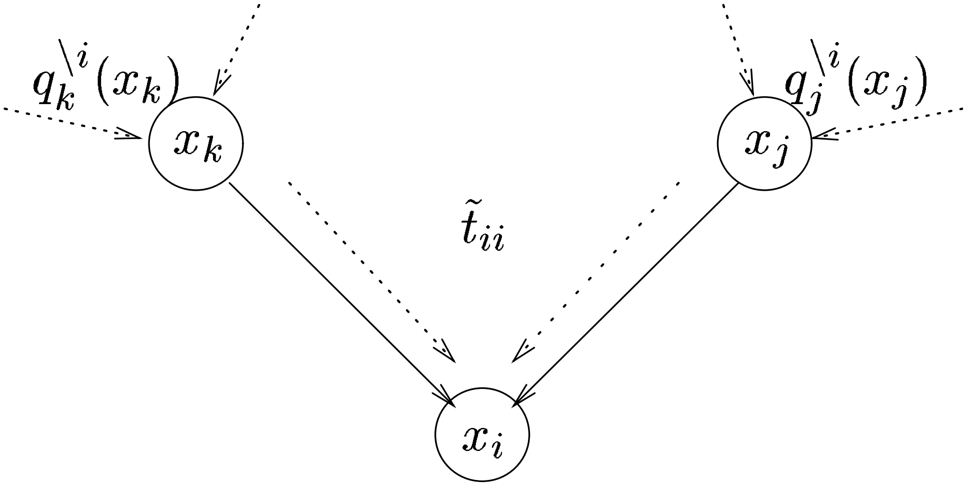
Child to parent:



$$\tilde{t}_{ik}(x_k) = \int_{x_i, x_j} p(x_i | x_k, x_j) q_i^{old}(x_i) q_i^{old}(x_j)$$

Factorized EP cont'd

Parents to child:



$$\tilde{t}_{ii}(x_i) = \int_{x_k,x_j} p(x_i|x_k,x_j)q_i^{old}(x_k)q_i^{old}(x_j)$$

Loopy belief propagation is factorized EP

EP can use structured approximations too, e.g. Markov chain

−> more accurate posteriors

EP can restrict parametric form, e.g. Gaussian

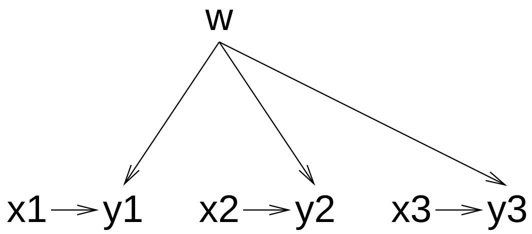
−> simpler messages

Bayes point machine

Bayesian approach to linear classification

Use w to classify x:

$$\begin{array}{rcll} \mathbf{w}^{\mathrm{T}}\mathbf{x}_i & > & 0 & \text{(class 1)} \\ \mathbf{w}^{\mathrm{T}}\mathbf{x}_i & < & 0 & \text{(class 2)} \end{array}$$



