Mathe III

Bayesianische Statistik

Ralf Herbrich

Hasso Plattner Institut
Digital Engineering · Universität Potsdam

Overview

1. Bayesian Probabilities and Conjugacy
2. Graphical Models
3. The Sum-Product Algorithm
4. Bayesian Ranking: TrueSkill
5. Information Theory
6. Arithmetic Coding

Overview

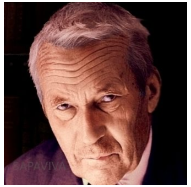
1. **Bayesian Probabilities and Conjugacy**
2. Graphical Models
3. The Sum-Product Algorithm
4. Bayesian Ranking: TrueSkill
5. Information Theory
6. Arithmetic Coding

Frequentist vs. Subjective Probabilities

- **Kolmogorov (1933)**: *The rules of probability for **sets** follow from the following 3 axioms*
    1. $P(A) \geq 0$ *for all* $A \subseteq S$
    2. $P(S) = 1$
    3. $P(\bigcup_i A_i) = \sum_i P(A_i)$ *if for all* $i \neq j\colon A_i \cap A_j = \emptyset$

- **Cox (1944)**: *The rules of probability for **logic** follow from the following 3 axioms*
    1. $P(A) \in [0,1]$ *for all logical statements* A
    2. $P(A)$ *is independent of how the statement is represented*
    3. *If* $P(A|C') > P(A|C)$ *and* $P(B|A \wedge C') = P(B|A \wedge C)$ *then*

$$P(A \wedge B|C') \geq P(A \wedge B|C)$$



Andrey Kolmogorov (1903 – 1987)

Richard Threlkeld Cox (1898 – 1991)

Frequentist vs. Subjectivist Interpretation

- **Frequentist Interpretation**
  - Probability is a property of the event (“it rains tomorrow in Bangalore”)
  - Is operationalized by repeated experiments
  - Typically used by scientists and engineers

- **Subjective Interpretation**
  - Probability is an expression of belief of the person makes a statement
  - Is subjective and people-dependent: Two people with identical data can come to different probabilities
  - Typically used by philosophers and economists

1. Probability is not a physical measure but a thought model for randomness!
2. The mathematical rules for probability are **identical** for both interpretations!

Probability Distributions: Conjugacy

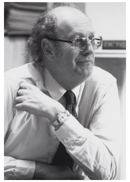
- **Bayes Rule for Random Variables**. *For any probability distribution p over two random variables X and Θ, it holds*

$$\underbrace{p(\theta|x)}_{\text{Posterior}} = \frac{\overbrace{\boxed{\overbrace{p(x|\theta)}^{\text{Likelihood}} \cdot \underbrace{p(\theta)}_{\text{Prior}}}}^{p(x,\theta)}}{p(x)}$$

- **Conjugacy**. *A family $\{p(x,\theta)\}_{x,\theta}$ is conjugate if the posterior $p(\theta|x)$ is part of the same family as the prior $p(\theta)$ for any value of x.*

Likelihood $p(x\|\theta)$	Model Parameter	Conjugate Prior $p(\theta)$
$\mathrm{Ber}(x;\pi)$	π	$\mathrm{Beta}(\pi;\alpha,\beta)$
$\mathrm{Bin}(x;n,\pi)$	π	$\mathrm{Beta}(\pi;\alpha,\beta)$
$\mathcal{N}(x;\mu,\sigma^2)$	μ,σ^2	$\mathcal{N}(\mu;m,s^2)$

- **Big Advantage**: Computing the exact posterior is computationally efficient!



Howard Raiffa
(1924 – 2016)

Robert Osher Schlaifer
(1914 – 1994)

Probability Distributions: Normal

- **Normal Distribution**. *A continuous random variable X is said to have a normal distribution if the density is given by*

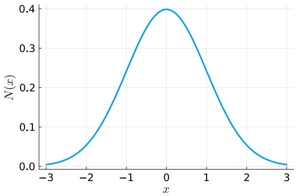
$$p_X(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- **Properties**:

$$E[X] = \mu$$
$$\mathrm{var}[X] = \sigma^2$$

- **Importance**. The Normal distribution plays a fundamental role in ML!
  - **Data Modelling**: The limit distribution for the sum of a large number of indepedent and identically distributed random variables.
  - **Machine Learning**: The most common prior distribution for the parameters of prediction functions!
  - **Information Theory**: The distribution function with the most uncertainty ("entropy") when fixing mean and variance of the random variable.

