Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

An Introduction to Factor Graphs

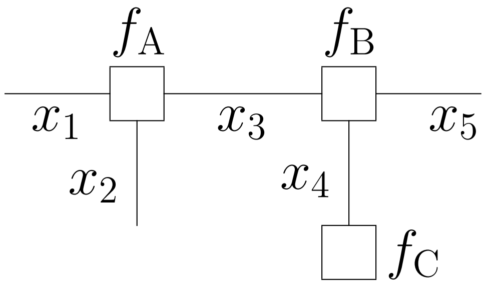
Hans-Andrea Loeliger

Definition

A factor graph represents the factorization of a function of several variables. We use Forney-style factor graphs (Forney, 2001).

Example:

$$f(x_1, x_2, x_3, x_4, x_5) = f_\mathrm{A}(x_1, x_2, x_3) \cdot f_\mathrm{B}(x_3, x_4, x_5) \cdot f_\mathrm{C}(x_4).$$



Rules:

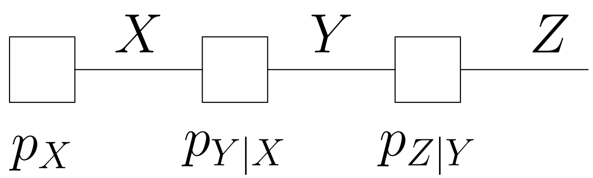
- A node for every factor.
- An edge or half-edge for every variable.
- Node g is connected to edge x iff variable x appears in factor g.

(What if some variable appears in more than 2 factors?)

Example:

Markov Chain

$$p_{XYZ}(x, y, z) = p_X(x)\, p_{Y|X}(y|x)\, p_{Z|Y}(z|y).$$

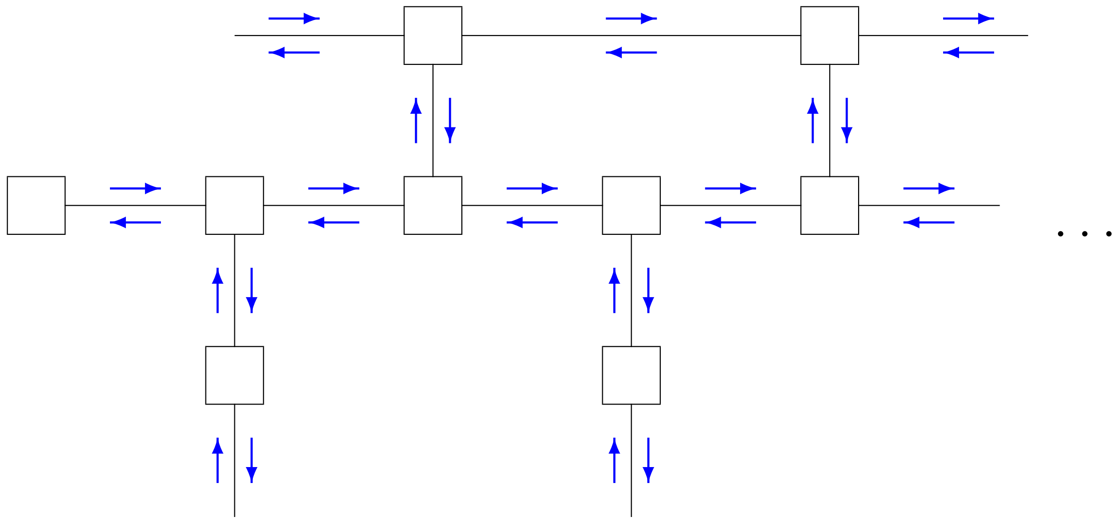


We will often use capital letters for the variables. (Why?)

Further examples will come later.

Message Passing Algorithms

operate by passing messages along the edges of a factor graph:



A main point of factor graphs (and similar graphical notations):

A Unified View of Historically Different Things

Statistical physics:

- Markov random fields (Ising 1925)

Signal processing:

- linear state-space models and Kalman filtering: Kalman 1960...
- recursive least-squares adaptive filters
- Hidden Markov models: Baum et al. 1966...
- unification: Levy et al. 1996...

Error correcting codes:

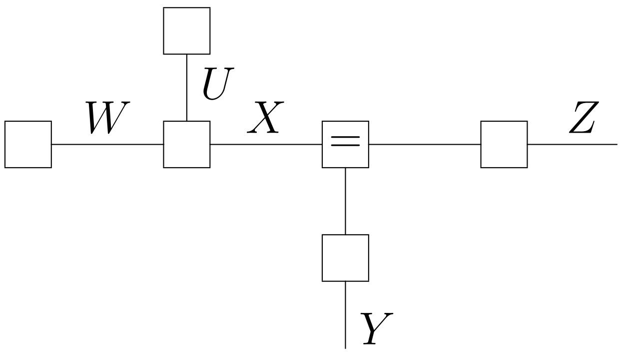
- Low-density parity check codes: Gallager 1962; Tanner 1981; MacKay 1996; Luby et al. 1998...
- Convolutional codes and Viterbi decoding: Forney 1973...
- Turbo codes: Berrou et al. 1993...

Machine learning, statistics:

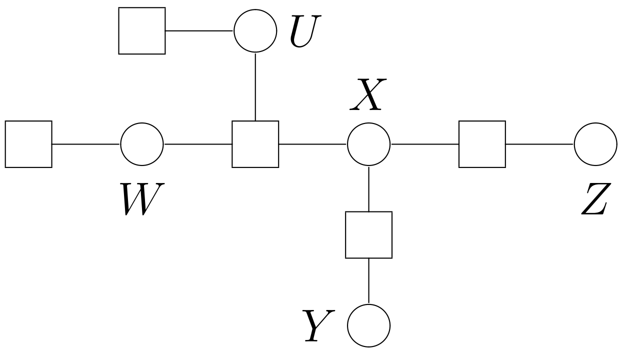
- Bayesian networks: Pearl 1988; Shachter 1988; Lauritzen and Spiegelhalter 1988; Shafer and Shenoy 1990...

Other Notation Systems for Graphical Models

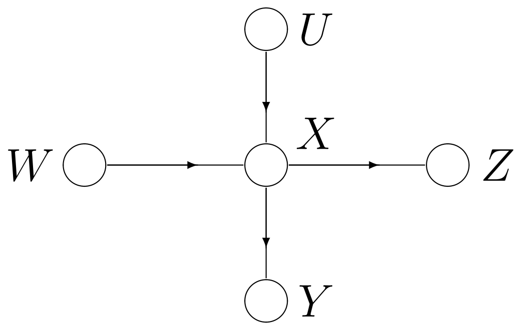
Example: $p(u, w, x, y, z) = p(u)p(w)p(x|u, w)p(y|x)p(z|x)$.



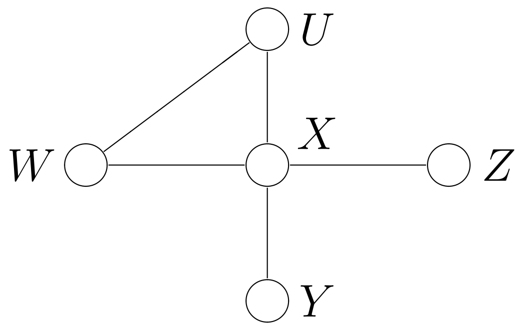
Forney-style factor graph.



Original factor graph [FKLW 1997].



Bayesian network.



Markov random field.

Outline

1. Introduction
2. The sum-product and max-product algorithms
3. More about factor graphs
4. Applications of sum-product & max-product to hidden Markov models
5. Graphs with cycles, continuous variables, . . .

The Two Basic Problems

1. Marginalization: Compute

$$\bar{f}_k(x_k) \stackrel{\triangle}{=} \sum_{\substack{x_1, \ldots, x_n \\ \text{except } x_k}} f(x_1, \ldots, x_n)$$

2. Maximization: Compute the "max-marginal"

$$\hat{f}_k(x_k) \stackrel{\triangle}{=} \max_{\substack{x_1, \ldots, x_n \\ \text{except } x_k}} f(x_1, \ldots, x_n)$$

assuming that f is real-valued and nonnegative and has a maximum. Note that

$$\operatorname{argmax} f(x_1, \ldots, x_n) = \left(\operatorname{argmax} \hat{f}_1(x_1), \ldots, \operatorname{argmax} \hat{f}_n(x_n) \right).$$

For large n, both problems are in general intractable even for $x_1, \ldots, x_n \in \{0, 1\}$.

Factorization Helps

For example, if $f(x_1, \ldots, f_n) = f_1(x_1) f_2(x_2) \cdots f_n(x_n)$ then

$$\bar{f}_k(x_k) = \left(\sum_{x_1} f_1(x_1) \right) \cdots \left(\sum_{x_{k-1}} f_{k-1}(x_{k-1}) \right) f_k(x_k) \cdots \left(\sum_{x_n} f_n(x_n) \right)$$

and

$$\hat{f}_k(x_k) = \left(\max_{x_1} f_1(x_1) \right) \cdots \left(\max_{x_{k-1}} f_{k-1}(x_{k-1}) \right) f_k(x_k) \cdots \left(\max_{x_n} f_n(x_n) \right).$$

Factorization helps also beyond this trivial example.

