Hasso Plattner Institut
Digital Engineering · Universität Potsdam

Introduction to Probabilistic Machine Learning

Ralf Herbrich, Rainer Schlosser

Graphical Models: Exact Inference

Overview

1. Factor Graphs
2. Marginalization by Importance Sampling
3. Marginalization by the Sum-Product Algorithm
4. Practical Considerations in Message Passing

Overview

1. **Factor Graphs**
2. Marginalization by Importance Sampling
3. Marginalization by the Sum-Product Algorithm
4. Practical Considerations in Message Passing

Inference in Probabilistic Models

- **Inference**: In order to learn from data D we follow a three-step procedure
  1. **Modelling**: Formulate a joint model $p(\boldsymbol{\theta}, D)$ of parameters $\boldsymbol{\theta} = \theta_1, \dots, \theta_n$ and data D
  2. **Conditioning**: Clamp the variables that represent data D (as they are observed)
  3. **Marginalize**: Sum-out all variables that we are not interested in (latent parameters)
- **Example**: Two player game with one winner
  1. **Modelling**: Parameters $\boldsymbol{\theta} = (s_1, s_2, p_1, p_2)$ are skills and performances; data is $y \in \{-1, +1\}$

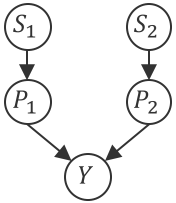
$$p(s_1, s_2, p_1, p_2, y) = \mathcal{N}(s_1; \mu_1, \sigma_1^2) \cdot \mathcal{N}(s_2; \mu_2, \sigma_2^2) \cdot \mathcal{N}(p_1; s_1, \beta^2) \cdot \mathcal{N}(p_2; s_2, \beta^2) \cdot \mathbb{I}(y(p_1 - p_2) > 0)$$

  2. **Conditioning**: Player 1 wins ($y = 1$)

$$p(s_1, s_2, p_1, p_2 | y = 1) \propto \mathcal{N}(s_1; \mu_1, \sigma_1^2) \cdot \mathcal{N}(s_2; \mu_2, \sigma_2^2) \cdot \mathcal{N}(p_1; s_1, \beta^2) \cdot \mathcal{N}(p_2; s_2, \beta^2) \cdot \mathbb{I}(p_1 - p_2 > 0)$$

  3. **Marginalize**: We are only interested in the skills and need to sum-out p_1 and p_2

$$p(s_1, s_2 | y = 1) \propto p(s_1, s_2) \cdot \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \mathcal{N}(p_1; s_1, \beta^2) \cdot \mathcal{N}(p_2; s_2, \beta^2) \cdot \mathbb{I}(p_1 - p_2 > 0) \, \mathrm{d}p_1 \mathrm{d}p_2$$



Factors, Variables and Probabilistic Inference

- **Observation I**: The joint probability model of data and parameters is a *product* of conditional probabilities and has many factors with (few) variables!
- **Observation II**: Conditioning does not reduce factors; it removes variables!
- **Problem**: Naïve summation scales exponentially because we have a sum of products (i.e., product of conditional disitrubtions of all latent variables)!
  - **Example**: Consider an example of n Bernoulli variables $x_1, \dots, x_n$

$$p(x_1) = \underbrace{\sum_{x_2=0}^{1} \sum_{x_3=0}^{1} \cdots \sum_{x_n=0}^{1}}_{2^{n-1} \text{ summations}} p(x_1, x_2, \dots, x_n)$$

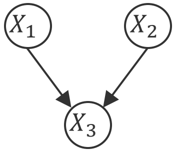
- **Idea**: We exploit the product structure of the probabilisitic model of our data and parameters because not every variable depends on all other variables
  - **Example (ctd)**. Consider $p(x_1, x_2, \dots, x_n) = \prod_i p(x_i)$: then there are only $O(n)$ sums and $n-1$ sum to one!

Factor Graphs

- **Factor Graph (Frey, 1998).** *Given a product of m positive functions $f_1, \dots, f_m$, each over a subset of n variables $X_1, X_2, \dots, X_n$, a factor graph is a bipartite graphical model with m factor nodes and n variable nodes where an undirected edge connects f_i and X_j if and only if the function f_i depends on the value of X_j.*
- Factor graphs are more expressive than a Bayesian network!