Carl Friedrich Gauss
(1777 - 1855)



Mathe III

Unit 13a –
Bayesianische Statistik

Normal Distribution: Representations

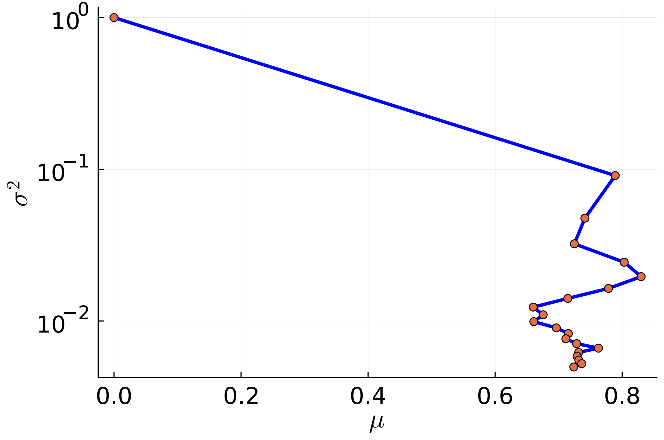
- **Scale-Location Parameters**

$$\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\cdot\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

- **Conversions**

$$\mathcal{N}(x;\mu,\sigma^2) = \mathcal{G}\left(x;\frac{\mu}{\sigma^2},\frac{1}{\sigma^2}\right)$$

- **Posterior Inference**



Two divisions only!

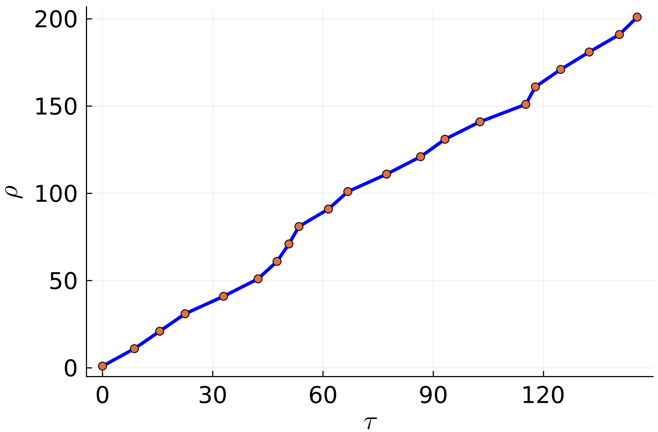
- **Natural Parameters**

$$\mathcal{G}(x;\tau,\rho) = \sqrt{\frac{\rho}{2\pi}}\cdot\exp\left(-\frac{\tau^2}{2\rho}\right)\cdot\exp\left(\tau\cdot x-\rho\cdot\frac{x^2}{2}\right)$$

- **Conversions**

$$\mathcal{G}(x;\tau,\rho) = \mathcal{N}\left(x;\frac{\tau}{\rho},\frac{1}{\rho}\right)$$

- **Posterior Inference**



Normal Distributions: Efficient Products & Divisions

- **Theorem (Multiplication).** *Given two one-dimensional Gaussian distributions* $\mathcal{G}(x;\tau_1,\rho_1)$ *and* $\mathcal{G}(x;\tau_2,\rho_2)$ *we have*

Gaussian density

$$\mathcal{G}(x;\tau_1,\rho_1)\cdot\mathcal{G}(x;\tau_2,\rho_2)=\mathcal{G}(x;\tau_1+\tau_2,\rho_1+\rho_2)\cdot\mathcal{N}(\mu_1;\mu_2,\sigma_1^2+\sigma_2^2)$$

Additive updates!

- **Theorem (Division).** *Given two one-dimensional Gaussian distributions* $\mathcal{G}(x;\tau_1,\rho_1)$ *and* $\mathcal{G}(x;\tau_2,\rho_2)$ *where* $\rho_1 \geq \rho_2$ *we have*

Correction factor

$$\frac{\mathcal{G}(x;\tau_1,\rho_1)}{\mathcal{G}(x;\tau_2,\rho_2)}=\frac{\mathcal{G}(x;\tau_1-\tau_2,\rho_1-\rho_2)}{\mathcal{N}(\mu_1;\mu_2,\sigma_2^2-\sigma_1^2)}\cdot\frac{\sigma_2^2}{\sigma_2^2-\sigma_1^2}$$

Subtractive updates!

