

Online Algorithms for Sum-Product Networks with Continuous Variables

Priyank Jaini
Ph.D. Seminar



Mixture Models

Consistent/Robust

Tensor Decomposition
And
Spectral Learning

Offline

Bayesian
Learning

ADF, EP, SGD, oEM

Non convergence and
local optima

Online

Can be distributed;
Practical problems

Bayesian Moment Matching
algorithm

SPNs

Bayesian Moment
Matching for continuous
SPNs

Comparison with other
deep networks

Streaming Data

Activity Recognition



Recommendation



Streaming Data

Activity Recognition



Recommendation



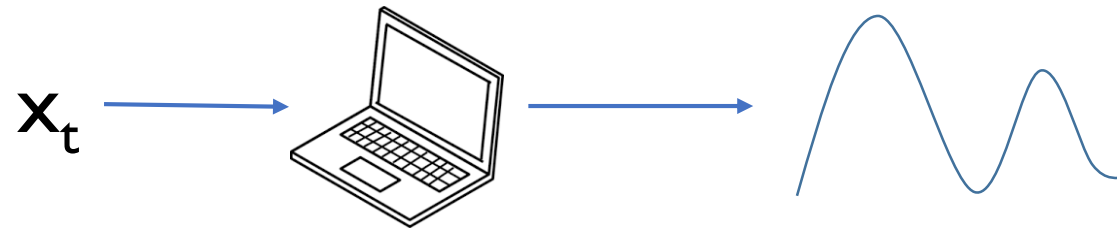
Challenge : **update model after each observation**