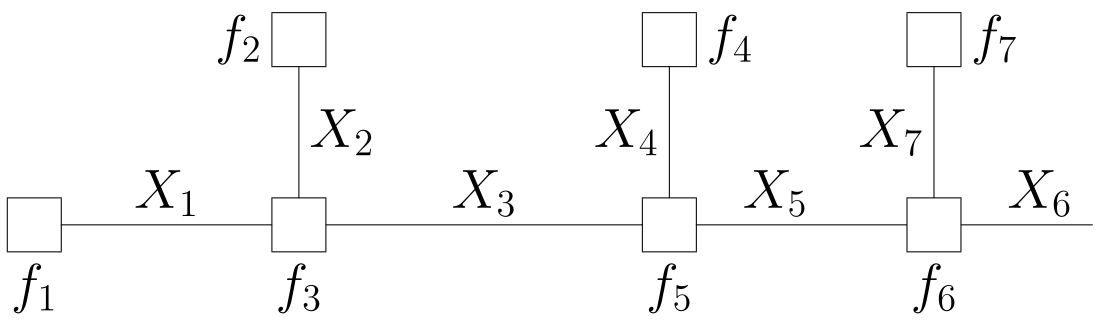
Towards the sum-product algorithm:

Computing Marginals—A Generic Example

Assume we wish to compute

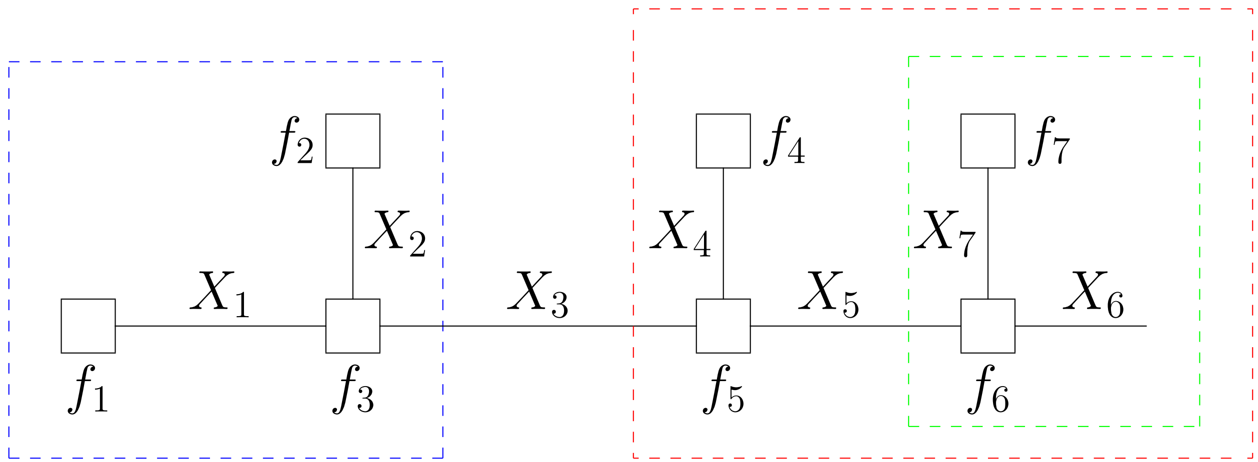
$$\bar{f}_3(x_3) = \sum_{\substack{x_1,\ldots,x_7 \\ \text{except } x_3}} f(x_1,\ldots,x_7)$$

and assume that f can be factored as follows:



Example cont'd:

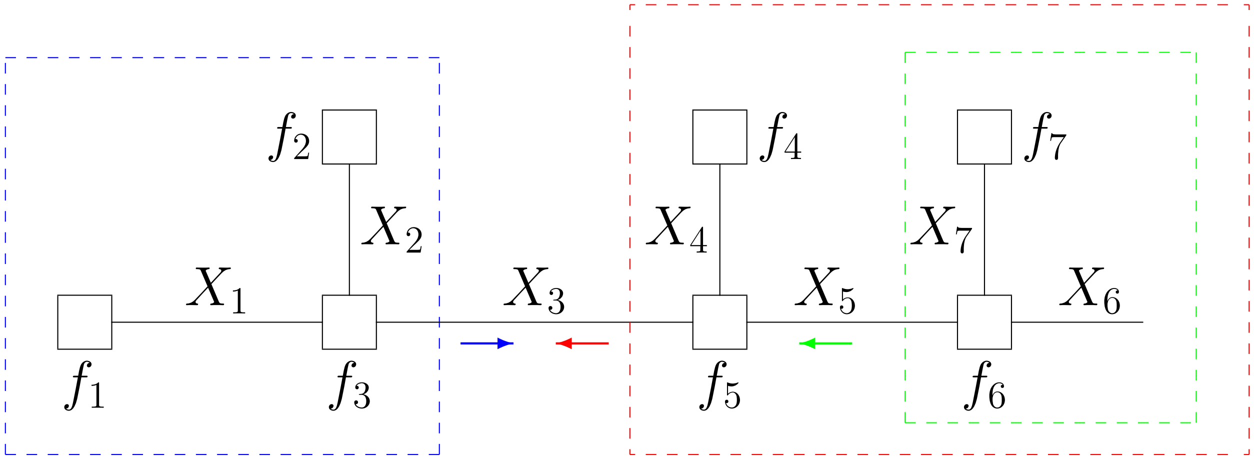
Closing Boxes by the Distributive Law



$$\bar{f}_3(x_3) = \left(\sum_{x_1,x_2} f_1(x_1) f_2(x_2) f_3(x_1,x_2,x_3) \right)$$

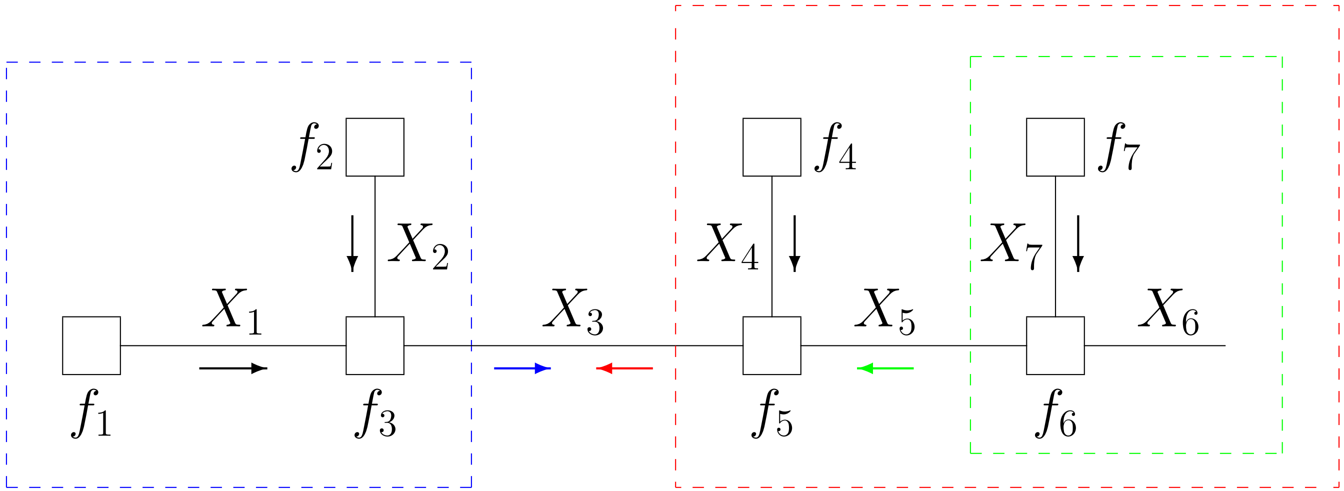
$$\cdot \left(\sum_{x_4,x_5} f_4(x_4) f_5(x_3,x_4,x_5) \left(\sum_{x_6,x_7} f_6(x_5,x_6,x_7) f_7(x_7) \right) \right)$$

Example cont'd: **Message Passing View**



$$\bar{f}_3(x_3) = \underbrace{\left(\sum_{x_1,x_2} f_1(x_1) f_2(x_2) f_3(x_1,x_2,x_3) \right)}_{\overrightarrow{\mu}_{X_3}(x_3)} \cdot \underbrace{\left(\sum_{x_4,x_5} f_4(x_4) f_5(x_3,x_4,x_5) \underbrace{\left(\sum_{x_6,x_7} f_6(x_5,x_6,x_7) f_7(x_7) \right)}_{\overleftarrow{\mu}_{X_5}(x_5)} \right)}_{\overleftarrow{\mu}_{X_3}(x_3)}$$