Gaussian density

Limit Normal Distributions: Dirac Delta and Uniform

- **Dirac Delta**. The Dirac delta function $\delta(\cdot)$ is defined as the limit $\sigma^2 \to 0$

$$\delta(x) = \lim_{\sigma^2 \to 0} \mathcal{N}(x; 0, \sigma^2)$$

- **Gaussian Uniform**. The Gaussian uniform $\mathcal{U}(\cdot)$ is defined as the limit $\sigma^2 \to \infty$

$$\mathcal{U}(x) = \lim_{\sigma^2 \to +\infty} \mathcal{N}(x; 0, \sigma^2)$$

- **Theorem (Convolution of Normal with Dirac)**. *For any $\mu \in \mathbb{R}$ and $\sigma^2 \in \mathbb{R}^+$*

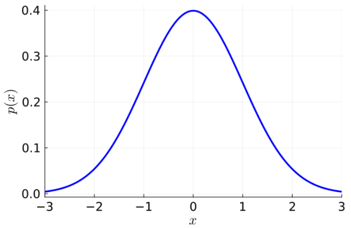
$$\int_{-\infty}^{+\infty} \delta(x) \cdot \mathcal{N}(x; \mu, \sigma^2)\, \mathrm{d}x = \boxed{\mathcal{N}(0; \mu, \sigma^2)}$$

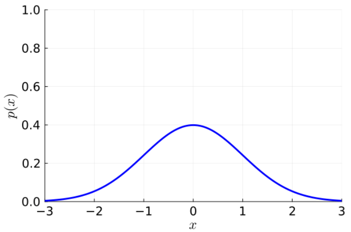
Gaussian density at $x = 0$

- **Theorem (Product of Normal with Uniform)**. *For any $\mu \in \mathbb{R}$ and $\sigma^2 \in \mathbb{R}^+$*

$$\frac{\mathcal{U}(x) \cdot \mathcal{N}(x; \mu, \sigma^2)}{\int_{-\infty}^{+\infty} \mathcal{U}(\tilde{x}) \cdot \mathcal{N}(\tilde{x}; \mu, \sigma^2)\, \mathrm{d}\tilde{x}} = \boxed{\mathcal{N}(x; \mu, \sigma^2)}$$

Equivalent to multiplying with 1





Overview

1. Bayesian Probabilities and Conjugacy
2. **Graphical Models**
3. The Sum-Product Algorithm
4. Bayesian Ranking: TrueSkill
5. Information Theory
6. Arithmetic Coding

Graphical Models

- **Challenge**: How to formulate complex likelihoods/data models & priors for *actual* data?
  - **Example 1**: Match outcomes $y \in \{-1,1\}$ (data) for a head-to-head match between two players
    - **Prior**: $p(\boldsymbol{s}) = \mathcal{N}(s_1; \mu_1, \sigma_1^2) \cdot \mathcal{N}(s_2; \mu_2, \sigma_2^2)$ ← skill belief
    - **Likelihood**: $p(y|\boldsymbol{s}) = \int \mathcal{N}(p_1; s_1, \beta^2) \cdot \mathcal{N}(p_2; s_2, \beta^2) \cdot \mathbb{I}(y(p_1 - p_2) > 0) \; \mathrm{d}p_1 \, \mathrm{d}p_2$ ← marginalization

      Match outcome

      Player performance
  - **Example 2**: Time series $\boldsymbol{y}$ of temperatures
    - **Prior**: $p(w) = \mathcal{N}(w; \mu, \sigma^2)$ ← External state mapping parameter belief
    - **Likelihood**: $p(\boldsymbol{y}|w, X) = \int \mathcal{N}(z_1;\, w \cdot x_1, \tau^2) \cdot \mathcal{N}(y_1; z_1, \beta^2) \cdot \mathcal{N}(z_2; z_1 + w \cdot x_2, \tau^2) \cdots \, \mathrm{d}\boldsymbol{z}$

      marginalization

      Dynamics model

      Observed temperature model

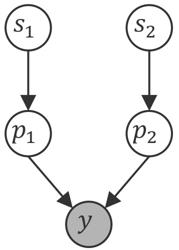
      Conditional hidden state model

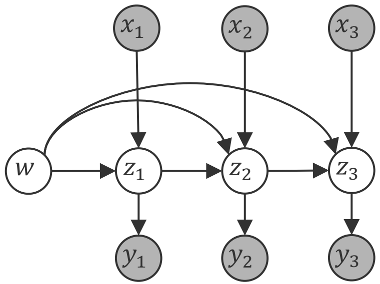
Graphical Models

- **Observation**: The product structure of the probabilities seems crucial
- **Idea**: Define a graph where each of the variables are nodes and edges indicate factor relationships between variables