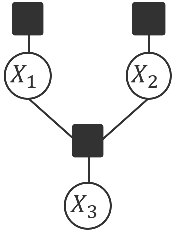
Brendan Frey
(1968 –)

Bayesian network



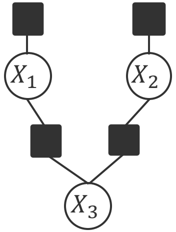
$p(x_1, x_2, x_3) = p(x_1) \cdot p(x_2) \cdot p(x_3 | x_1, x_2)$

Corresponding factor graph



$p(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3, x_1, x_2)$

Factor graph with more structure



$p(x_1, x_2, x_3) = f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_1, x_3) \cdot f_4(x_2, x_3)$

Structure in $p(x_3 | x_1, x_2)$

Overview

1. Factor Graphs
2. **Marginalization by Importance Sampling**
3. Marginalization by the Sum-Product Algorithm
4. Practical Considerations in Message Passing

Inference by Importance Sampling

- The key operation is summing-out all but one variable (marginalization).
- **Idea**: If we can get J samples $x_{j,1}, \dots, x_{j,n}$, $j \in \{1, \dots, J\}$ which are drawn according to
$$p(x_1, \dots, x_n) \propto \prod_{i=1}^{m} f_i(x_1, \dots, x_n)$$
then we can approximate the marginals $p(x_k)$ arbitrarily well via
$$p(x_k) \approx \frac{1}{J} \sum_{j=1}^{J} \delta(x_k - x_{j,k})$$
- **Challenge**: How do we sample $p(x_1, \dots, x_n)$ if we only have access to (a few) known efficient samplers $q(x_1, \dots, x_n)$ such as (pseudo-random) numbers from the uniform distribution or normal distribution over each X_k?
- **Importance Sampling**: We get J samples $x_{j,1}, \dots, x_{j,n}$, $j \in \{1, \dots, J\}$ from $q(x_1, \dots, x_n)$ and re-weight them with

$$\frac{p(x_{j,1}, \dots, x_{j,n})}{q(x_{j,1}, \dots, x_{j,n})} \longleftarrow \text{Importance weight}$$

Weighted-Empirical Distribution

- **Weighted Empirical Distribution**. *Given a sample $\boldsymbol{x} = (x_1, \dots, x_J) \in \mathbb{R}^J$ and J coefficients $\boldsymbol{w} = (w_1, \dots, w_J) \in \mathbb{R}^{+J}$, a random variable $X \in \mathbb{R}$ is said to have the weighted empirical distribution $\mathcal{E}(\cdot; \boldsymbol{x}, \boldsymbol{w})$ if the density is*

$$p(x; \boldsymbol{x}, \boldsymbol{w}) = \frac{1}{Z} \cdot \sum_{i=1}^{J} w_i \cdot \delta(x - x_i) \;, \qquad Z := \sum_{j=1}^{J} w_j$$

Normalization constant so that $1/Z \cdot \sum_{i=1}^{J} w_i = 1$

- **Cumulative Distribution Function**. *Let $X \sim \mathcal{E}(\cdot; \boldsymbol{x}, \boldsymbol{w})$ be distributed according to the weighted empirical distribution. Then the cumulative distribution is given by*

$$\int_{-\infty}^{t} \mathcal{E}(x; \boldsymbol{x}, \boldsymbol{w})\, \mathrm{d}x = \frac{1}{Z} \cdot \sum_{i=1}^{J} w_i \cdot \mathbb{I}(x_i \le t)$$

Counting all samples that are smaller than t weighing each by $1/Z \cdot w_i$

- **Moments**. *Let $X \sim \mathcal{E}(\cdot; \boldsymbol{x}, \boldsymbol{w})$ be distributed according to the weighted empirical distribution. Then we have for the k-th moment $E[X^k]$*