Algorithm's characteristics

Algorithm's characteristics

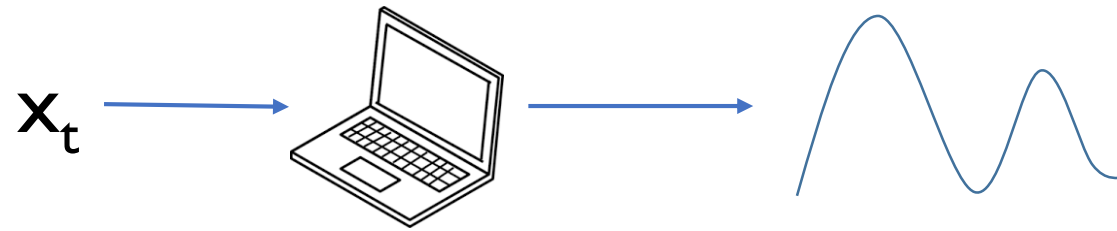
Algorithm's characteristics

Online

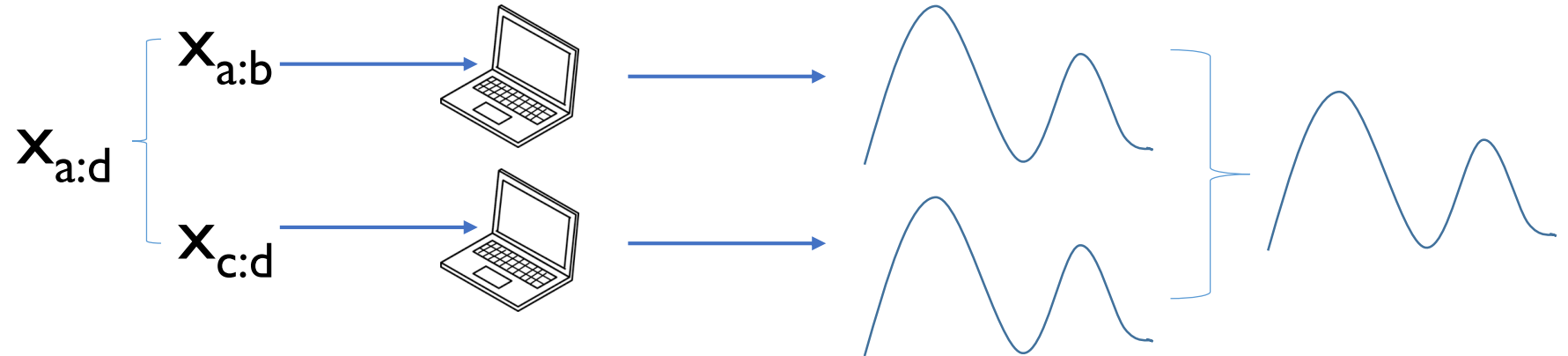


Algorithm's characteristics

Online

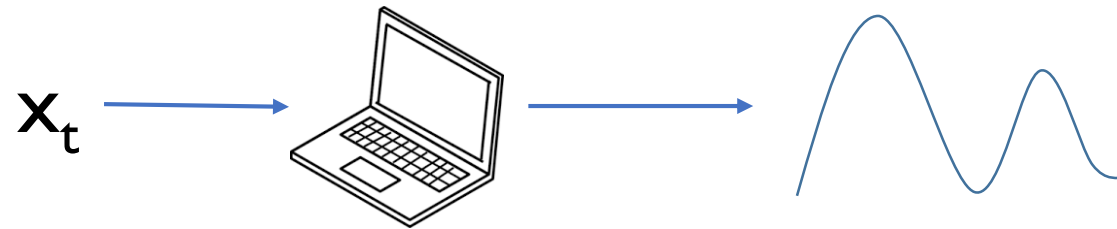


Distributed

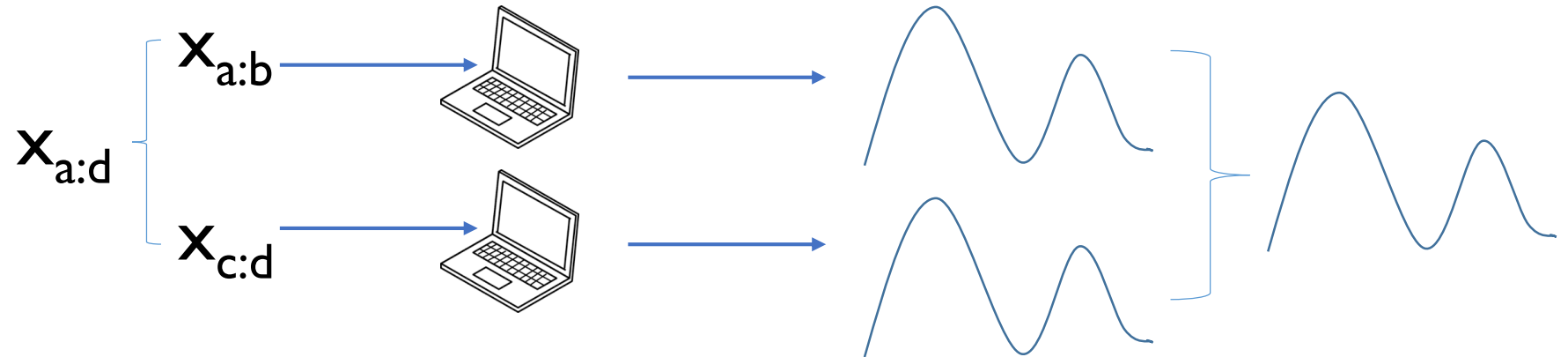


Algorithm's characteristics

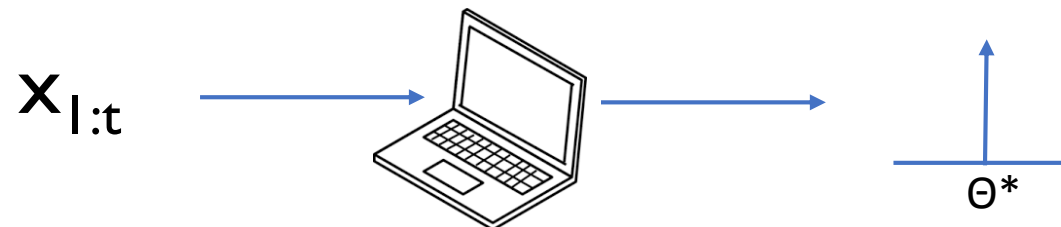
Online



Distributed



Consistent



How can we learn mixture models robustly from streaming data ?

Learning Algorithms

Learning Algorithms

- **Robust** : Tensor Decomposition (Anandkumar et.al, 2014), Spectral Learning (Hsu et al, 2012, Parikar and Xing, 2011);
offline

Learning Algorithms

- **Robust** : Tensor Decomposition (Anandkumar et.al, 2014), Spectral Learning (Hsu et al, 2012, Parikar and Xing, 2011); **offline**
- **Online** :
 - Assumed Density Filtering (Maybeck 1982; Lauritzen 1992; Oppen & Winther 1999); **not robust**

Learning Algorithms

- **Robust** : Tensor Decomposition (Anandkumar et.al, 2014), Spectral Learning (Hsu et al, 2012, Parikar and Xing, 2011); **offline**
- **Online** :
 - Assumed Density Filtering (Maybeck 1982; Lauritzen 1992; Oppen & Winther 1999); **not robust**
 - Expectation Propagation (Minka 2001); **does not converge**

Learning Algorithms

- **Robust** : Tensor Decomposition (Anandkumar et.al, 2014), Spectral Learning (Hsu et al, 2012, Parikar and Xing, 2011);
offline
- **Online** :
 - Assumed Density Filtering (Maybeck 1982; Lauritzen 1992; Oppor & Winther 1999); **not robust**
 - Expectation Propagation (Minka 2001); **does not converge**
 - Stochastic Gradient Descent (Zhang 2004)
 - online Expectation Maximization (Cappe 2012)**SGD and oEM : local optimum and cannot be distributed**

Learning Algorithms

- **Exact Bayesian Learning** : Dirichlet Mixtures(Ghosal et al 1999), Gaussian Mixtures(Lijoi et al, 2005), Non-parametric Problems (Barron et al, 1999), (Freedman, 1999)

Learning Algorithms

- **Exact Bayesian Learning** : Dirichlet Mixtures(Ghosal et al 1999), Gaussian Mixtures(Lijoi et al, 2005), Non-parametric Problems (Barron et al, 1999), (Freedman, 1999)

Exact Bayesian Learning  Online

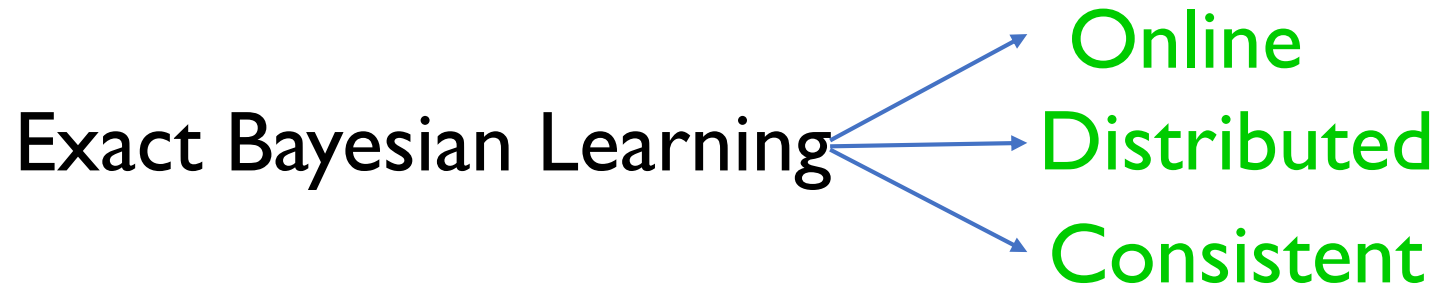
Learning Algorithms

- **Exact Bayesian Learning** : Dirichlet Mixtures(Ghosal et al 1999), Gaussian Mixtures(Lijoi et al, 2005), Non-parametric Problems (Barron et al, 1999), (Freedman, 1999)