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

$$\mathcal{N}(w;\mu,\sigma^2)\cdot\mathcal{N}(z_1;\ w\cdot x_1,\tau^2)\cdot\mathcal{N}(y_1;z_1,\beta^2)\cdot\mathcal{N}(z_2;z_1+w\cdot x_2,\tau^2)\cdot\mathcal{N}(y_2;z_2,\beta^2)\cdots$$



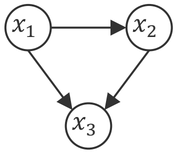


- **Advantages**: Simple way to visualize factor structure of the joint probability
  - **Bayesian Networks**: Insights into (conditional) independence based on graph properties
  - **Factor Graphs**: Insights into efficient inference and approximation algorithms

Bayesian Networks

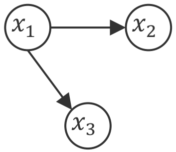
- **Observation**. Any joint distribution $p(x_1, \dots, x_n)$ can be written as
$$p(x_1, \dots, x_n) = \prod_{i=1}^{n} p(x_i | x_1, \dots, x_{i-1})$$
- **Bayesian Network**. *Given a joint distribution as a product of conditional distributions,* $p(x_1, \dots, x_n) = \prod_{i=1}^{n} p(x_i | \mathrm{parents}_i)$*, a* Bayesian network *is a graph with a node for every variable* x_i*, and a directed edge from every variable* $x \in \mathrm{parent}_i$ to x_i*. If the variable is independent of all other variables, it has no incoming edges.*

- **Examples**: For 3 variables, we have these four generic Bayesian networks



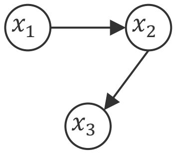
$$p(x_1, x_2, x_3) = p(x_1) \cdot p(x_2|x_1) \cdot p(x_3|x_1, x_2)$$

full mesh



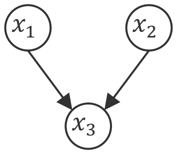
$$p(x_1, x_2, x_3) = p(x_1) \cdot \prod_{i=2}^{3} p(x_i|x_1)$$

star



$$p(x_1, x_2, x_3) = p(x_1) \cdot \prod_{i=2}^{3} p(x_i|x_{i-1})$$

chain



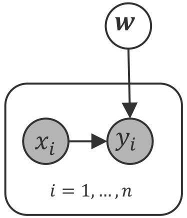
$$p(x_1, x_2, x_3) = p(x_3|x_1, x_2) \cdot \prod_{i=1}^{2} p(x_i)$$

sink

Bayesian Network Models

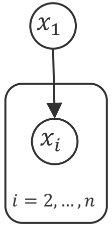
- **Plate**. If a subset of variables has the same relation only differing in their index, we use a "plate" to collapse them into a single graphical element.
  - Increase readability of models for large amounts of parameters and data
- A Bayesian network must always be a **directed acyclic graph** because only those have a topological order corresponding to a variable order.
- **Observed Variables**. If a subset of variables has been observed ("data"), the variable nodes are usually shaded ("clamped").
  - **Example**: Discriminatory Models

$$p\big(\boldsymbol{w}, (x_1, y_1), \dots, (x_n, y_n)\big) = \prod_{i=1}^{n} p(y_i | x_i, \boldsymbol{w}) \cdot p(\boldsymbol{w})$$



$$p\big(\boldsymbol{w} | (x_1, y_1), \dots, (x_n, y_n)\big) \propto \prod_{i=1}^{n} p(y_i | x_i, \boldsymbol{w}) \cdot p(\boldsymbol{w})$$

$$p(x_1, x_2, \dots, x_n) = p(x_1) \cdot \prod_{i=2}^{n} p(x_i | x_1)$$



Sampling a Bayesian Network

- One advantage of a Bayesian network is the ability to *sample* $p(x_1, \dots, x_n)$

Ancestral Sampling

1. Topologically sort all variables $x_1, \dots, x_n$ into $x_{(1)}, \dots, x_{(n)}$
2. Sample each variable $x_{(i)}$ using distribution $p\left(x_{(i)} | x_{(1)}, \dots, x_{(i-1)}\right)$

- **Assumption**
  1. Sampling from the conditional distributions is simpler than from the joint distribution
  2. There are no clamped nodes, that is, we do not condition on any variable
- **Problems**
  1. Sampling is *sequential* one variable at the time
  2. Conditioning happens only on frequent events because for samples $x_{j,1}, \dots, x_{j,n}$

$$\frac{\left|\left\{\left(x_{j,1}, \dots, x_{j,n}\right) \mid x_{j,1} = x_1 \wedge \dots \wedge x_{j,n} = x_n\right\}\right|}{\left|\left\{x_{j,n} = x_n\right\}\right|} \approx \frac{p(x_1, \dots, x_n)}{p(x_n)} = p(x_1, \dots, x_{n-1} | x_n)$$