Example cont'd: **Messages Everywhere**



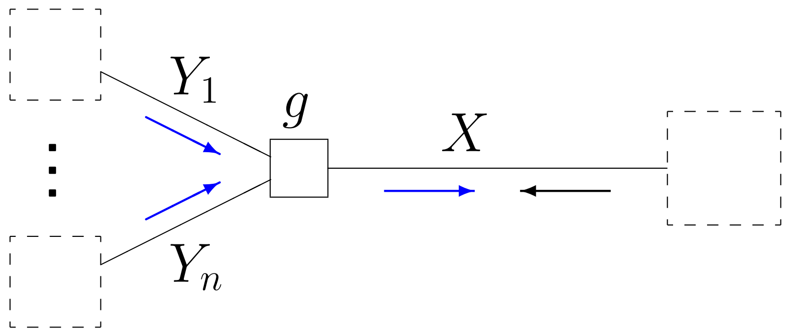
With $\overrightarrow{\mu}_{X_1}(x_1) \triangleq f_1(x_1)$, $\overrightarrow{\mu}_{X_2}(x_2) \triangleq f_2(x_2)$, etc., we have

$$\overrightarrow{\mu}_{X_3}(x_3) = \sum_{x_1,x_2} \overrightarrow{\mu}_{X_1}(x_1)\,\overrightarrow{\mu}_{X_2}(x_2)\,f_3(x_1,x_2,x_3)$$

$$\overleftarrow{\mu}_{X_5}(x_5) = \sum_{x_6,x_7} \overrightarrow{\mu}_{X_7}(x_7)\,f_6(x_5,x_6,x_7)$$

$$\overleftarrow{\mu}_{X_3}(x_3) = \sum_{x_4,x_5} \overrightarrow{\mu}_{X_4}(x_4)\,\overleftarrow{\mu}_{X_5}(x_5)\,f_5(x_3,x_4,x_5)$$

The Sum-Product Algorithm (Belief Propagation)



Sum-product message computation rule:

$$\overrightarrow{\mu}_X(x) = \sum_{y_1,\ldots,y_n} g(x, y_1, \ldots, y_n)\overrightarrow{\mu}_{Y_1}(y_1)\cdots\overrightarrow{\mu}_{Y_n}(y_n)$$

Sum-product theorem:

If the factor graph for some function f has no cycles, then

$$\bar{f}_X(x) = \overrightarrow{\mu}_X(x)\overleftarrow{\mu}_X(x).$$

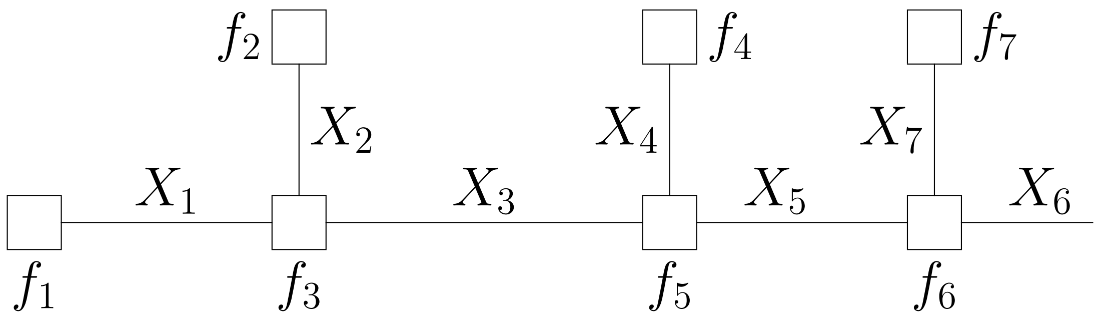
Towards the max-product algorithm:

Computing Max-Marginals—A Generic Example

Assume we wish to compute

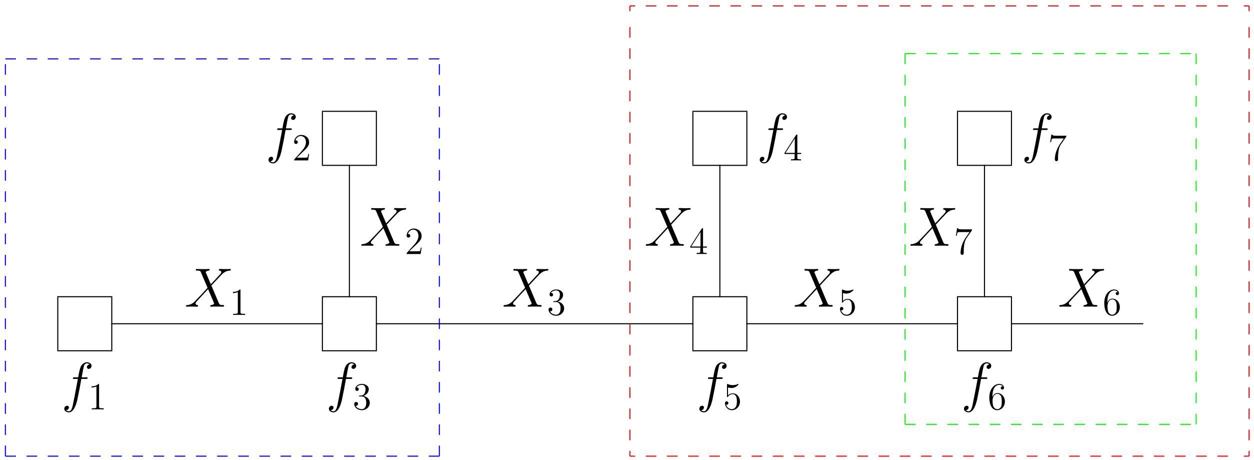
$$\hat{f}_3(x_3) = \max_{\substack{x_1, \ldots, x_7 \\ \text{except } x_3}} f(x_1, \ldots, x_7)$$

and assume that f can be factored as follows:



Example:

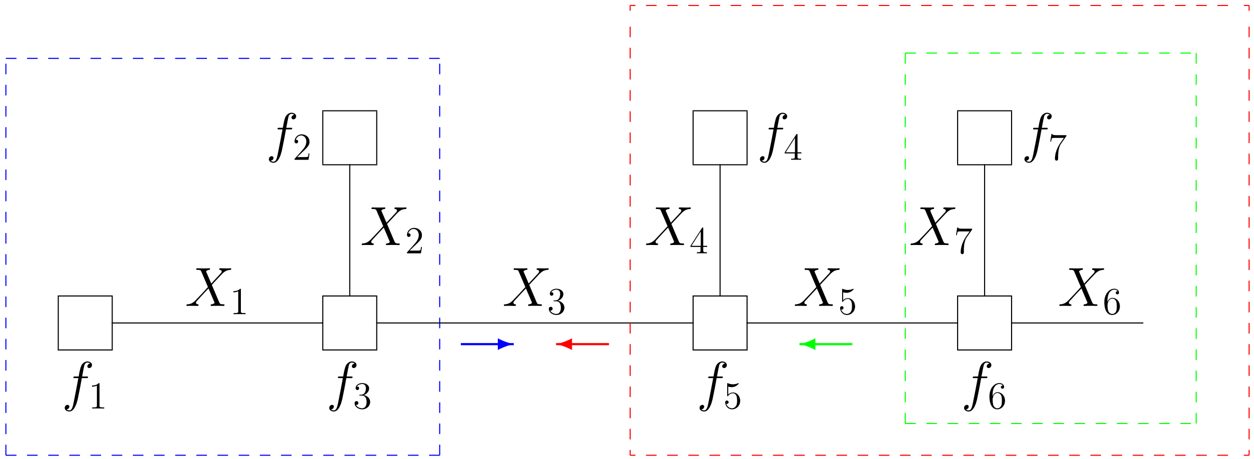
Closing Boxes by the Distributive Law



$$\hat{f}_3(x_3) = \left(\max_{x_1,x_2} f_1(x_1) f_2(x_2) f_3(x_1, x_2, x_3) \right)$$

$$\cdot \left(\max_{x_4,x_5} f_4(x_4) f_5(x_3, x_4, x_5) \left(\max_{x_6,x_7} f_6(x_5, x_6, x_7) f_7(x_7) \right) \right)$$

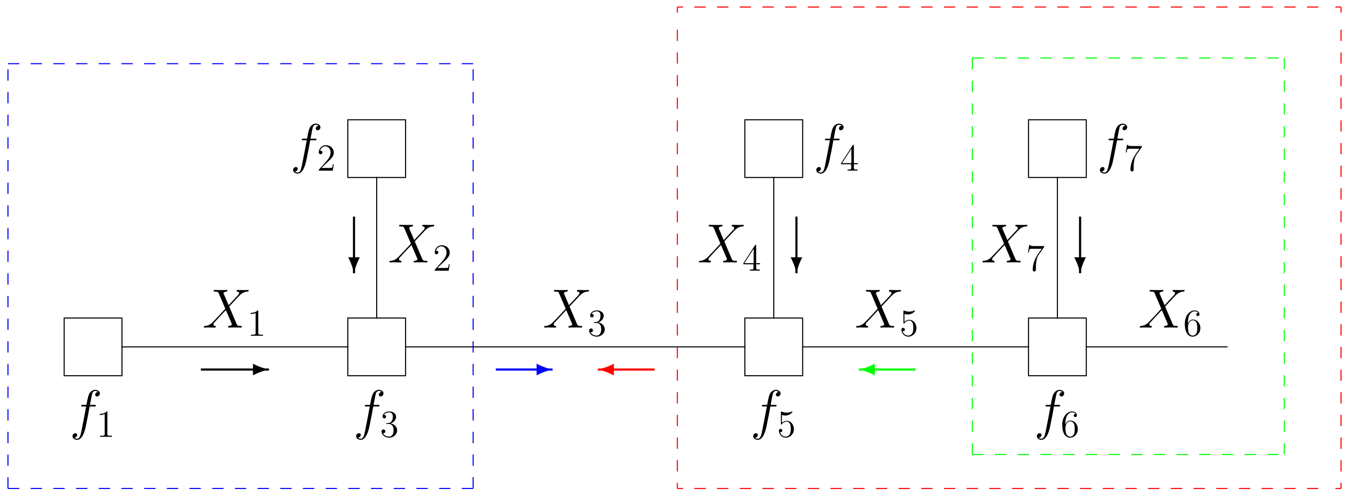
Example cont'd: **Message Passing View**



$$\hat{f}_3(x_3) = \left(\underbrace{\max_{x_1,x_2} f_1(x_1) f_2(x_2) f_3(x_1,x_2,x_3)}_{\overrightarrow{\mu}_{X_3}(x_3)} \right)$$

$$\cdot \underbrace{\left(\max_{x_4,x_5} f_4(x_4) f_5(x_3,x_4,x_5) \left(\underbrace{\max_{x_6,x_7} f_6(x_5,x_6,x_7) f_7(x_7)}_{\overleftarrow{\mu}_{X_5}(x_5)} \right) \right)}_{\overleftarrow{\mu}_{X_3}(x_3)}$$