$$p(\mathbf{w}, D) = p(\mathbf{w}) \prod_i p(y_i|\mathbf{x}_i, \mathbf{w})$$

p(w) is uniform

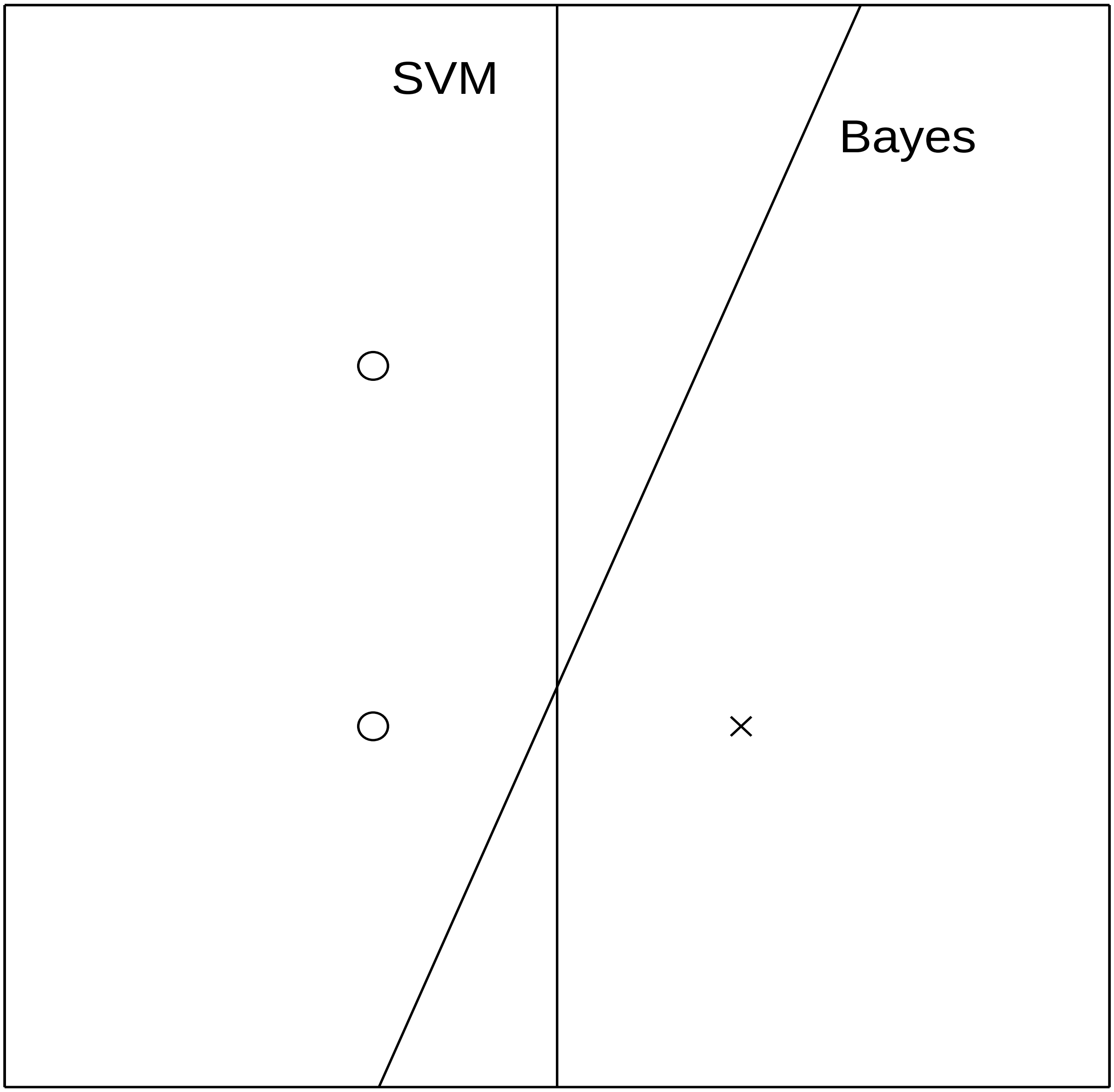
$$\begin{array}{rcl} p(y|\mathbf{x},\mathbf{w}) & = & \Theta(y\mathbf{w}^{\mathrm{T}}\mathbf{x}) \\ & = & \begin{cases} 1 & \text{if } \mathbf{w} \text{ is a perfect separator} \\ 0 & \text{otherwise} \end{cases} \end{array}$$

Classify a new data point by voting:

$$\begin{array}{rcl} p(y|\mathbf{x}, D) & = & \int_{\mathbf{w}} p(y|\mathbf{x},\mathbf{w}) p(\mathbf{w}|D) d\mathbf{w} \\ y & = & E[\mathrm{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})|D] \\ & \approx & \mathrm{sign}(E[\mathbf{w}|D]^{\mathrm{T}}\mathbf{x}) \end{array}$$