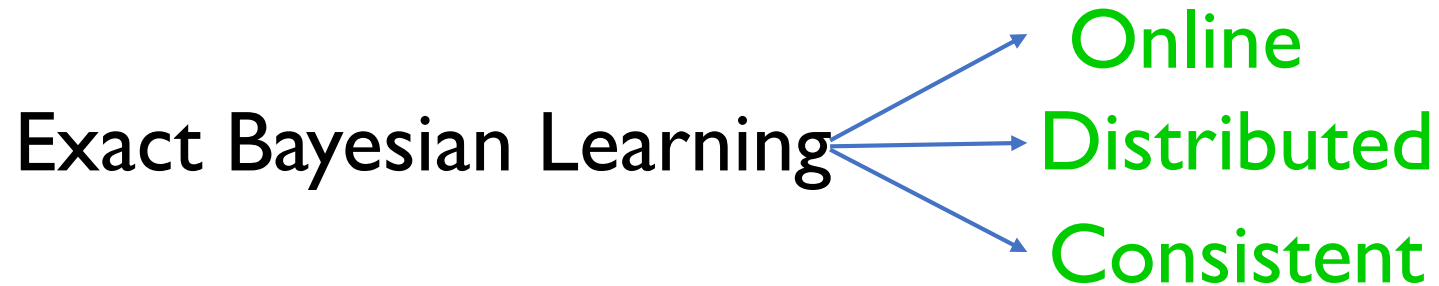
Learning Algorithms

- **Exact Bayesian Learning** : Dirichlet Mixtures(Ghosal et al 1999), Gaussian Mixtures(Lijoi et al, 2005), Non-parametric Problems (Barron et al, 1999), (Freedman, 1999)



Learning Algorithms

- **Exact Bayesian Learning** : Dirichlet Mixtures(Ghosal et al 1999), Gaussian Mixtures(Lijoi et al, 2005), Non-parametric Problems (Barron et al, 1999), (Freedman, 1999)



In theory; practical problems!





Thomas Bayes
(c. 1700-1761)

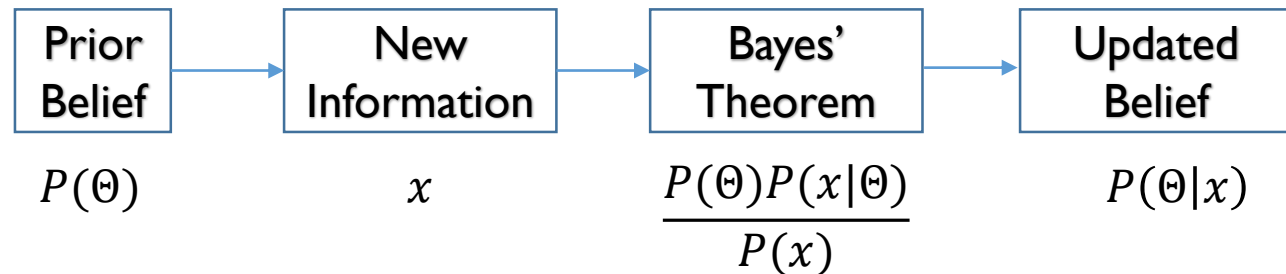
Bayesian Learning



Thomas Bayes
(c. 1700-1761)

- Uses Bayes' Theorem

$$P(\Theta|x) = \frac{P(\Theta)P(x|\Theta)}{P(x)}$$



Bayesian Learning – Mixture models

Bayesian Learning – Mixture models

Data : $\mathbf{x}_{1:n}$ where $\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$

Bayesian Learning – Mixture models

Data : $\mathbf{x}_{1:n}$ where $\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$

$$P_n(\Theta) = \Pr(\Theta | \mathbf{x}^{1:n})$$

Bayesian Learning – Mixture models

Data : $\mathbf{x}_{1:n}$ where $\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$

$$\begin{aligned} P_n(\Theta) &= \Pr(\Theta | x^{1:n}) \\ &\propto P_{n-1}(\Theta) \Pr(x_n | \Theta) \end{aligned}$$

Bayesian Learning – Mixture models

Data : $\mathbf{x}_{1:n}$ where $\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$

$$\begin{aligned} P_n(\Theta) &= \Pr(\Theta | \mathbf{x}^{1:n}) \\ &\propto P_{n-1}(\Theta) \Pr(\mathbf{x}_n | \Theta) \\ &\propto P_{n-1}(\Theta) \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j) \end{aligned}$$

Bayesian Learning – Mixture models

Data : $\mathbf{x}_{1:n}$ where $\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$

$$\begin{aligned} P_n(\Theta) &= \Pr(\Theta | \mathbf{x}^{1:n}) \\ &\propto P_{n-1}(\Theta) \Pr(\mathbf{x}_n | \Theta) \\ &\propto P_{n-1}(\Theta) \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j) \end{aligned}$$

Intractable!!!

Bayesian Learning – Mixture models

Data : $\mathbf{x}_{1:n}$ where $\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$

$$\begin{aligned} P_n(\Theta) &= \Pr(\Theta | x^{1:n}) \\ &\propto P_{n-1}(\Theta) \Pr(x_n | \Theta) \\ &\propto P_{n-1}(\Theta) \sum_{j=1}^M w_j N(x_i; \mu_j, \Sigma_j) \end{aligned}$$

Intractable!!!

Solution : Bayesian Moment Matching Algorithm





Karl Pearson
(c. 1837-1936)

Method of Moments



Karl Pearson
(c. 1837-1936)

- Probability distributions defined by set of parameters
- Parameters can be estimated by a set of moments

$$\begin{aligned}X &\sim N(X; \mu, \sigma^2) \\E[X] &= \mu \\E[(X - \mu)^2] &= \sigma^2\end{aligned}$$