Example cont'd: **Messages Everywhere**



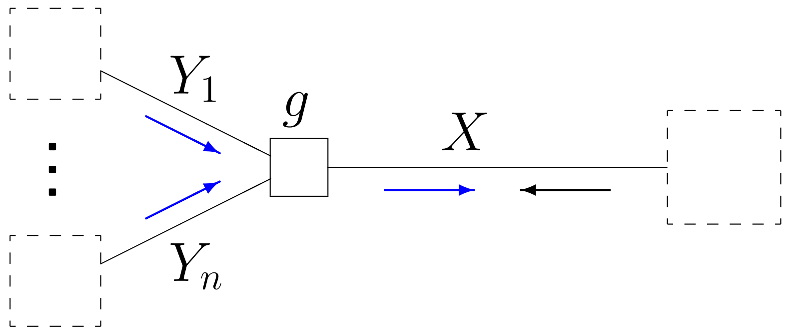
With $\overrightarrow{\mu}_{X_1}(x_1) \stackrel{\triangle}{=} f_1(x_1)$, $\overrightarrow{\mu}_{X_2}(x_2) \stackrel{\triangle}{=} f_2(x_2)$, etc., we have

$$\overrightarrow{\mu}_{X_3}(x_3) = \max_{x_1,x_2} \overrightarrow{\mu}_{X_1}(x_1)\overrightarrow{\mu}_{X_2}(x_2) f_3(x_1, x_2, x_3)$$

$$\overleftarrow{\mu}_{X_5}(x_5) = \max_{x_6,x_7} \overrightarrow{\mu}_{X_7}(x_7) f_6(x_5, x_6, x_7)$$

$$\overleftarrow{\mu}_{X_3}(x_3) = \max_{x_4,x_5} \overrightarrow{\mu}_{X_4}(x_4)\overleftarrow{\mu}_{X_5}(x_5) f_5(x_3, x_4, x_5)$$

The Max-Product Algorithm



Max-product message computation rule:

$$\overrightarrow{\mu}_X(x) = \max_{y_1,\ldots,y_n} g(x, y_1, \ldots, y_n)\,\overrightarrow{\mu}_{Y_1}(y_1) \cdots \overrightarrow{\mu}_{Y_n}(y_n)$$

Max-product theorem:

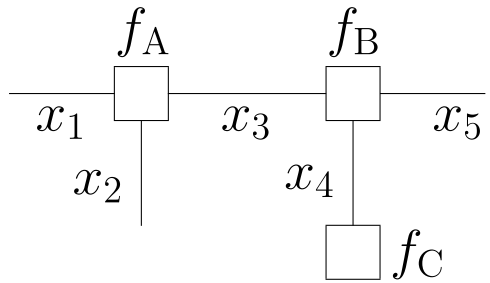
If the factor graph for some global function f has no cycles, then

$$\hat{f}_X(x) = \overrightarrow{\mu}_X(x)\overleftarrow{\mu}_X(x).$$

Outline

1. Introduction
2. The sum-product and max-product algorithms
3. More about factor graphs
4. Applications of sum-product & max-product to hidden Markov models
5. Graphs with cycles, continuous variables, . . .

Terminology



Global function $f =$ product of all factors; usually (but not always!) real and nonnegative.

A configuration is an assignment of values to all variables.

The configuration space is the set of all configurations, which is the domain of the global function.

A configuration $\omega = (x_1, \ldots, x_5)$ is valid iff $f(\omega) \neq 0$.

Invalid Configurations Do Not Affect Marginals

A configuration $\omega = (x_1, \ldots, x_n)$ is valid iff $f(\omega) \neq 0$.

Recall:

1. Marginalization: Compute

$$\bar{f}_k(x_k) \triangleq \sum_{\substack{x_1, \ldots, x_n \\ \text{except } x_k}} f(x_1, \ldots, x_n)$$

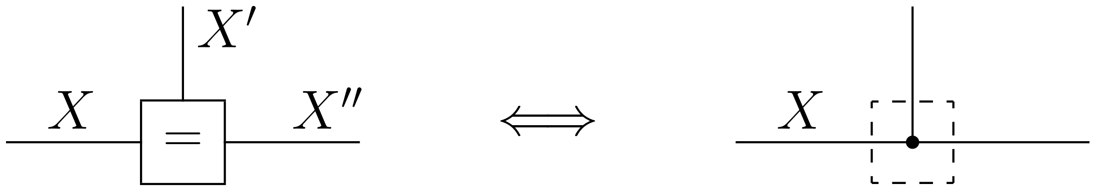
2. Maximization: Compute the "max-marginal"

$$\hat{f}_k(x_k) \triangleq \max_{\substack{x_1, \ldots, x_n \\ \text{except } x_k}} f(x_1, \ldots, x_n)$$

assuming that f is real-valued and nonnegative and has a maximum.

Constraints may be expressed by factors that evaluate to 0 if the constraint is violated.

Branching Point = Equality Constraint Factor



The factor

$$f_=(x, x', x'') \triangleq \begin{cases} 1, & \text{if } x = x' = x'' \\ 0, & \text{else} \end{cases}$$

enforces $X = X' = X''$ for every valid configuration.

More general:

$$f_=(x, x', x'') \triangleq \delta(x - x')\delta(x - x'')$$

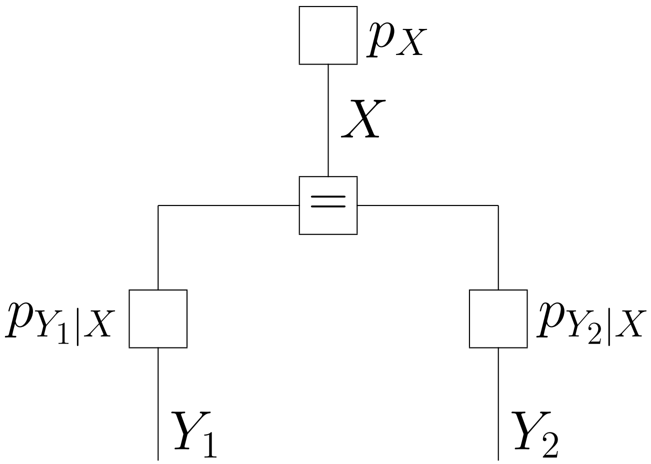
where δ denotes the Kronecker delta for discrete variables and the Dirac delta for discrete variables.

Example:

Independent Observations

Let Y_1 and Y_2 be two independent observations of X:

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



Literally, the factor graph represents an extended model

$$p(x, x', x'', y_1, y_2) = p(x)p(y_1|x')p(y_2|x'')f_=(x, x', x'')$$

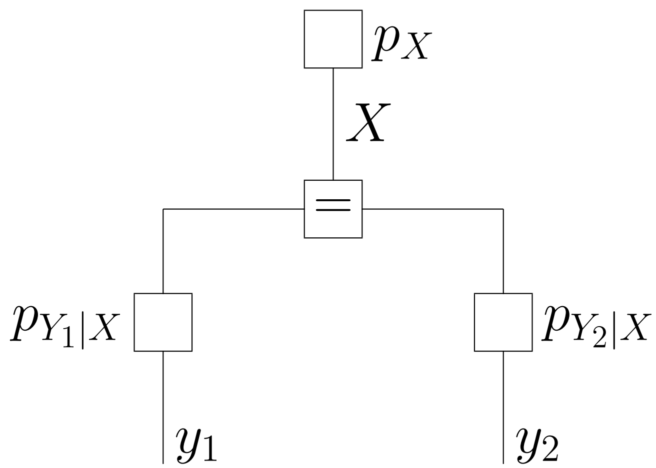
with the same marginals and max-marginals as $p(x, y_1, y_2)$.

From A Priori to A Posteriori Probability

Example (cont'd): Let $Y_1 = y_1$ and $Y_2 = y_2$ be two independent observations of X, i.e., $p(x, y_1, y_2) = p(x)p(y_1|x)p(y_2|x)$.

For fixed y_1 and y_2, we have

$$p(x|y_1, y_2) \;=\; \frac{p(x, y_1, y_2)}{p(y_1, y_2)} \;\propto\; p(x, y_1, y_2).$$



The factorization is unchanged (except for a scale factor).

Known variables will be denoted by small letters;
unknown variables will usually be denoted by capital letters.