E[w|D] is the Bayes Point

Bayes point machine example



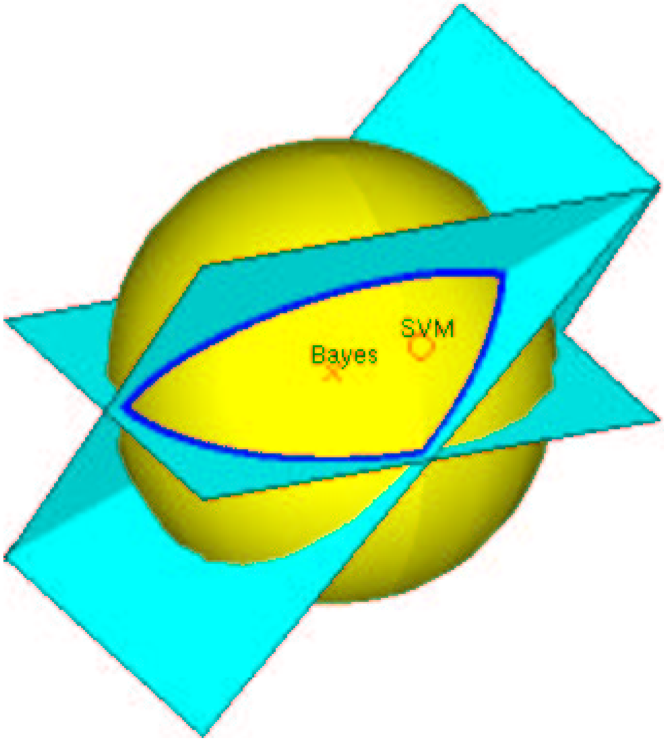
SVM → Maximize margin

(distance to closest data point)

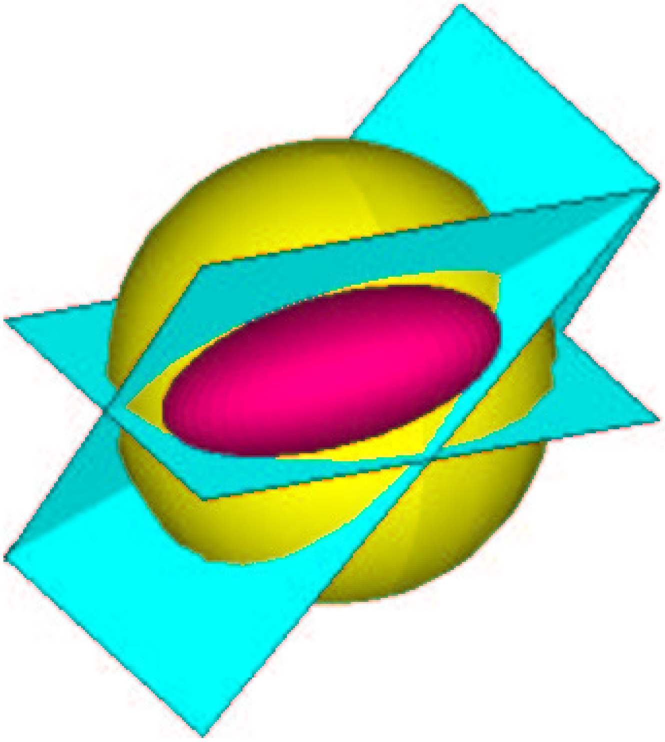
Bayes → Vote all perfect separators

Performance of EP

Version space



EP Gaussian posterior

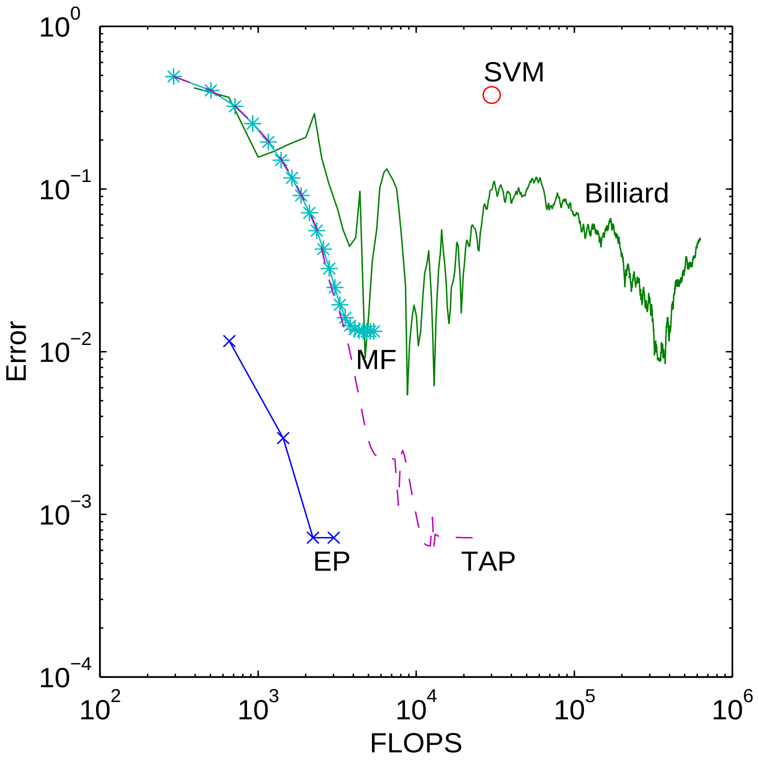


Billiard = Monte Carlo

Opper&Winther’s algs:

MF = mean−field theory

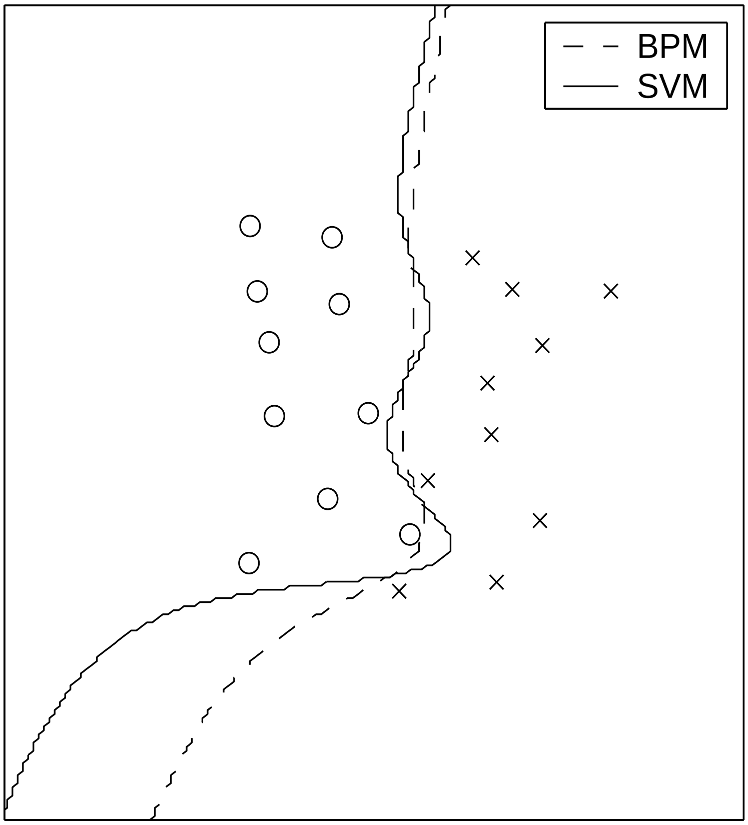
TAP = cavity method
   (equiv to Gaussian EP)