Make Bayesian Learning Great Again

Make Bayesian Learning Great Again

Bayesian Learning
And
Method of Moments

Make Bayesian Learning Great Again

Bayesian Learning

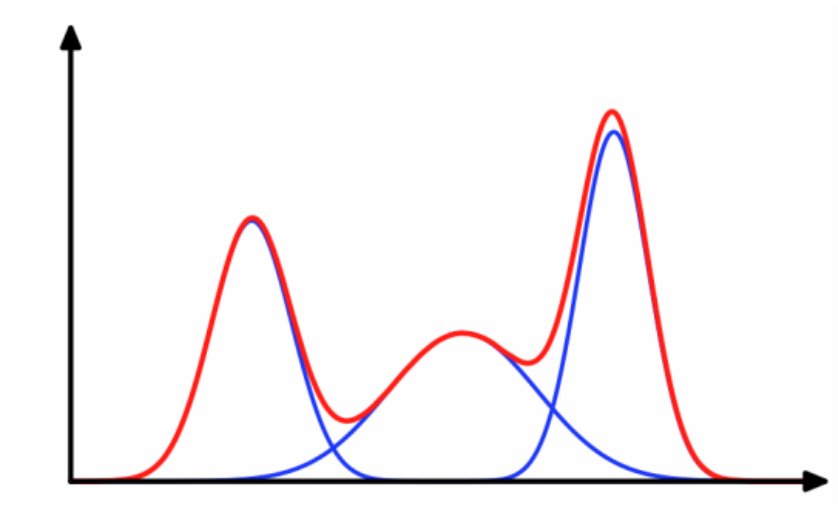
And

Method of Moments

**STRONGER
TOGETHER**

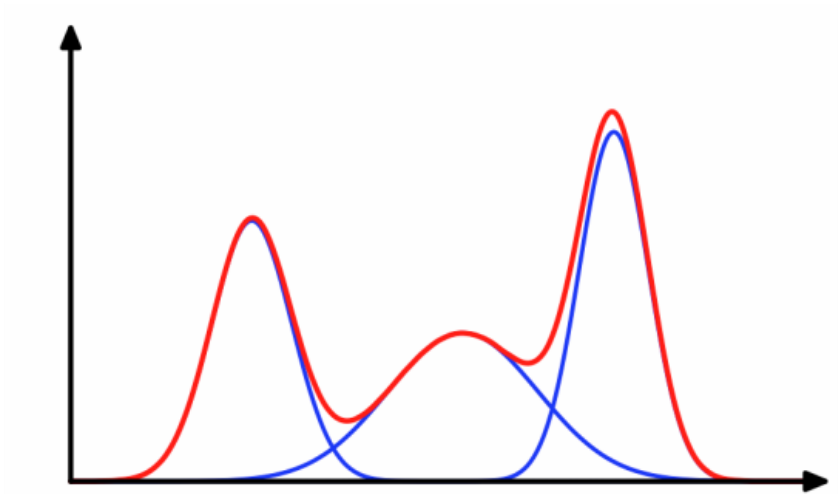
Gaussian Mixture Models

$$\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$$



Gaussian Mixture Models

$$\mathbf{x}_i \sim \sum_{j=1}^M w_j N(\mathbf{x}_i; \mu_j, \Sigma_j)$$



Parameters : **weights**,
means and **precisions**
(inverse covariance matrices)

Bayesian Moment Matching for Gaussian Mixture Models

Bayesian Moment Matching for Gaussian Mixture Models

Parameters : **weights**, **means** and **precisions**
(inverse covariance matrices)

$$P(\Theta|x) = \frac{P(\Theta)P(x|\Theta)}{P(x)}$$

Parameters



Bayesian Moment Matching for Gaussian Mixture Models

Parameters : **weights**, **means** and **precisions**
(inverse covariance matrices)

$$P(\Theta|x) = \frac{P(\Theta)P(x|\Theta)}{P(x)}$$

Prior

Parameters

Prior : $P(\mathbf{w}, \mu, \Lambda)$; product of **Dirichlets** and **Normal-Wisharts**

Bayesian Moment Matching for Gaussian Mixture Models

Parameters : **weights**, **means** and **precisions**
(inverse covariance matrices)

Diagram illustrating the Bayesian formula:

$$P(\Theta|x) = \frac{P(\Theta)P(x|\Theta)}{P(x)}$$

Labels and arrows:

- Prior** points to $P(\Theta)$
- Likelihood** points to $P(x|\Theta)$
- Parameters** points to Θ in $P(\Theta|x)$

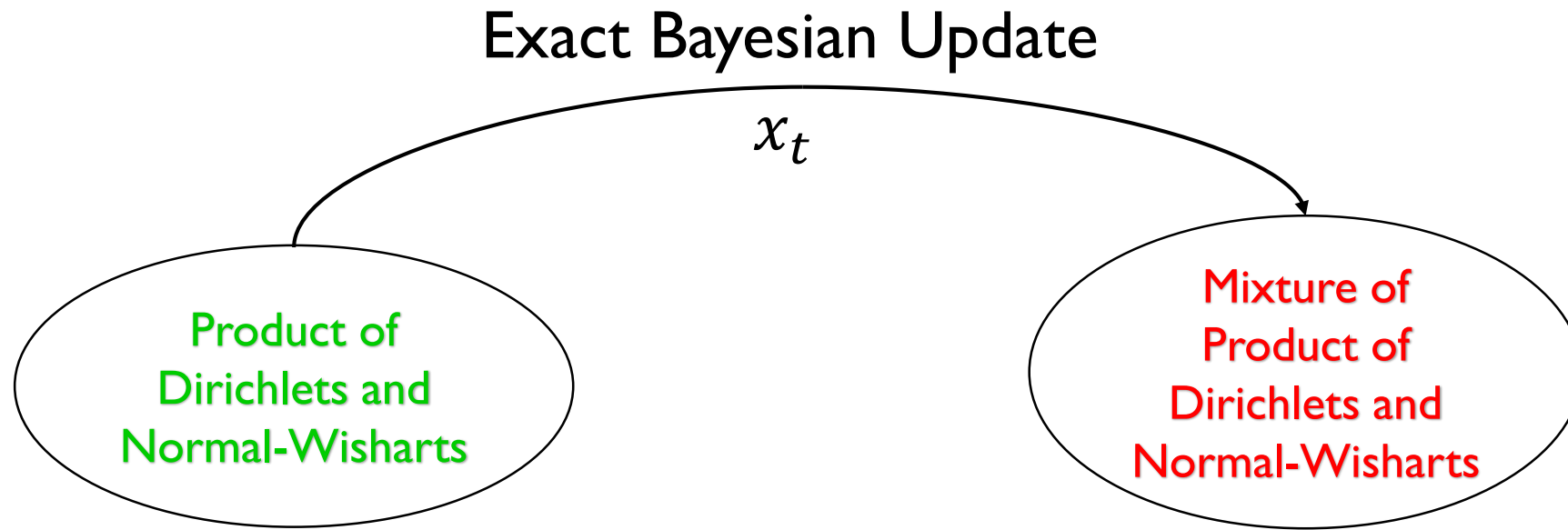
Prior : $P(\mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Lambda})$; product of **Dirichlets** and **Normal-Wisharts**