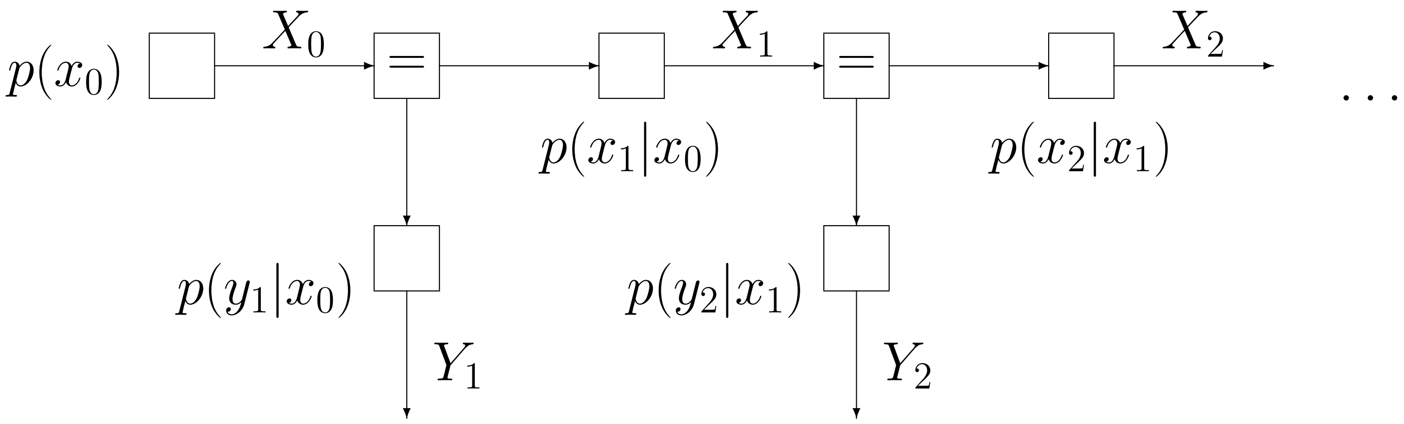
Outline

1. Introduction
2. The sum-product and max-product algorithms
3. More about factor graphs
4. Applications of sum-product & max-product
   to hidden Markov models
5. Graphs with cycles, continuous variables, . . .

Example:

Hidden Markov Model

$$p(x_0, x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n) = p(x_0) \prod_{k=1}^{n} p(x_k|x_{k-1}) p(y_k|x_{k-1})$$

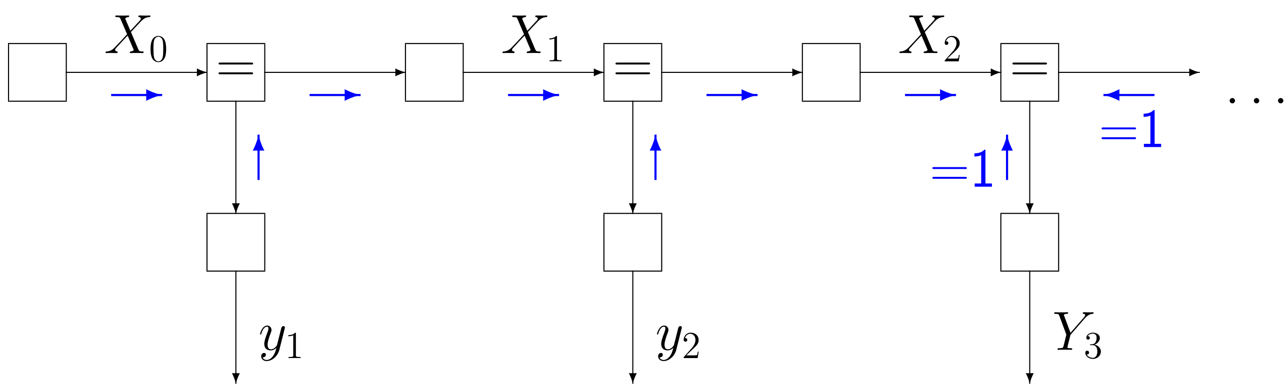


Sum-product algorithm applied to HMM:

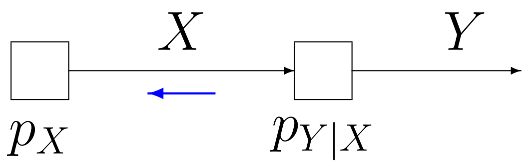
Estimation of Current State

$$
\begin{aligned}
p(x_n|y_1,\ldots,y_n) &= \frac{p(x_n,y_1,\ldots,y_n)}{p(y_1,\ldots,y_n)} \\
&\propto p(x_n,y_1,\ldots,y_n) \\
&= \sum_{x_0}\cdots\sum_{x_{n-1}} p(x_0,x_1,\ldots,x_n,y_1,y_2,\ldots,y_n) \\
&= \overrightarrow{\mu}_{X_n}(x_n).
\end{aligned}
$$

For $n=2$:



Backward Message in Chain Rule Model



If $Y = y$ is known (observed):

$$\overleftarrow{\mu}_X(x) = p_{Y|X}(y|x),$$

the likelihood function.

If Y is unknown:

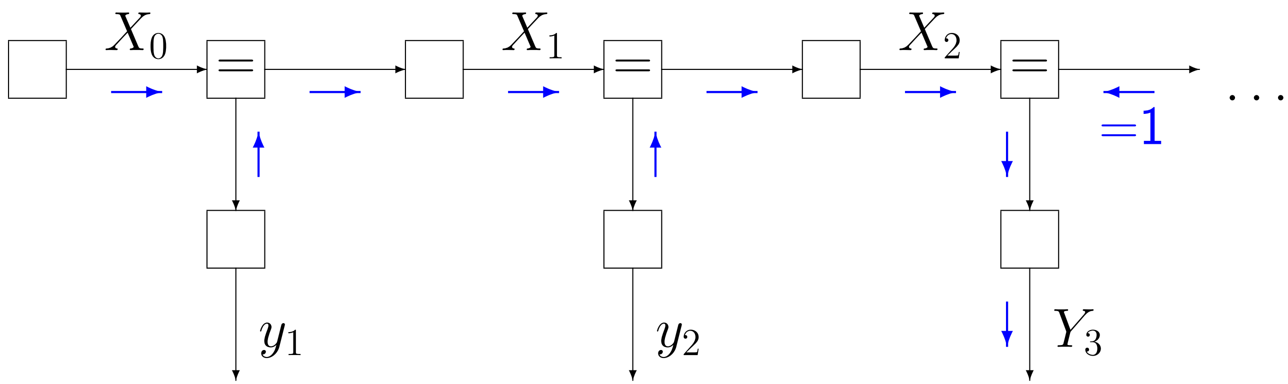
$$\begin{aligned}\overleftarrow{\mu}_X(x) &= \sum_y p_{Y|X}(y|x) \\ &= 1.\end{aligned}$$

Sum-product algorithm applied to HMM:

Prediction of Next Output Symbol

$$
\begin{aligned}
p(y_{n+1}|y_1,\ldots,y_n) &= \frac{p(y_1,\ldots,y_{n+1})}{p(y_1,\ldots,y_n)} \\
&\propto p(y_1,\ldots,y_{n+1}) \\
&= \sum_{x_0,x_1,\ldots,x_n} p(x_0,x_1,\ldots,x_n,y_1,y_2,\ldots,y_n,y_{n+1}) \\
&= \overrightarrow{\mu}_{Y_n}(y_n).
\end{aligned}
$$

For $n=2$:

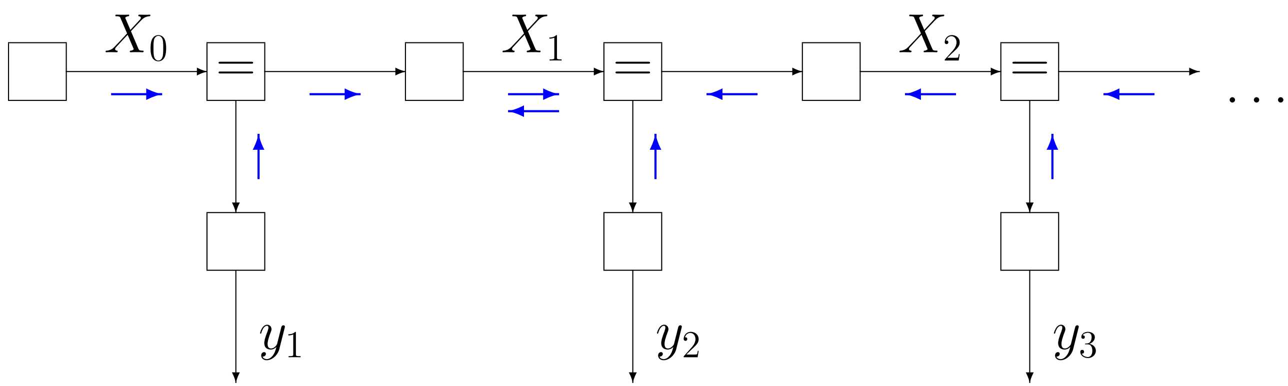


Sum-product algorithm applied to HMM:

Estimation of Time-k State

$$
\begin{aligned}
p(x_k \mid y_1, y_2, \ldots, y_n) &= \frac{p(x_k, y_1, y_2, \ldots, y_n)}{p(y_1, y_2, \ldots, y_n)} \\
&\propto p(x_k, y_1, y_2, \ldots, y_n) \\
&= \sum_{\substack{x_0, \ldots, x_n \\ \text{except } x_k}} p(x_0, x_1, \ldots, x_n, y_1, y_2, \ldots, y_n) \\
&= \overrightarrow{\mu}_{X_k}(x_k)\, \overleftarrow{\mu}_{X_k}(x_k)
\end{aligned}
$$