≈ 0 for rare events

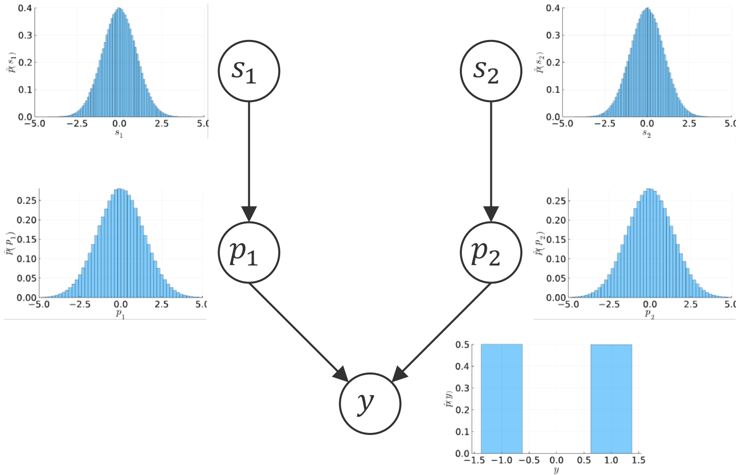
Sampling a Bayesian Network: Example

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

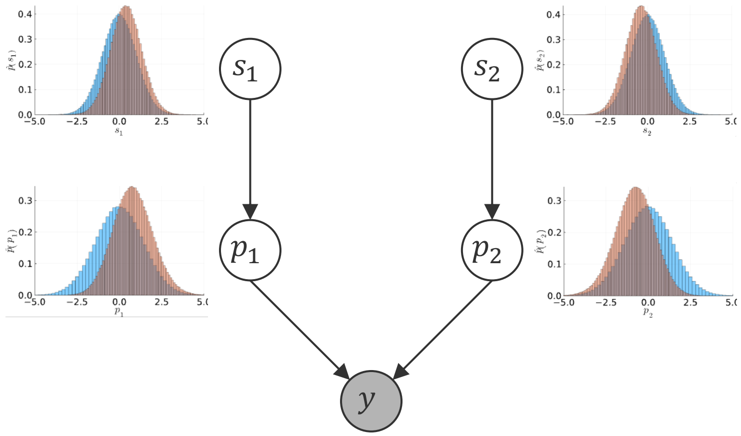
$\mu_1 = \mu_2$

```julia
# samples from the TrueSkill graphical model
function sample(; n = 100000, μ1=0.0, σ1=1.0, μ2=0.0, σ2=1.0, β=1.0)
    samples = Vector{Vector{Float64}}(undef, n)
    for i in 1:n
        s1 = rand(Normal(μ1, σ1))
        s2 = rand(Normal(μ2, σ2))
        p1 = rand(Normal(s1, β))
        p2 = rand(Normal(s2, β))
        y = p1 > p2 ? 1.0 : -1.0
        samples[i] = [s1, s2, p1, p2, y]
    end
    return samples
end
```

Without match outcome



With match outcome ($y = 1$)

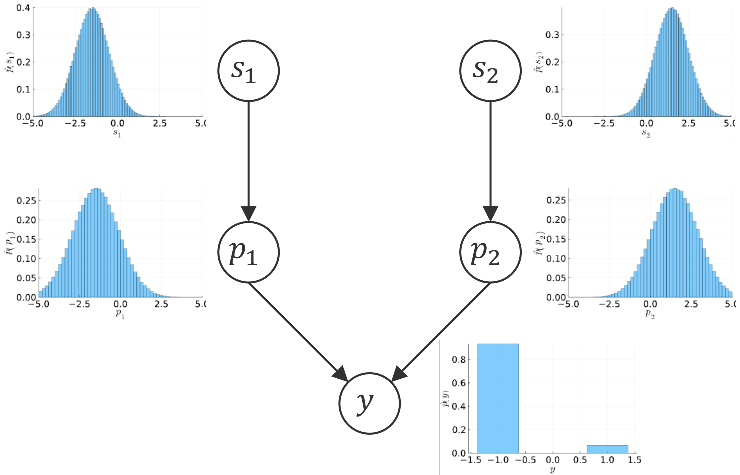


Sampling a Bayesian Network: Example (ctd)