Likelihood :

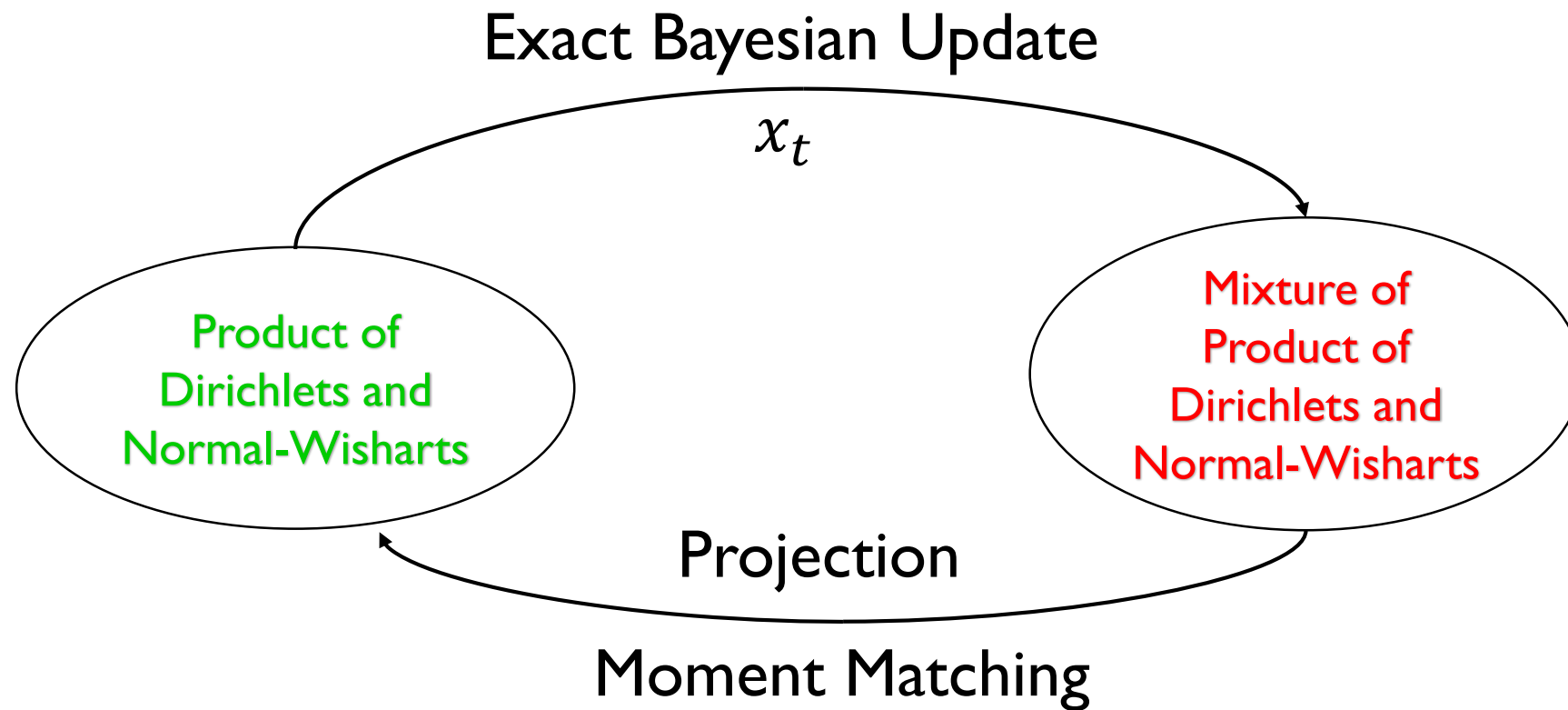
$$P(\mathbf{x} ; \mathbf{w}, \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \sum_{j=1}^M w_j N(\mathbf{x}; \boldsymbol{\mu}_j, \boldsymbol{\Lambda}_j^{-1})$$

Bayesian Moment Matching Algorithm

Bayesian Moment Matching Algorithm



Bayesian Moment Matching Algorithm



Sufficient Moments

Sufficient Moments

Dirichlet : $Dir(w_1, w_2 \dots w_M; \alpha_1, \alpha_2 \dots, \alpha_M)$

Sufficient Moments

Dirichlet : $Dir(w_1, w_2 \dots w_M; \alpha_1, \alpha_2 \dots, \alpha_M)$

$$E[w_i] = \frac{\alpha_i}{\sum_j \alpha_j} ; \quad E[w_i^2] = \frac{\alpha_i(\alpha_i+1)}{(\sum_j \alpha_j)(1+\sum_j \alpha_j)}$$

Sufficient Moments

Dirichlet : $Dir(w_1, w_2 \dots w_M; \alpha_1, \alpha_2 \dots, \alpha_M)$

$$E[w_i] = \frac{\alpha_i}{\sum_j \alpha_j} ; \quad E[w_i^2] = \frac{\alpha_i(\alpha_i+1)}{(\sum_j \alpha_j)(1+\sum_j \alpha_j)}$$

Normal-Wishart : $NW(\mu, \Lambda ; \mu_0, \kappa, W, \nu)$

$$\Lambda \sim Wi(W, \nu) \quad \text{and} \quad \mu|\Lambda \sim N_d(\mu_0, (\kappa\Lambda)^{-1})$$