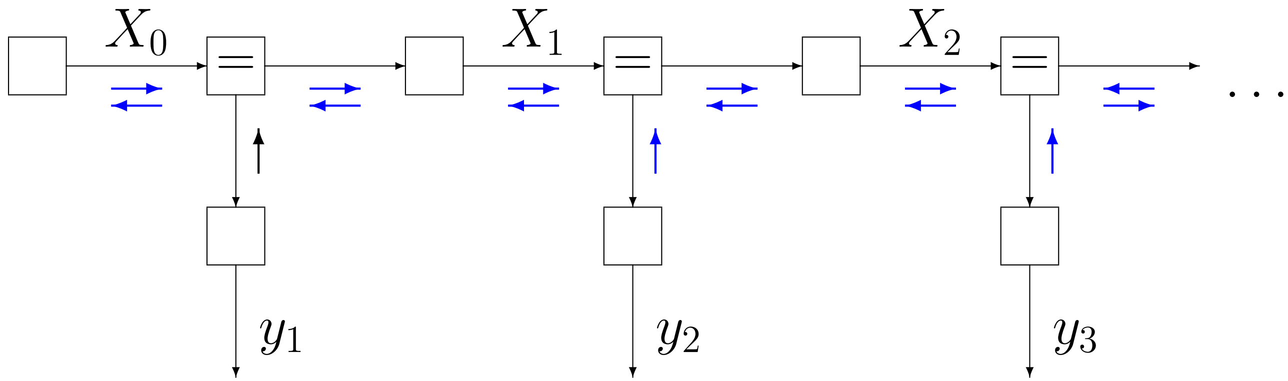
For $k = 1$:



Sum-product algorithm applied to HMM:

All States Simultaneously

$p(x_k|y_1,\ldots,y_n)$ for all k:



In this application, the sum-product algorithm coincides with the Baum-Welch / BCJR forward-backward algorithm.

Scaling of Messages

In all the examples so far:

- The final result (such as $\overrightarrow{\mu}_{X_k}(x_k)\overleftarrow{\mu}_{X_k}(x_k)$) equals the desired quantity (such as $p(x_k|y_1, \ldots, y_n)$) only up to a scale factor.
- The missing scale factor γ may be recovered at the end from the condition

$$\sum_{x_k} \gamma \overrightarrow{\mu}_{X_k}(x_k)\overleftarrow{\mu}_{X_k}(x_k) = 1.$$

- It follows that messages may be scaled freely along the way.
- Such message scaling is often mandatory to avoid numerical problems.

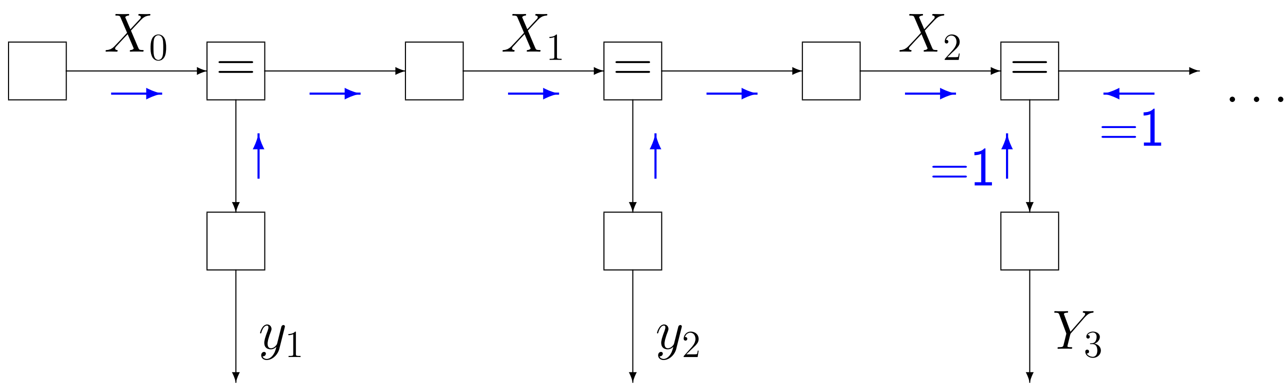
Sum-product algorithm applied to HMM:

Probability of the Observation

$$p(y_1, \ldots, y_n) = \sum_{x_0} \ldots \sum_{x_n} p(x_0, x_1, \ldots, x_n, y_1, y_2, \ldots, y_n)$$
$$= \sum_{x_n} \overrightarrow{\mu}_{X_n}(x_n).$$

This is a number. Scale factors cannot be neglected in this case.

For $n = 2$:



Max-product algorithm applied to HMM:

MAP Estimate of the State Trajectory

The estimate

$$\begin{aligned}(\hat{x}_0, \ldots, \hat{x}_n)_{\text{MAP}} &= \operatorname*{argmax}_{x_0,\ldots,x_n} p(x_0, \ldots, x_n | y_1, \ldots, y_n) \\ &= \operatorname*{argmax}_{x_0,\ldots,x_n} p(x_0, \ldots, x_n, y_1, \ldots, y_n)\end{aligned}$$

may be obtained by computing

$$\begin{aligned}\hat{p}_k(x_k) &\stackrel{\triangle}{=} \max_{\substack{x_1, \ldots, x_n \\ \text{except } x_k}} p(x_0, \ldots, x_n, y_1, \ldots, y_n) \\ &= \overrightarrow{\mu}_{X_k}(x_k) \overleftarrow{\mu}_{X_k}(x_k)\end{aligned}$$

for all k by forward-backward max-product sweeps.

In this example, the max-product algorithm is a time-symmetric version of the Viterbi algorithm with soft output.

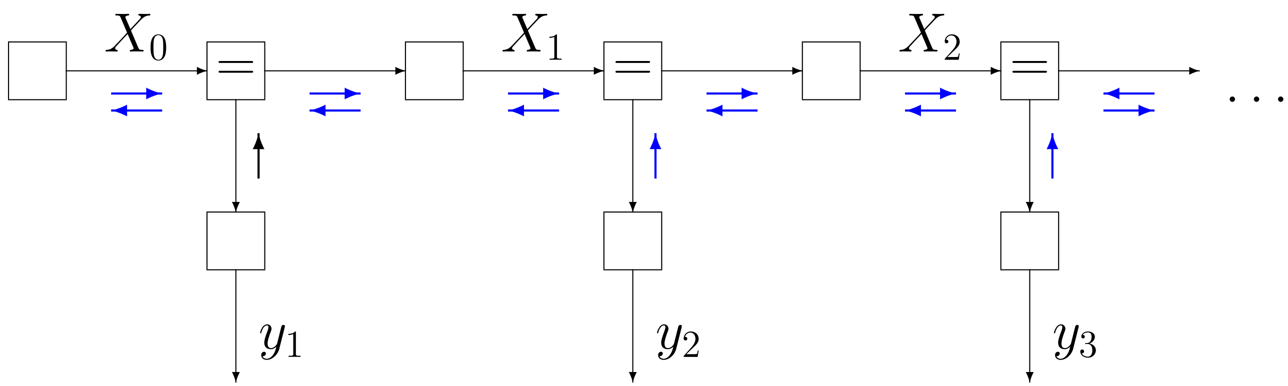
Max-product algorithm applied to HMM:

MAP Estimate of the State Trajectory cont'd

Computing

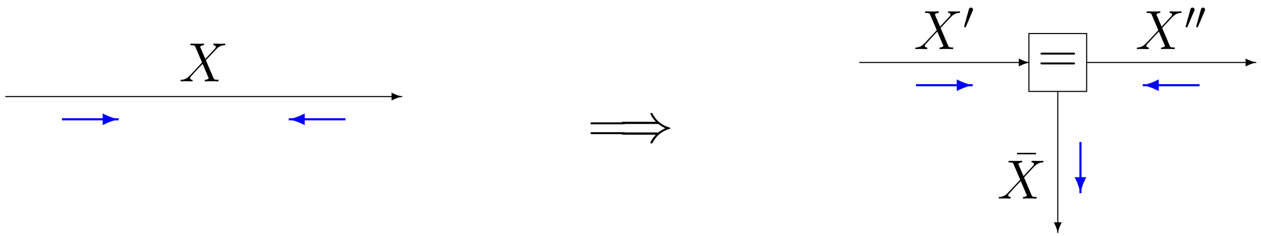
$$\begin{aligned}\hat{p}_k(x_k) &\triangleq \max_{\substack{x_1,\ldots,x_n\\ \text{except } x_k}} p(x_0,\ldots,x_n,y_1,\ldots,y_n)\\ &= \overrightarrow{\mu}_{X_k}(x_k)\overleftarrow{\mu}_{X_k}(x_k)\end{aligned}$$

simultaneously for all k:



Marginals and Output Edges

Marginals such $\overrightarrow{\mu}_X(x)\overleftarrow{\mu}_X(x)$ may be viewed as messages out of a "output half edge" (without incoming message):