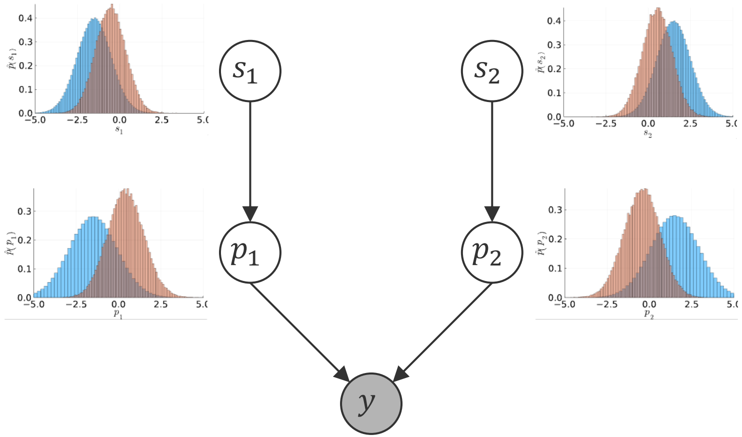
$$\boxed{\mathcal{N}(s_1;\mu_1,\sigma_1^2)\cdot\mathcal{N}(s_2;\mu_2,\sigma_2^2)\cdot\mathcal{N}(p_1;s_1,\beta^2)\cdot\mathcal{N}(p_2;s_2,\beta^2)\cdot\mathbb{I}(y(p_1-p_2)>0)}$$

$\mu_1 \ll \mu_2$

Without match outcome



With match outcome ($y=1$)



Overview

1. Bayesian Probabilities and Conjugacy
2. Graphical Models
3. **The Sum-Product Algorithm**
4. Bayesian Ranking: TrueSkill
5. Information Theory
6. Arithmetic Coding

Inference in Probabilistic Models

- **Inference**: In order to learn from data we follow a three-step procedure
  1. **Modelling**: Formulate a joint model $p(\boldsymbol{\theta}, D)$ of parameters $\boldsymbol{\theta} = \theta_1, \ldots, \theta_n$ 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 $y \in \{0,1\}$

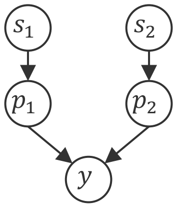
$$p(s_1, s_2, p_1, p_2, y) = \underbrace{\mathcal{N}(s_1; \mu_1, \sigma_1^2) \cdot \mathcal{N}(s_2; \mu_2, \sigma_2^2)}_{p(s_1, s_2)} \cdot \mathcal{N}(p_1; s_1, \beta^2) \cdot \mathcal{N}(p_2; s_2, \beta^2) \cdot \mathbb{I}(y(p_1 - p_2) > 0)$$

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

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

  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 $x_1, \ldots, x_n$

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

- **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 $p(x_1, x_2, \ldots, x_n) = \prod_i p(x_i)$: then there are only $O(n)$ sums and $n-1$ sum to one!

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).

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

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

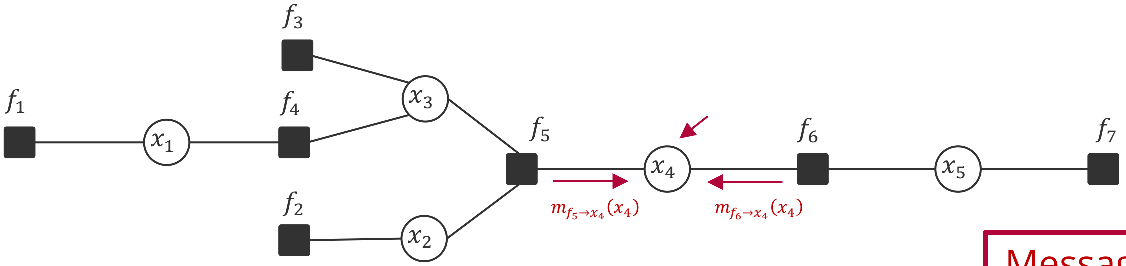
$$a \cdot b + a \cdot c = a \cdot (b + c)$$