$$E[X^k] = \frac{1}{Z} \cdot \sum_{i=1}^{J} w_i \cdot x_i^k$$

Weighted sample averages

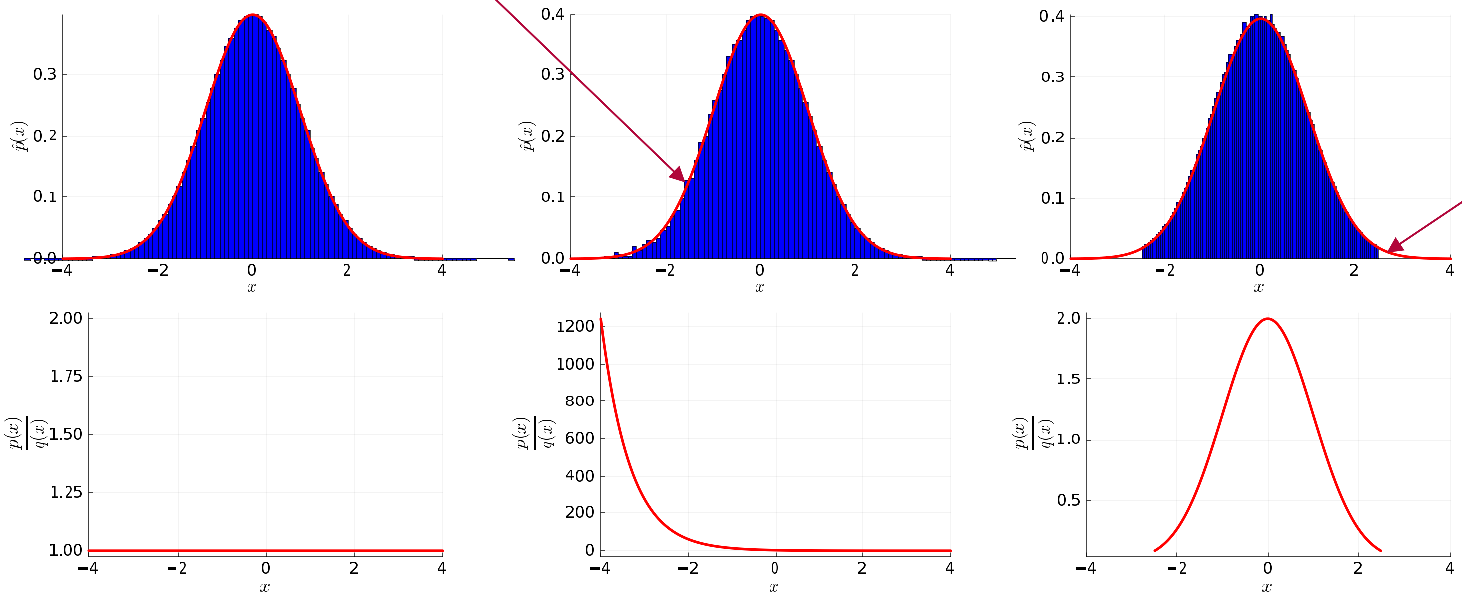
Importance Sampling in Pictures

```julia
function plot_importance_sampling(p, q; n=100000)
    xs = rand(q, n)
    ws = pdf(p, xs) ./ pdf(q, xs)
    p = histogram(
        xs,
        weights=ws,
        normalize = :pdf,
    )
    xlabel!(L"x")
    ylabel!(L"\hat{p}(x)")
    display(p)
end
```

Too little weight in the proposal

$$\boxed{p(x) = \mathcal{N}(x; 0,1)}$$

$q(x) = \mathcal{N}(x; 0,1)$ $\quad$ $q(x) = \mathcal{N}(x; 1.5,1)$ $\quad$ $q(x) = \mathcal{U}(x; -2.5,2.5)$



Zero weight in the proposal

Introduction to Probabilistic Machine Learning

Unit 4 – Graphical Models: Exact Inference

Marginalization by Sampling a Factor Graph

Importance Sampling

Given: Proposal distributions $q_1(\cdot), \dots, q_n(\cdot)$ for $X_1, \dots, X_n$

For $j \in \{1, \dots, J\}$

1. Sample $x_{j,1} \sim q_1$, $x_{j,2} \sim q_2$ up to $x_{j,n} \sim q_n$ into $\boldsymbol{x}_j = (x_{j,1}, \dots, x_{j,n})$
2. Compute

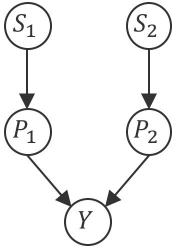
$$w_j = \prod_{i=1}^{m} f_i(x_{j,1}, \dots, x_{j,n}) \Bigg/ \prod_{k=1}^{n} q_k(x_{j,k})$$