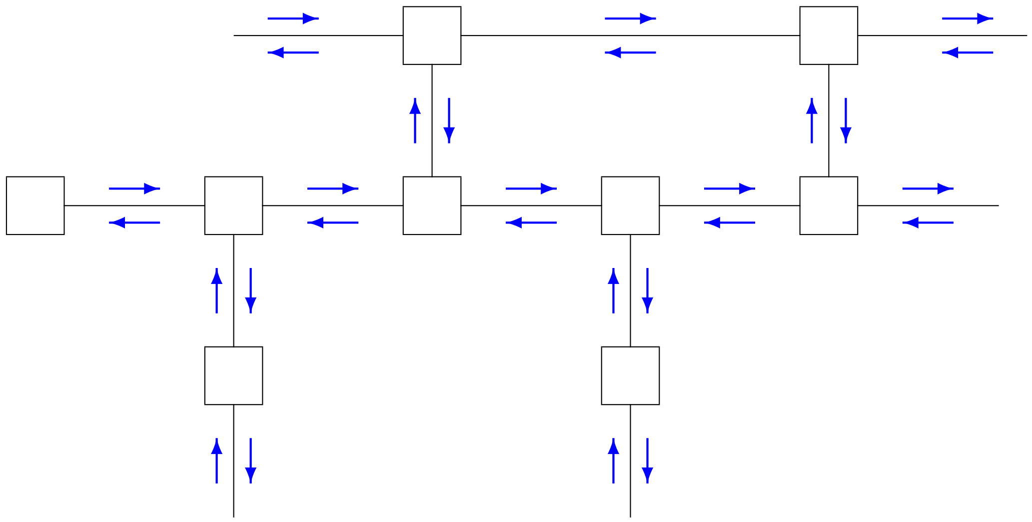
$$
\begin{aligned}
\overrightarrow{\mu}_{\bar{X}}(x) &= \int_{x'} \int_{x''} \overrightarrow{\mu}_{X'}(x')\,\overleftarrow{\mu}_{X''}(x'')\,\delta(x-x')\,\delta(x-x'')\,dx'\,dx'' \\
&= \overrightarrow{\mu}_{X'}(x)\,\overleftarrow{\mu}_{X''}(x)
\end{aligned}
$$

$\Longrightarrow$ Marginals are computed like messages out of "$=$"-nodes.

Outline

1. Introduction
2. The sum-product and max-product algorithms
3. More about factor graphs
4. Applications of sum-product & max-product to hidden Markov models
5. Graphs with cycles, continuous variables, . . .

Factor Graphs with Cycles? Continuous Variables?



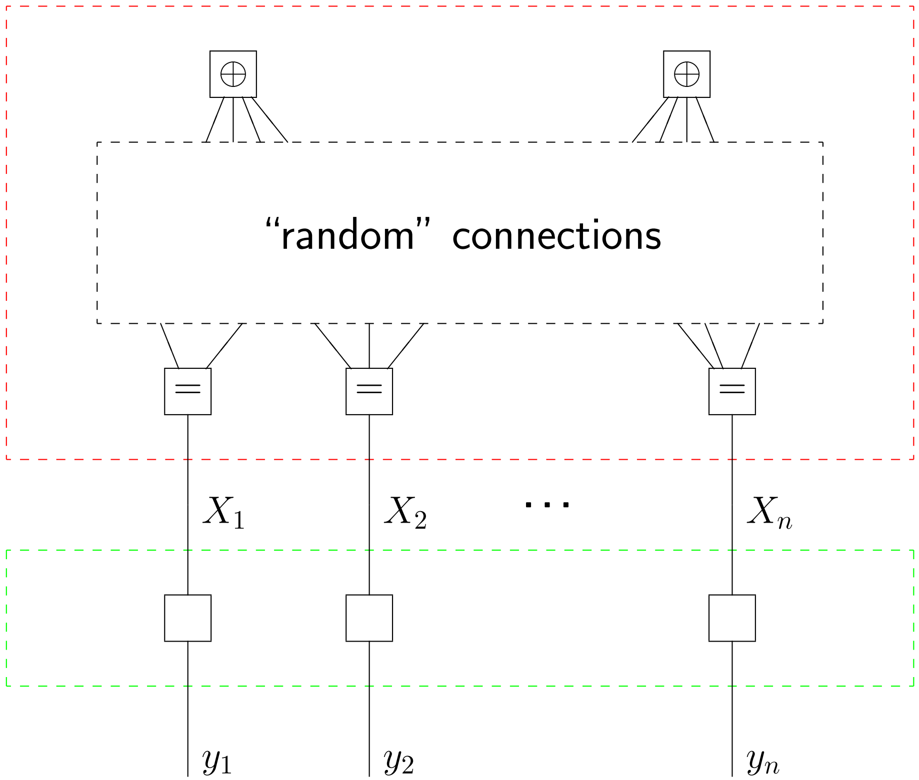
Example:

Error Correcting Codes (e.g. LDPC Codes)

Codeword $x = (x_1, \ldots, x_n) \in C \subset \{0,1\}^n$. Received $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$.

$p(x, y) \propto p(y|x)\delta_C(x)$.

$$f_\oplus(z_1, \ldots, z_m) = \begin{cases} 1, \text{if } z_1 + \ldots + z_m \equiv 0 \bmod 2 \\ 0, \text{else} \end{cases}$$



code membership function

$$\delta_C(x) = \begin{cases} 1, & \text{if } x \in C \\ 0, & \text{else} \end{cases}$$

channel model $p(y|x)$

e.g., $p(y|x) = \prod_{\ell=1}^{n} p(y_\ell|x_\ell)$

Decoding by iterative sum-product message passing.

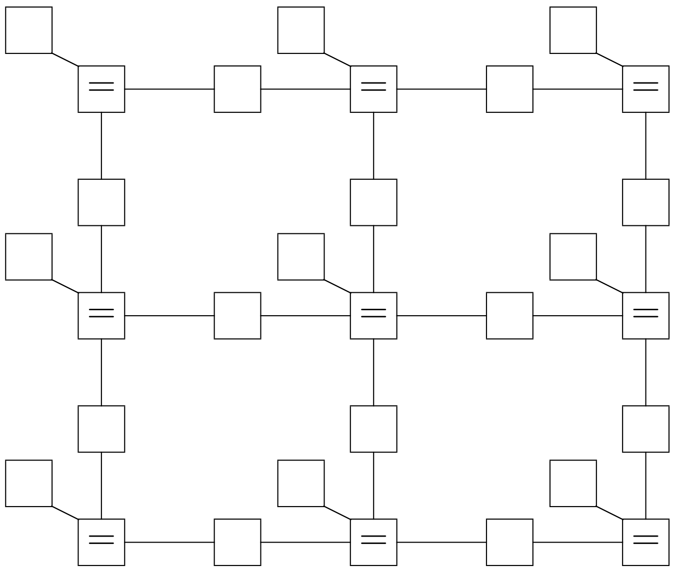
Example:

Magnets, Spin Glasses, etc.

Configuration $x = (x_1, \ldots, x_n)$, $x_k \in \{0, 1\}$.

"Energy" $w(x) = \sum_k \beta_k x_k + \sum_{\text{neighboring pairs } (k,\ell)} \beta_{k,\ell}(x_k - x_\ell)^2$

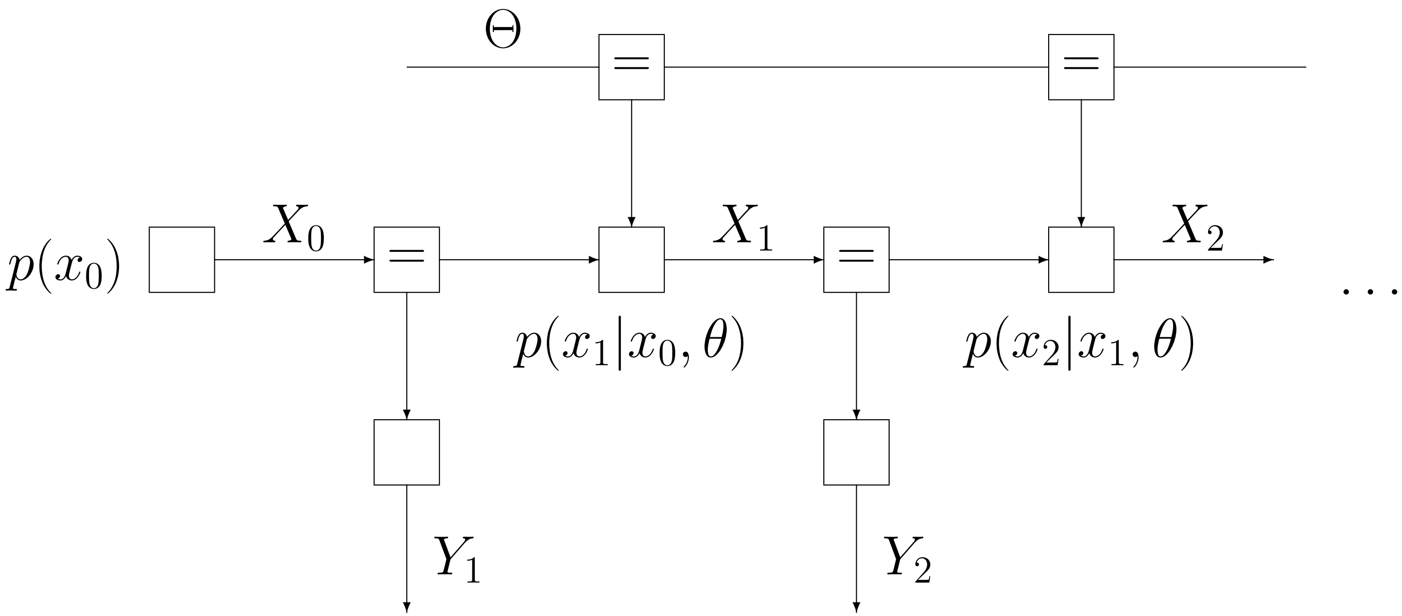
$$p(x) \propto e^{-w(x)} = \prod_k e^{-\beta_k x_k} \prod_{\text{neighboring pairs } (k,\ell)} e^{-\beta_{k,\ell}(x_k - x_\ell)^2}$$