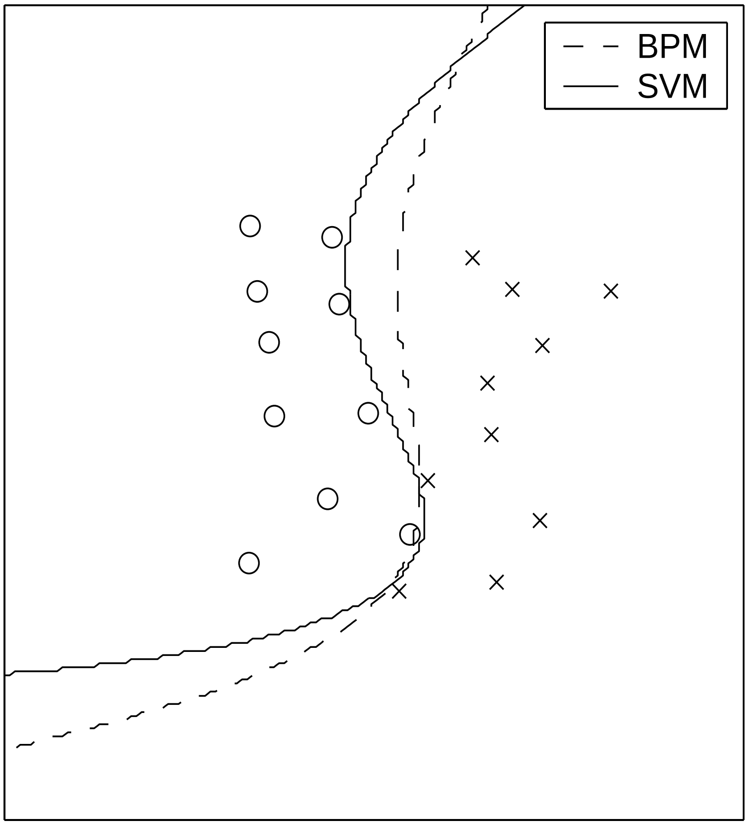
Gaussian kernels

Map data into high dimensional space so that

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



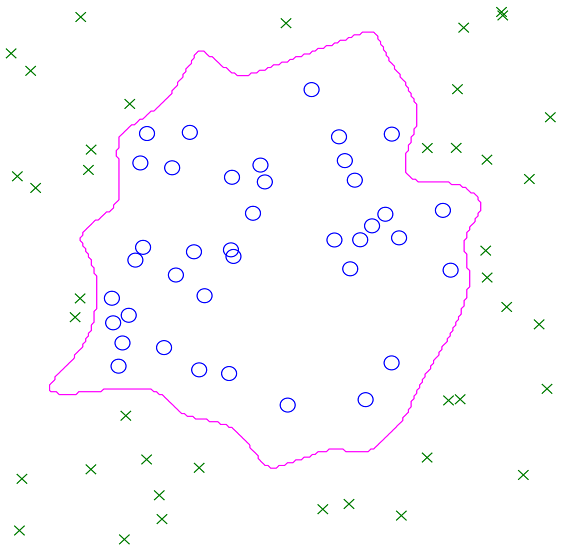
narrow width 0.2



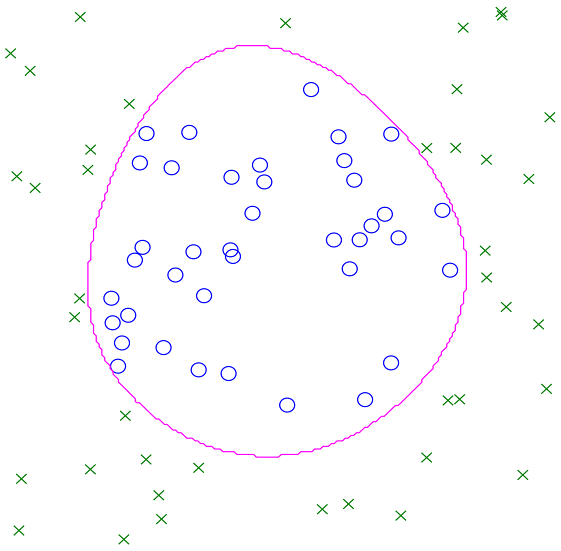
wide width 0.5

SVM boundaries are more contrived, sensitive to kernel

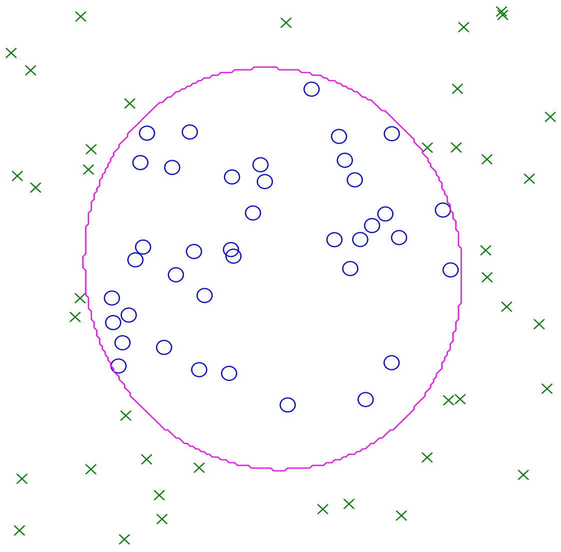
Kernel selection



Gaussian kernel, width 0.08
(SVM choice)



Gaussian kernel, width 0.6
(Bayes choice among Gaussians)



Quadratic kernel
(Bayes choice)

Kernel	R^2/ρ^2	$\log(p(D))$
$\sigma = 0.08$	18	-39
$\sigma = 0.6$	108	-19
quadratic	656	-16

SVM and EP have similar boundaries, but prefer different kernels

Summary

- Expectation propagation = assumed−density filtering plus iterative refinement
- Batch operation, more accurate
- Generalizes belief propagation to hybrid nets and non−factorized approximations
- Generalizes TAP method for Bayes point machine
- Like belief propagation, may not converge, local minima
- No error estimate available