Sufficient Moments

Dirichlet : $Dir(w_1, w_2 \dots w_M; \alpha_1, \alpha_2 \dots, \alpha_M)$

$$E[w_i] = \frac{\alpha_i}{\sum_j \alpha_j}; \quad E[w_i^2] = \frac{\alpha_i(\alpha_i+1)}{(\sum_j \alpha_j)(1+\sum_j \alpha_j)}$$

Normal-Wishart : $NW(\mu, \Lambda; \mu_0, \kappa, W, \nu)$

$$\Lambda \sim Wi(W, \nu) \quad \text{and} \quad \mu|\Lambda \sim N_d(\mu_0, (\kappa\Lambda)^{-1})$$

$$E[\mu] = \mu_0$$

$$E[(\mu - \mu_0)(\mu - \mu_0)^T] = \frac{\kappa+1}{\kappa(\nu-d-1)} W^{-1}$$

$$E[\Lambda] = \nu W$$

$$Var(\Lambda_{ij}) = \nu(W_{ij}^2 + W_{ii}W_{jj})$$

Overall Algorithm

- Bayesian Step
 - Compute posterior $P_t(\Theta)$ based on observation x_t
- Sufficient Moments
 - Compute set of sufficient moments **S** for $P_t(\Theta)$
- Moment Matching
 - System of linear equations
 - Linear complexity in the number of components

Make Bayesian Learning Great Again

Bayesian Moment Matching Algorithm

- Uses Bayes' Theorem + Method of Moments
- Analytic solutions to Moment matching (unlike EP, ADF)
- One pass over data

Make Bayesian Learning Great Again

Bayesian Moment Matching Algorithm

- Uses Bayes' Theorem + Method of Moments
- Analytic solutions to Moment matching (unlike EP, ADF)
- One pass over data



Experiments

Data Set	Instances	oEM	oBMM
Abalone	4177	-2.65	-1.82
Banknote	1372	-9.74	-9.65
Airfoil	1503	-15.86	-16.53
Arabic	8800	-15.83	-14.99
Transfusion	748	-13.26	-13.09
CCPP	9568	-16.53	-16.51
Comp.Ac	8192	-132.04	-118.82
Kinematics	8192	-10.37	-10.32
Northridge	2929	-18.31	-17.97
Plastic	1650	-9.46	-9.01

Experiments

Data (Features)	Instances	oEM	oBMM	oDMM
Heterogeneity(16)	3,930,257	-176.2	-174.3	-180.7
Magic (10)	19,000	-33.4	-32.1	-35.4
Year MSD (91)	515,345	-513.7	-506.5	-513.8
Miniboone (50)	130,064	-58.1	-54.7	-60.3

Avg. Log-Likelihood

Data (Features)	Instances	oEM	oBMM	oDMM
Heterogeneity(16)	3,930,257	77.3	81.7	17.5
Magic (10)	19,000	7.3	6.8	1.4
Year MSD (91)	515,345	336.5	108.2	21.2
Miniboone (50)	130,064	48.6	12.1	2.3

Running Time

Bayesian Moment Matching

- **Discrete Data** : Omar (2015, PhD Thesis) for Dirichlets; Rashwan, Zhao & Poupart (AISTATS'16) for SPNs; Hsu & Poupart (NIPS'16) for Topic Modelling
- **Continuous Data** : Jaini & Poupart, 2016 (*arxiv*); Jaini, Rashwan et al, (PGM'16) for SPNs; Poupart, Chen, Jaini et al (NetworksML'16)
- **Sequence Data and Transfer Learning** : Jaini, Poupart et al, (submitted to ICLR'17)

Mixture Models

Consistent/Robust

Tensor Decomposition
And
Spectral Learning

Offline

Bayesian
Learning

ADF, EP, SGD, oEM

Non convergence and
local optima

Online

Can be distributed;
Practical problems

Bayesian Moment Matching
algorithm
Consistent?

SPNs

Bayesian Moment
Matching for continuous
SPNs

Comparison with other
deep networks

What is a Sum-Product Network?

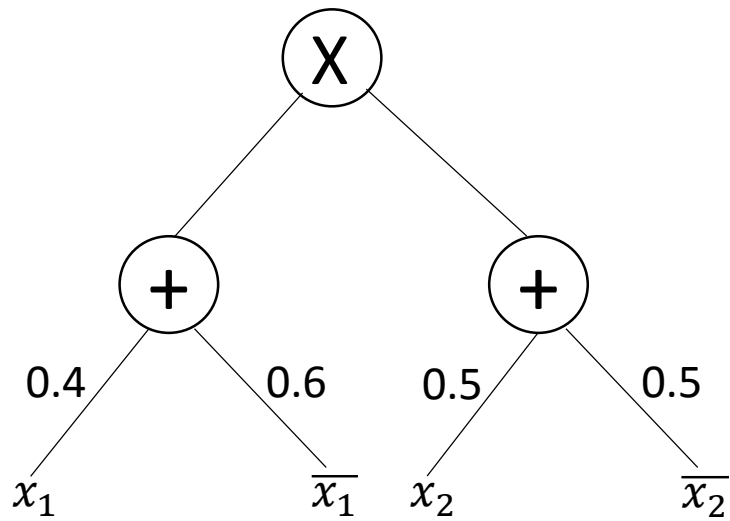
Proposed by Poon and Domingos (UAI 2011)
equivalent to Arithmetic Circuits (Darwiche 2003)