3 operations

2 operations

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

Sum-Product Algorithm: Marginals



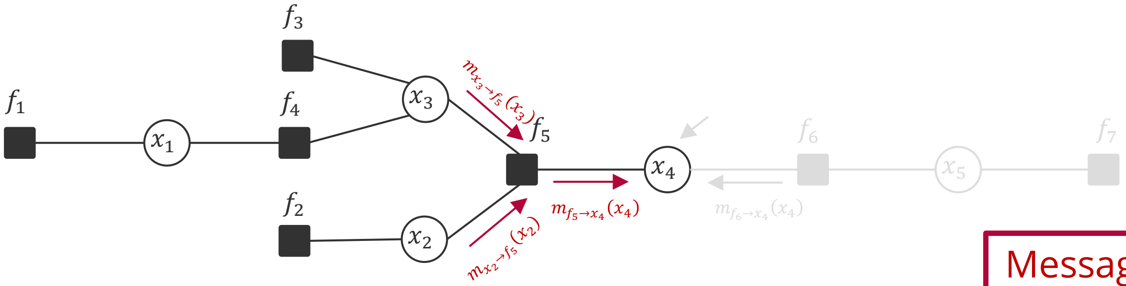
Message $m_{f_j \to x_i}(x_i)$ is the sum over all variables in the subtree rooted at f_j

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

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

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

Sum-Product Algorithm: Message from Factor to Variable



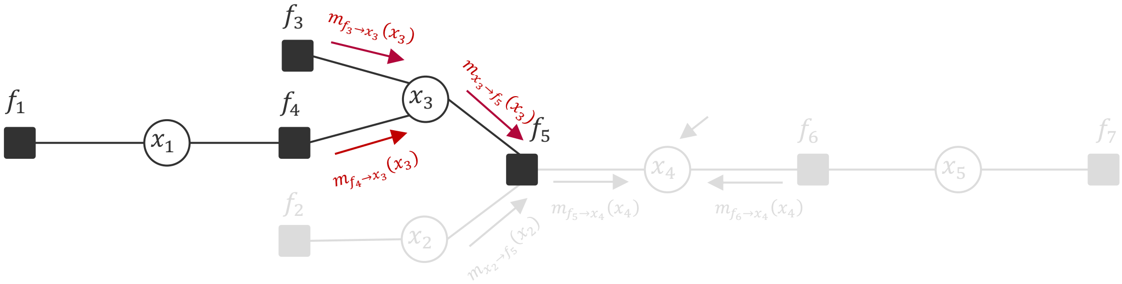
Message $m_{x_i \to f_j}(x_i)$ is the sum over all variables in the subtree rooted at x_i

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

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

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 message from neighboring factors

Sum-Product Algorithm

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

$$p(x) = \prod_{f \in \mathrm{ne}(x)} m_{f \to x}(x)$$

$$m_{f \to x}(x) = \sum_{\{x' \in \mathrm{ne}(f) \setminus \{x\}\}} \cdots \sum_{\{x'' \in \mathrm{ne}(f) \setminus \{x\}\}} f(x, x', \dots, x'') \prod_{x' \in \mathrm{ne}(f) \setminus \{x\}} m_{x' \to f}(x')$$

$$m_{x \to f}(x) = \prod_{f' \in \mathrm{ne}(x) \setminus \{f\}} m_{f' \to x}(x)$$

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

Robert McEliece (1942 – 2019)

Even more efficiency

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

$$p(x) = \prod_{f \in \mathrm{ne}(x)} m_{f \to x}(x) = m_{f' \to x}(x) \cdot \boxed{\prod_{f \in \mathrm{ne}(x) \setminus \{f'\}} m_{f \to x}(x)} \longleftarrow m_{x \to f'}(x)$$

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

$$\underbrace{p(x)}_{\text{posterior}} = \underbrace{m_{f \to x}(x)}_{\text{Likelihood} \times \text{normalization}} \cdot \underbrace{m_{x \to f}(x)}_{\text{prior}}$$

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

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

- **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!
  - **Example**: If $p(x) = \mathcal{G}(x; \tau_1, \rho_1)$ and $m_{f \to x}(x) = \mathcal{G}(x; \tau_2, \rho_2)$ then $m_{x \to f}(x) \propto \mathcal{G}(x; \tau_1 - \tau_2, \rho_1 - \rho_2)$

Sum-Product Algorithm Revisited

- The key operation for factor $f(x_1, x_2, \ldots, x_n)$ and variable x_1 is

$$m_{f\to x_1}(x_1) = \sum_{\{x_2\}} \cdots \sum_{\{x_n\}} f(x_1, x_2, \ldots, x_n) \prod_{j=2}^{n} m_{x_j\to f}(x_j)$$

If all $m_{x_j\to f}(x_j)$ are Gaussian, the result **might not be** Gaussian!

- Based on outgoing messages, we can compute both marginals $p(x)$ and $m_{x\to f}(x)$

$$p(x) = \prod_{f\in\mathrm{ne}(x)} m_{f\to x}(x) \qquad m_{x\to f}(x) = \frac{p(x)}{m_{f\to x}(x)}$$

If all $m_{x_j\to f}(x_j)$ are Gaussian, the result **must be** Gaussian!