Return: $\{\boldsymbol{x}_j\} \in \mathbb{R}^{J \times n}$ and $\boldsymbol{w} \in \mathbb{R}^{+J}$ as weighted empirical distribution

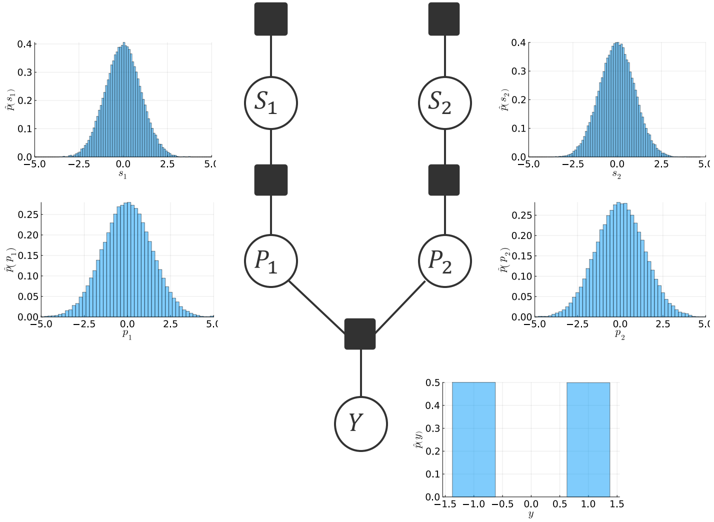
- **Pros**
  1. Sampling of the $\boldsymbol{x}_j$ is parallel rather than sequential (as in a Bayesian network)!
  2. The weights can also be computed in parallel!
- **Cons**
  1. If the proposal $\prod_{k=1}^{n} q_k(\cdot)$ is far from the marginal of $\prod_{i=1}^{m} f_i(\cdot, \dots, \cdot)$ then convergence is slow

Marginalization by Sampling a Factor Graph: Example

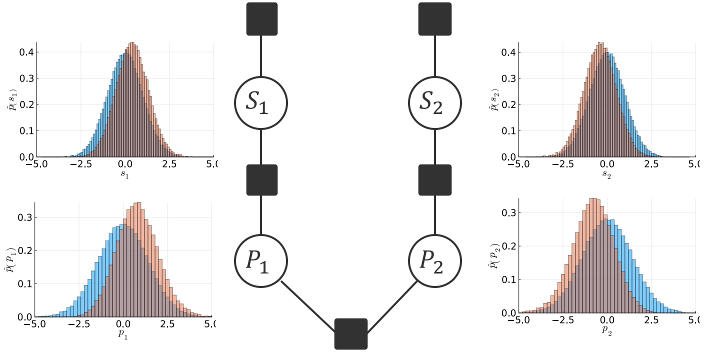
$$\mathcal{N}(s_1;\mu_1,\sigma_1^2)\cdot\mathcal{N}(s_2;\mu_2,\sigma_2^2)\cdot\mathcal{N}(p_1;s_1,\beta^2)\cdot\mathcal{N}(p_2;s_2,\beta^2)\cdot\mathbb{I}(y(p_1-p_2)>0)$$



Without match outcome



With match outcome ($y = 1$)



Introduction to Probabilistic Machine Learning

Unit 4 – Graphical Models: Exact Inference

Overview

1. Factor Graphs
2. Marginalization by Importance Sampling
3. **Marginalization by the Sum-Product Algorithm**
4. Practical Considerations in Message Passing

Marginalization using the Distributive Law

- **Observation 1**: The marginal of a factor graph is a sum (over all values of all hidden variables) of a product (of factor functions).

$$p(x_1) = \sum_{x_2=0}^{1} \sum_{x_3=0}^{1} \cdots \sum_{x_n=0}^{1} f_1(x_1, x_2, \dots, x_n) \cdot \dots \cdot f_m(x_1, x_2, \dots, x_n)$$

- **Observation 2**: Turning a sum of products *with a common factor* into a product of sums using the *distributive law* saves computation!