Deep architecture
with clear
semantics

Tractable
Probabilistic
Graphical Models

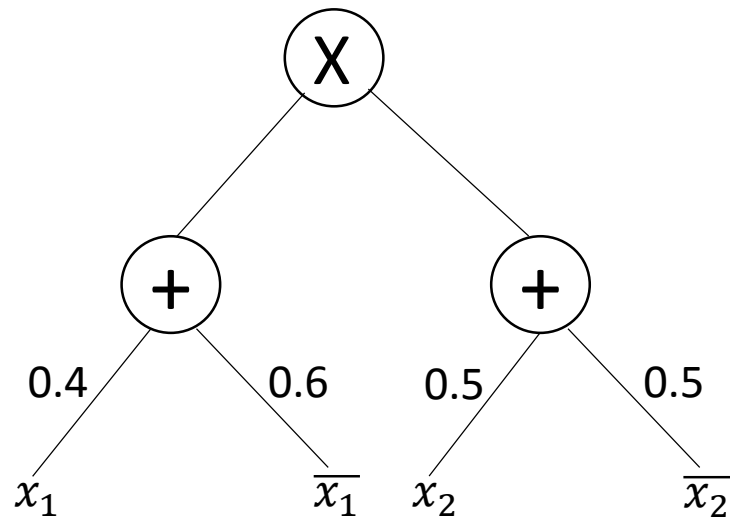
What is a Sum-Product Network?

$$P(x_1, x_2) = P(x_1, x_2)x_1x_2 + P(x_1, \bar{x}_2)x_1\bar{x}_2 + P(\bar{x}_1, x_2)\bar{x}_1x_2 + P(\bar{x}_1, \bar{x}_2)\bar{x}_1\bar{x}_2$$



What is a Sum-Product Network?

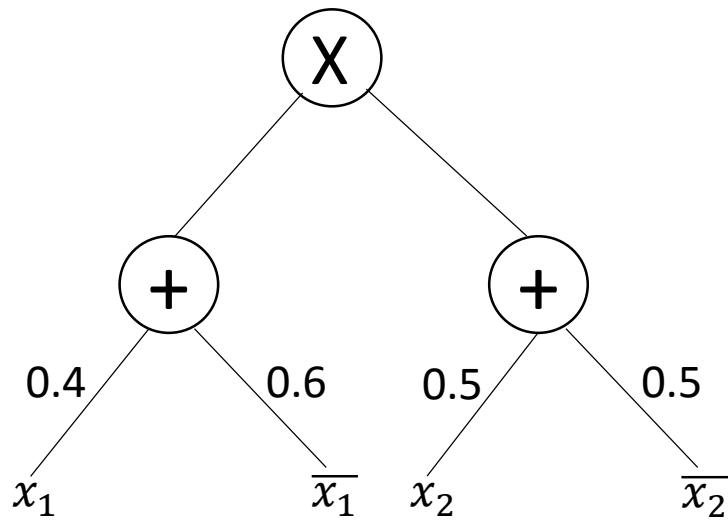
$$P(x_1, x_2) = P(x_1, x_2)x_1x_2 + P(x_1, \bar{x}_2)x_1\bar{x}_2 + P(\bar{x}_1, x_2)\bar{x}_1x_2 + P(\bar{x}_1, \bar{x}_2)\bar{x}_1\bar{x}_2$$



SPNs - Directed Acyclic Graphs

What is a Sum-Product Network?

$$P(x_1, x_2) = P(x_1, x_2)x_1x_2 + P(x_1, \bar{x}_2)x_1\bar{x}_2 + P(\bar{x}_1, x_2)\bar{x}_1x_2 + P(\bar{x}_1, \bar{x}_2)\bar{x}_1\bar{x}_2$$



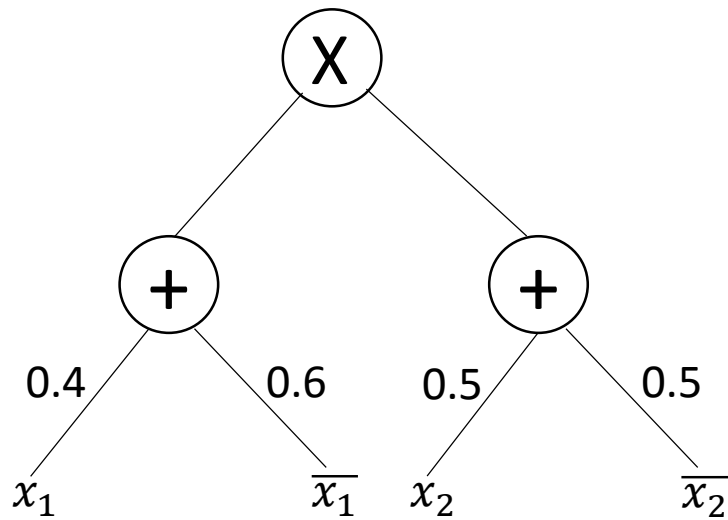
SPNs - Directed Acyclic Graphs

A valid SPN is

- **Complete** : Each sum node has children with same scope

What is a Sum-Product Network?

$$P(x_1, x_2) = P(x_1, x_2)x_1x_2 + P(x_1, \bar{x}_2)x_1\bar{x}_2 + P(\bar{x}_1, x_2)\bar{x}_1x_2 + P(\bar{x}_1, \bar{x}_2)\bar{x}_1\bar{x}_2$$

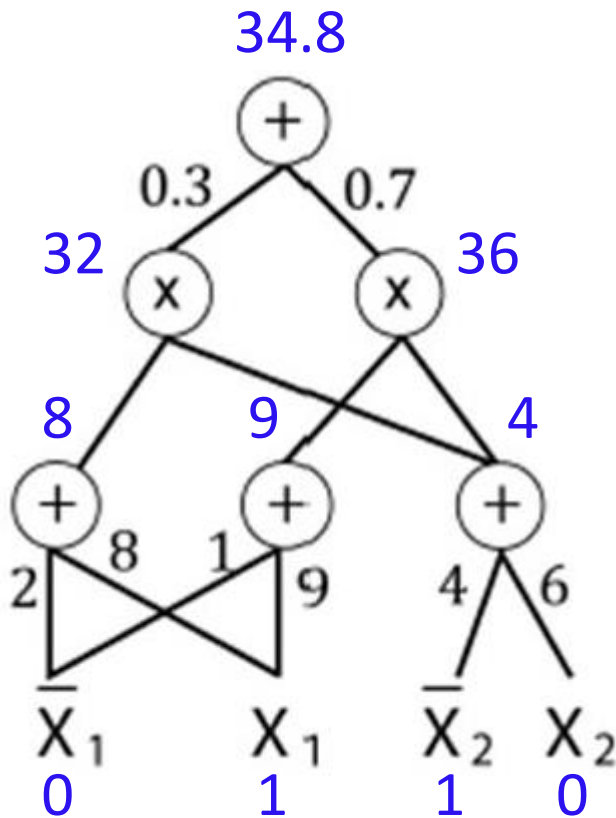


SPNs - Directed Acyclic Graphs

A valid SPN is

- **Complete** : Each sum node has children with same scope
- **Decomposable** : Each product node has children with disjoint scope