- **Idea**:
  1. We approximate all outgoing messages $m_{f\to x}(\cdot)$ by a Gaussian $\hat{m}_{f\to x}(\cdot) = \mathcal{N}(\cdot; \mu, \sigma^2)$
  2. We measure the approximation quality in the marginal, **not** the outgoing message

$$\hat{p}(\cdot) = \arg \min_{\mathcal{N}(\cdot;\mu,\sigma^2)} \mathrm{KL}\left[\frac{m_{f\to x}(\cdot)\cdot \hat{m}_{x\to f}(\cdot)}{\int_{-\infty}^{+\infty} m_{f\to x}(\tilde{x})\cdot \hat{m}_{x\to f}(\tilde{x})\, \mathrm{d}\tilde{x}}, \frac{\mathcal{N}(\cdot;\mu,\sigma^2)\cdot \hat{m}_{x\to f}(\cdot)}{\int_{-\infty}^{+\infty} \mathcal{N}(\tilde{x};\mu,\sigma^2)\cdot \hat{m}_{x\to f}(\tilde{x})\, \mathrm{d}\tilde{x}}\right]$$

True marginal with approximate incoming message

Approximate marginal with approximate incoming message

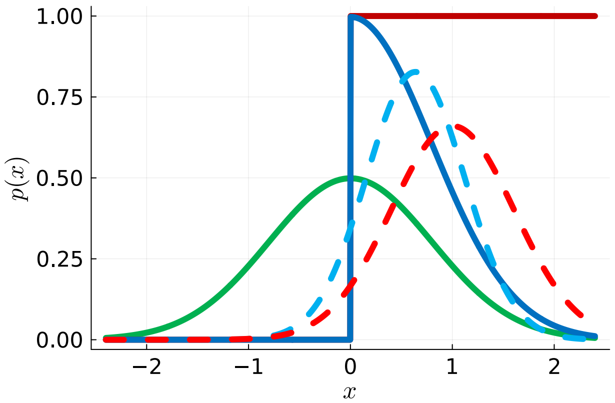
Approximate Message Passing: Example

$$\boxed{f(x) = \mathbb{I}(x > 0)}$$

$$\hat{m}_{x\to f}(x) \propto \frac{\hat{p}(x)}{\hat{m}_{f\to x}(x)} \longrightarrow p(x) \propto f(x) \cdot \hat{m}_{x\to f}(x)$$

$$\hat{p}(x) = \mathcal{N}\left(x; E_{X\sim p(x)}[X], \mathrm{var}_{X\sim p(x)}[X]\right)$$

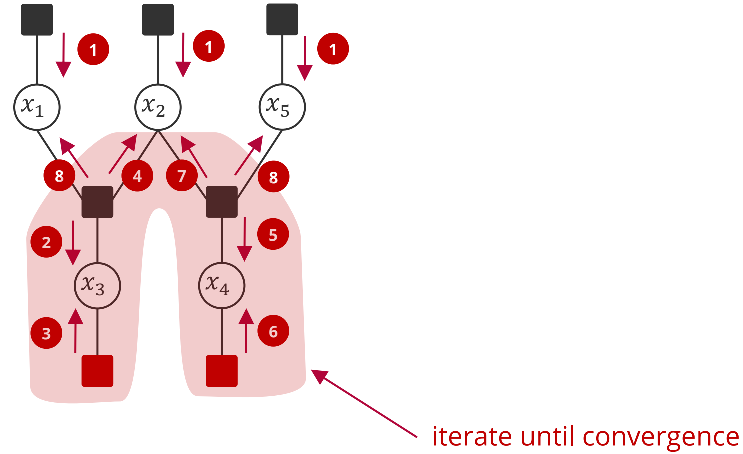
$$\hat{m}_{f\to x}(x) \propto \frac{\hat{p}(x)}{\hat{m}_{x\to f}(x)}$$



Expectation Propagation

- **Idea**: If we have factors in the factor graph that require approximate messages, we keep iterating on the whole path between them until convergence minimizing $\mathrm{KL}\left(p(\cdot)|\mathcal{N}(\cdot;\mu,\sigma^2)\right)$ locally for the affected marginals of the approximate factor.
- **Theorem (Minka, 2003)**: Approximate message passing will converge if the approximating distribution is in the exponential family!

Tom Minka



Viel Spaß bis zur nächsten Vorlesung!