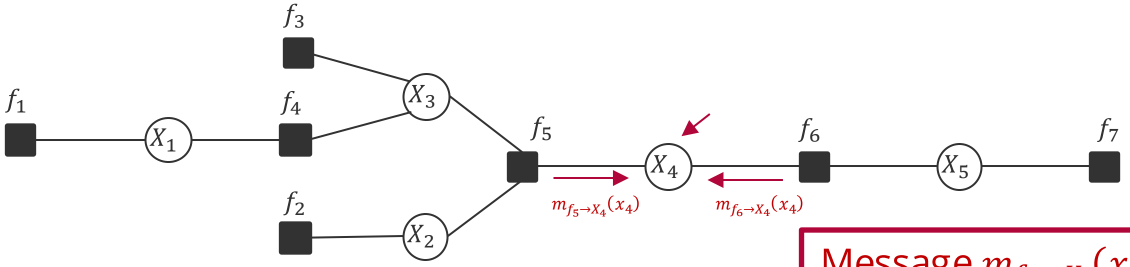
$$a \cdot b + a \cdot c = a \cdot (b + c)$$

3 operations

2 operations

- **Observation 3**: In a typical factor graph, functions only depend on a small number of variables.

Sum-Product Algorithm: Non-normalized Marginals



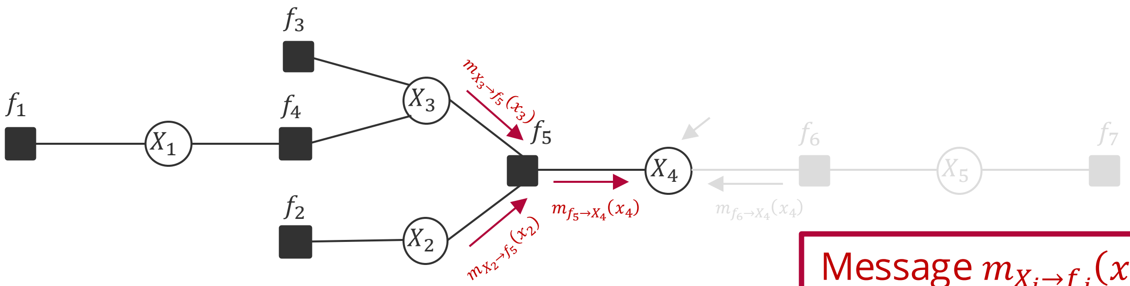
Message $m_{f_j \to X_i}(x_i)$ is the sum over all variables in the subtree rooted at f_j with $X_i = x_i$

$$p_{X_4}(x_4) = \sum_{\{x_1\}} \sum_{\{x_2\}} \sum_{\{x_3\}} \sum_{\{x_5\}} f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot f_4(x_1, x_3) \cdot f_5(x_2, x_3, x_4) \cdot f_6(x_4, x_5) \cdot f_7(x_5)$$

$$= \underbrace{\left[\sum_{\{x_1\}} \sum_{\{x_2\}} \sum_{\{x_3\}} f_1(x_1) \cdot f_2(x_2) \cdot f_3(x_3) \cdot f_4(x_1, x_3) \cdot f_5(x_2, x_3, x_4) \right]}_{m_{f_5 \to X_4}(x_4)} \cdot \underbrace{\left[\sum_{\{x_5\}} f_6(x_4, x_5) \cdot f_7(x_5) \right]}_{m_{f_6 \to X_4}(x_4)}$$

Non-normalized Marginals are the product of all incoming messages from neighbouring factors!

Sum-Product Algorithm: Message from Factor to Variable



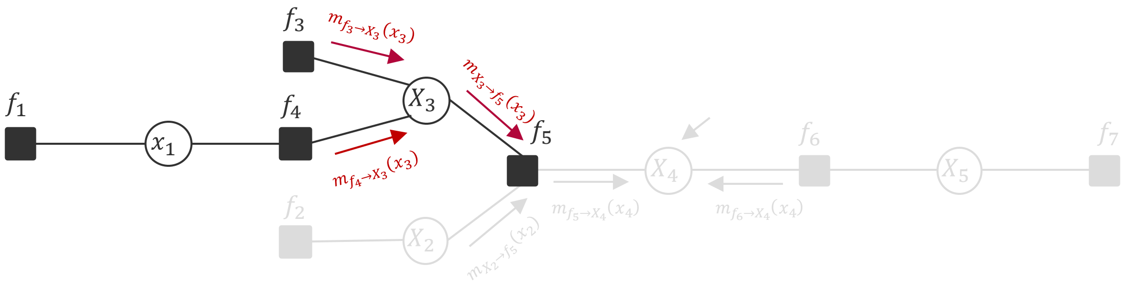
Message $m_{X_i\to f_j}(x_i)$ is the sum over all variables in the subtree rooted at X_i with $X_i = x_i$