Example:

Hidden Markov Model with Parameter(s)

$$p(x_0, x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n \mid \theta) = p(x_0) \prod_{k=1}^{n} p(x_k|x_{k-1}, \theta) p(y_k|x_{k-1})$$



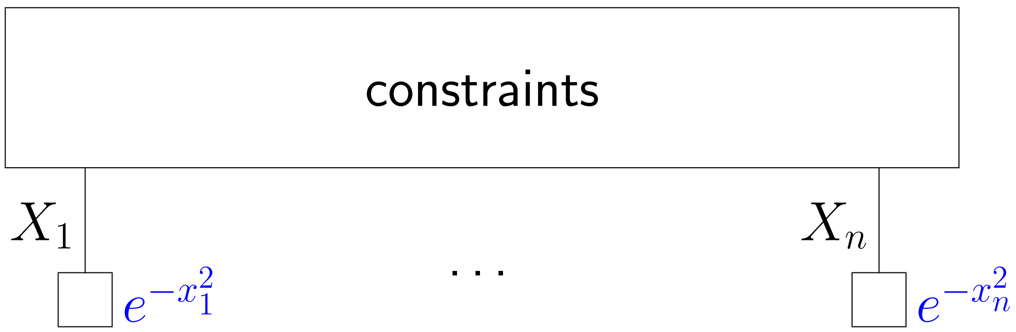
Example:

Least-Squares Problems

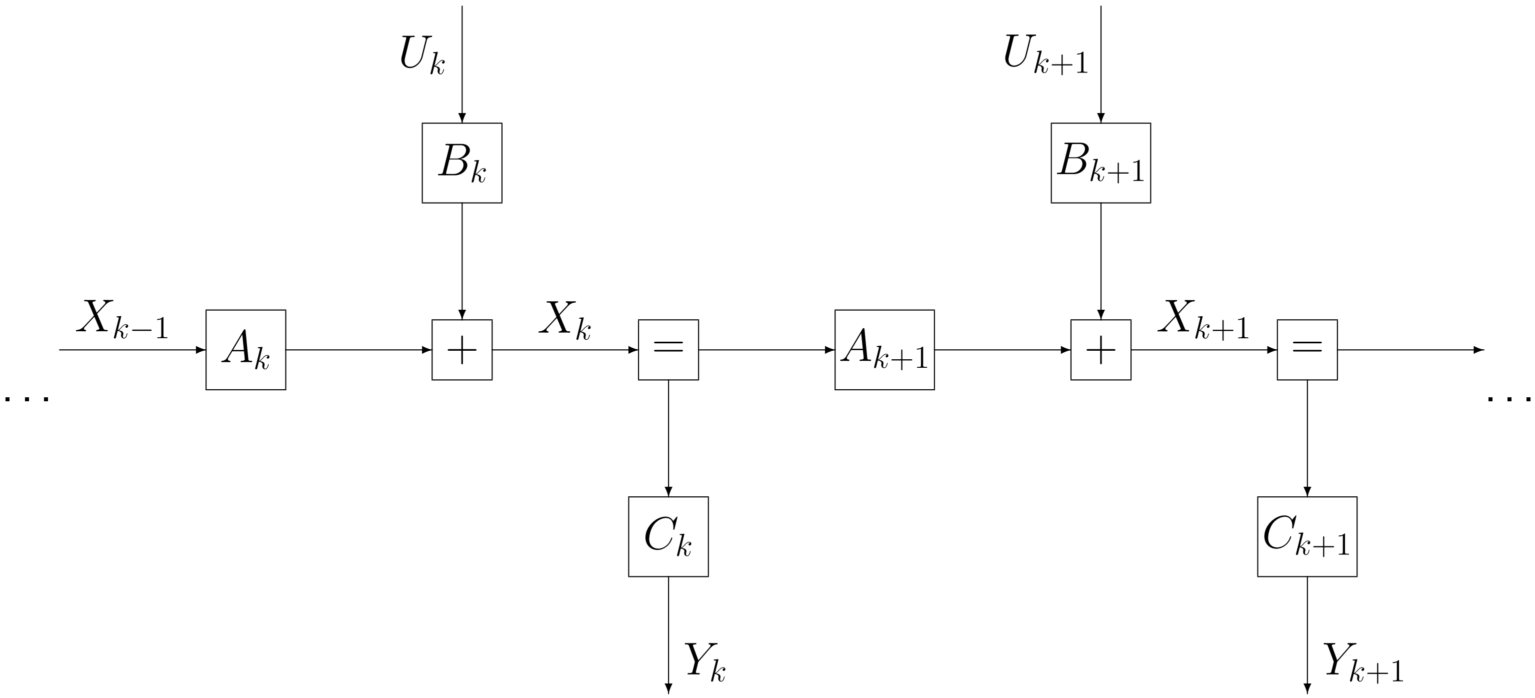
Minimizing $\sum_{k=1}^{n} x_k^2$ subject to (linear or nonlinear) constraints is equivalent to maximizing

$$e^{-\sum_{k=1}^{n} x_k^2} = \prod_{k=1}^{n} e^{-x_k^2}$$

subject to the given constraints.



General Linear State Space Model



$$
\begin{aligned}
X_k &= A_k X_{k-1} + B_k U_k \\
Y_k &= C_k X_k
\end{aligned}
$$

Gaussian Message Passing in Linear Models

encompasses much of classical signal processing and appears often as a component of more complex problems/algorithms.

Note:

1. Gaussianity of messages is preserved in linear models.
2. Includes Kalman filtering and recursive least-squares algorithms.
3. For Gaussian messages, the sum-product (integral-product) algorithm coincides with the max-product algorithm.
4. For jointly Gaussian random variables,
   MAP estimation = MMSE estimation = LMMSE estimation.
5. Even if X and Y are not jointly Gaussian, the LMMSE estimate of X from $Y = y$ may be obtained by pretending that X and Y are jointly Gaussian (with their actual means and second-order moments) and forming the corresponding MAP estimate.

See "The factor graph approach to model-based signal processing," *Proceedings of the IEEE,* June 2007.

Continuous Variables: Message Types

The following message types are widely applicable.

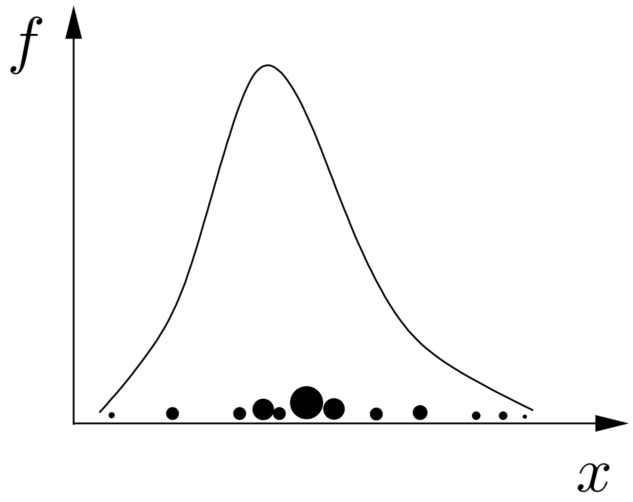
- Quantization of messages into discrete bins. Infeasible in higher dimensions.
- Single point: the message $\mu(x)$ is replaced by a temporary or final estimate $\hat{x}$.
- Function value and gradient at a point selected by the receiving node. Allows steepest-ascent (descent) algorithms.
- Gaussians. Works for Kalman filtering.
- Gaussian mixtures.
- List of samples: a pdf can be represented by a list of samples. This data type is the basis of particle filters, but it can be used as a general data type in a graphical model.
- Compound messages: the "product" of other message types.

Particle Methods as Message Passing

Basic idea: represent a probability density f by a list

$$\mathcal{L} = \Big((\hat{x}^{(1)}, w^{(1)}), \ldots (\hat{x}^{(L)}, w^{(L)}) \Big)$$

of weighted samples ("particles"):



- Versatile data type for sum-product, max-product, EM, . . .
- Not sensitive to dimensionality.

On Factor Graphs with Cycles

- Generally iterative algorithms.
- For example, alternating maximization

$$\hat{x}_{\text{new}} = \operatorname*{argmax}_{x} f(x, \hat{y}) \quad \text{and} \quad \hat{y}_{\text{new}} = \operatorname*{argmax}_{y} f(\hat{x}, y)$$

  using the max-product algorithm in each iteration.
- Iterative sum-product message passing gives excellent results for maximization(!) in some applications (e.g., the decoding of error correcting codes).
- Many other useful algorithms can be formulated in message passing form (e.g., J. Dauwels): gradient ascent, Gibbs sampling, expectation maximization, variational methods, clustering by "affinity propagation" (B. Frey) ...
- Factor graphs facilitate to mix and match different algorithmic techniques.

End of this talk.