$$m_{f_5\to X_4}(x_4) = \sum_{\{x_1\}}\sum_{\{x_2\}}\sum_{\{x_3\}} f_1(x_1)\cdot f_2(x_2)\cdot f_3(x_3)\cdot f_4(x_1,x_3)\cdot f_5(x_2,x_3,x_4)$$

$$= \sum_{\{x_2\}}\sum_{\{x_3\}} f_5(x_2,x_3,x_4)\cdot \underbrace{[f_2(x_2)]}_{m_{X_2\to f_5}(x_2)}\cdot \underbrace{\left[\sum_{\{x_1\}} f_1(x_1)\cdot f_3(x_3)\cdot f_4(x_1,x_3)\right]}_{m_{X_3\to f_5}(x_3)}$$

Messages from a factor to a variable sum out all neighboring variables weighted by their incoming message

Sum-Product Algorithm: Message from Variable to Factor



$$m_{X_3 \to f_5}(x_3) = \sum_{\{x_1\}} f_1(x_1) \cdot f_3(x_3) \cdot f_4(x_1, x_3)$$

$$= \underbrace{[f_3(x_3)]}_{m_{f_3 \to X_3}(x_3)} \cdot \underbrace{\left[\sum_{\{x_1\}} f_1(x_1) \cdot f_4(x_1, x_3)\right]}_{m_{f_4 \to X_3}(x_3)}$$

Messages from a variable to a factor multiply incoming messages from neighboring factors

Sum-Product Algorithm

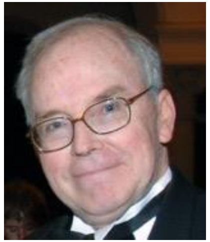
- **Sum-Product Algorithm (Aji-McEliece, 1997)**. Putting it all together, we have

$$p_{X_j}(x_j) = \prod_{i \in \mathrm{ne}(X_j)} m_{f_i \to X_j}(x_j)$$

$$m_{f_i \to X_j}(x_j) = \boxed{\sum_{\{\boldsymbol{x}_{\mathrm{ne}(f_i) \setminus \{j\}}\}}} f_i(\boldsymbol{x}_{\mathrm{ne}(f_i)}) \prod_{k \in \mathrm{ne}(f_i) \setminus \{j\}} m_{X_k \to f_i}(x_k)$$

$$m_{X_j \to f_k}(x_j) = \prod_{i \in \mathrm{ne}(X_j) \setminus \{k\}} m_{f_i \to X_j}(x_j)$$

- **Basis**: Generalized distributive law (which also holds for max-product)
- **Efficiency**: By storing messages, we
  - Only have to compute local summations in $\mathcal{O}(2^T)$ where degree $T = \max_f |\mathrm{ne}(f)|$!
  - All marginals can be computed recursively in $\mathcal{O}(E \cdot 2^T)$ vs $\mathcal{O}(2^n)$ (where E is the number of edges of the factor graph)!



Robert McEliece (1942 – 2019)

Overview

1. Factor Graphs
2. Marginalization by Importance Sampling
3. Marginalization by the Sum-Product Algorithm
4. **Practical Considerations in Message Passing**

Even more efficiency

- **Redundancies**. By the very definition of messages and non-normalized marginals

$$p_X(x) = \prod_{i \in \mathrm{ne}(X)} m_{f_i \to X}(x) = m_{f_k \to X}(x) \cdot \underbrace{\prod_{i \in \mathrm{ne}(X) \setminus \{k\}} m_{f_i \to X}(x)}_{m_{X \to f_k}(x)}$$

  - **Interpretation**. Application of Bayes' rule at a variable X at factor f

$$\underbrace{p_X(x)}_{\text{non-normalized posterior}} = \underbrace{m_{f \to X}(x)}_{\text{Likelihood}} \cdot \underbrace{m_{X \to f}(x)}_{\text{non-normalized prior}}$$

- **Storage Efficiency**. We only store the marginals $p_X(x)$ and $m_{f \to X}(x)$ because

$$m_{X \to f}(x) = \frac{p_X(x)}{m_{f \to X}(x)}$$

- **Exponential Family**. If all the messages from factors to variables are in the exponential family, then the marginals and messages from the variable to factors are simply additions and subtraction of natural parameters (up to normalization)!
  - **Example**: If $p_X(x) = \mathcal{G}(x; \tau_1, \rho_1)$ and $m_{f \to X}(x) = \mathcal{G}(x; \tau_2, \rho_2)$ then $m_{X \to f}(x) \propto \mathcal{G}(x; \tau_1 - \tau_2, \rho_1 - \rho_2)$

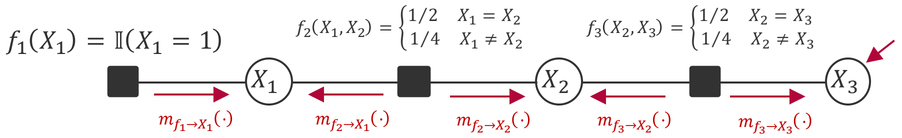
A Practical Implementation

1. Initialize all messages $m_{f\to X}(x)$ and marginals $p_X(x)$ with a constant function (i.e., uniform distribution)

2. Pick an arbitrary root (say, X_3)

3. Update all messages $m_{f\to X}(x)$ from the leaves of the tree rooted at X_3 **upwards**

4. Update all messages $m_{f\to X}(x)$ from the root X_3 to the leaves **downwards**



Update	$p_{X_1}(\cdot)$	$p_{X_2}(\cdot)$	$p_{X_3}(\cdot)$	$m_{f_1\to X_1}(\cdot)$	$m_{f_2\to X_1}(\cdot)$	$m_{f_2\to X_2}(\cdot)$	$m_{f_3\to X_2}(\cdot)$	$m_{f_3\to X_3}(\cdot)$
Initial	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$
$m_{f_1\to X_1}(\cdot)$	$[1,0,0]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$[1,0,0]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$
$m_{f_2\to X_2}(\cdot)$	$[1,0,0]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$[1,0,0]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$
$m_{f_3\to X_3}(\cdot)$	$[1,0,0]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{3}{8},\frac{5}{16},\frac{5}{16}\right]$	$[1,0,0]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{3}{8},\frac{5}{16},\frac{5}{16}\right]$
$m_{f_3\to X_2}(\cdot)$	$[1,0,0]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{3}{8},\frac{5}{16},\frac{5}{16}\right]$	$[1,0,0]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{3}{8},\frac{5}{16},\frac{5}{16}\right]$
$m_{f_2\to X_1}(\cdot)$	$[1,0,0]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{3}{8},\frac{5}{16},\frac{5}{16}\right]$	$[1,0,0]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{1}{2},\frac{1}{4},\frac{1}{4}\right]$	$\left[\frac{1}{3},\frac{1}{3},\frac{1}{3}\right]$	$\left[\frac{3}{8},\frac{5}{16},\frac{5}{16}\right]$

$$m_{f_2\to X_2}(x_2) = \sum_{x_1=1}^{3} f_2(x_1, x_2) \cdot \boxed{\frac{p_{X_1}(x_1)}{m_{f_2\to X_1}(x_1)}} \longleftarrow m_{X_1\to f_2}(x_1)$$

Summary

1. **Factor Graphs**
   - Generalization of Bayesian networks specifically designed for fast and easy inference
   - Date back to coding algorithms
2. **Marginalization by Importance Sampling**
   - There exists an easy-to-implement sampling algorithm for the marginal distributions
   - Sampling efficiency depends on the “match” of the proposal distributions with the marginal distributions
3. **Marginalization by the Sum-Product Algorithm**
   - Application of generalized distributive law
   - Trades memory (“messages”) for computation (“sums”)
   - Reduces the computational complexity to exponential in the largest out-degree of a factor rather than exponential in the number of variables!
   - If messages can be computed efficiently, there is no faster, exact algorithm for marginalization!

See you next week!