Probabilistic Inference - SPN



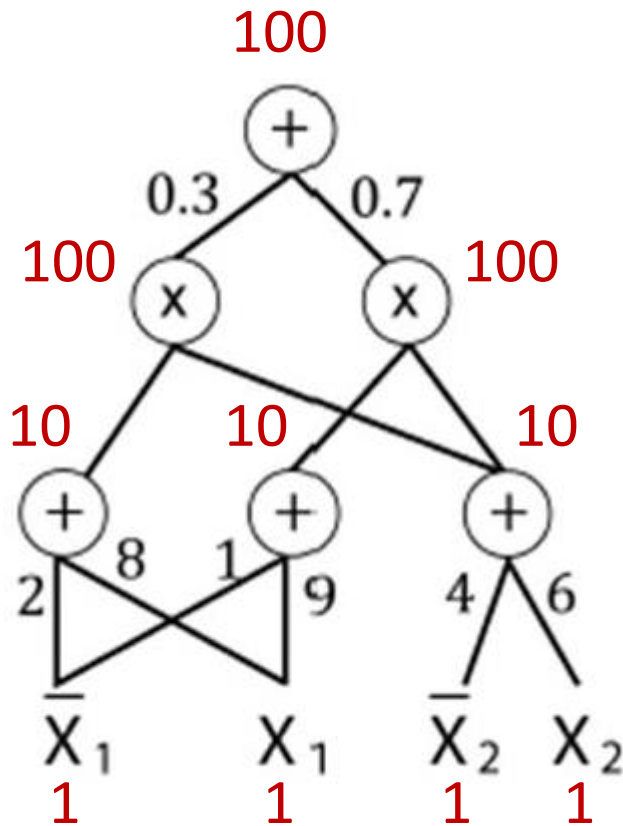
SPN represents a joint distribution over a set of random variables

Example :

Query : $\Pr(X_1 = 1, X_2 = 0)$

$$\Pr(X_1 = 1, X_2 = 0) = 34.8/\text{constant}$$

Probabilistic Inference - SPN



SPN represents a joint distribution over a set of random variables

Example :

Query : $\Pr(X_1 = 1, X_2 = 0)$

$$\Pr(X_1 = 1, X_2 = 0) = 34.8/100$$

Linear Complexity – two bottom passes for any query

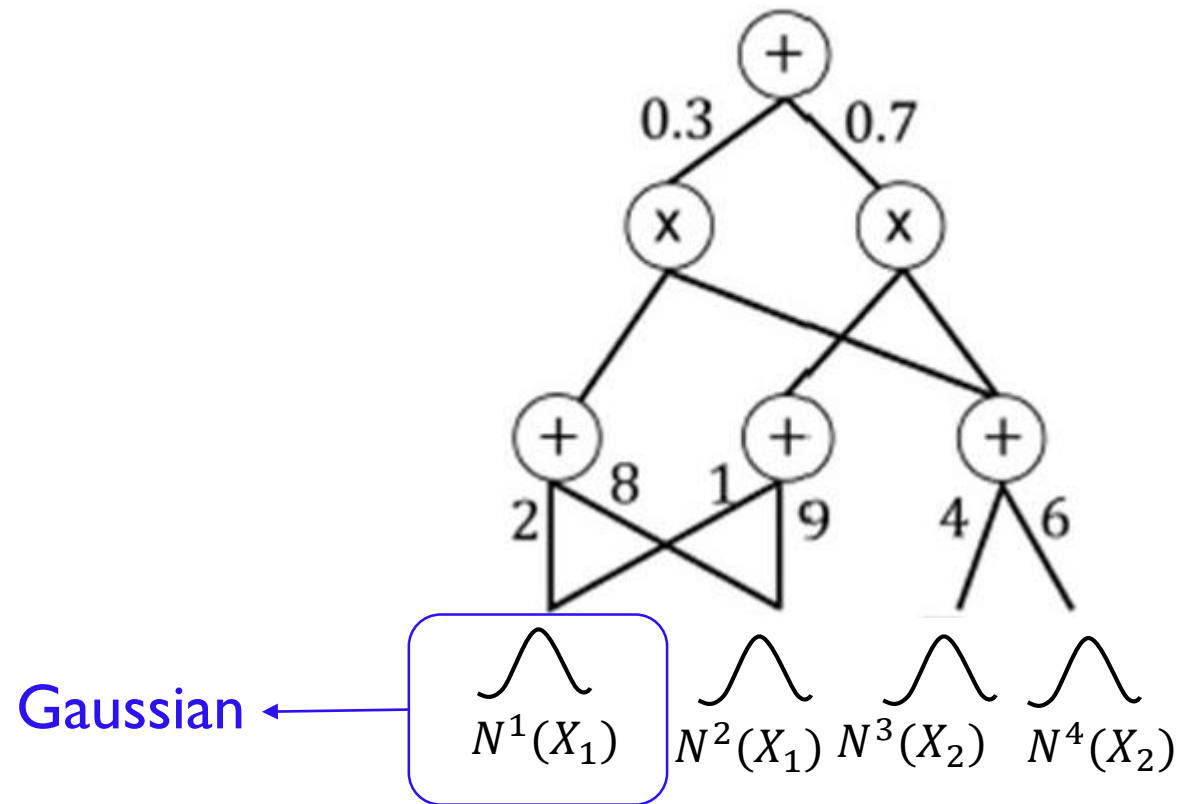
Learning SPNS

- Parameter Learning :

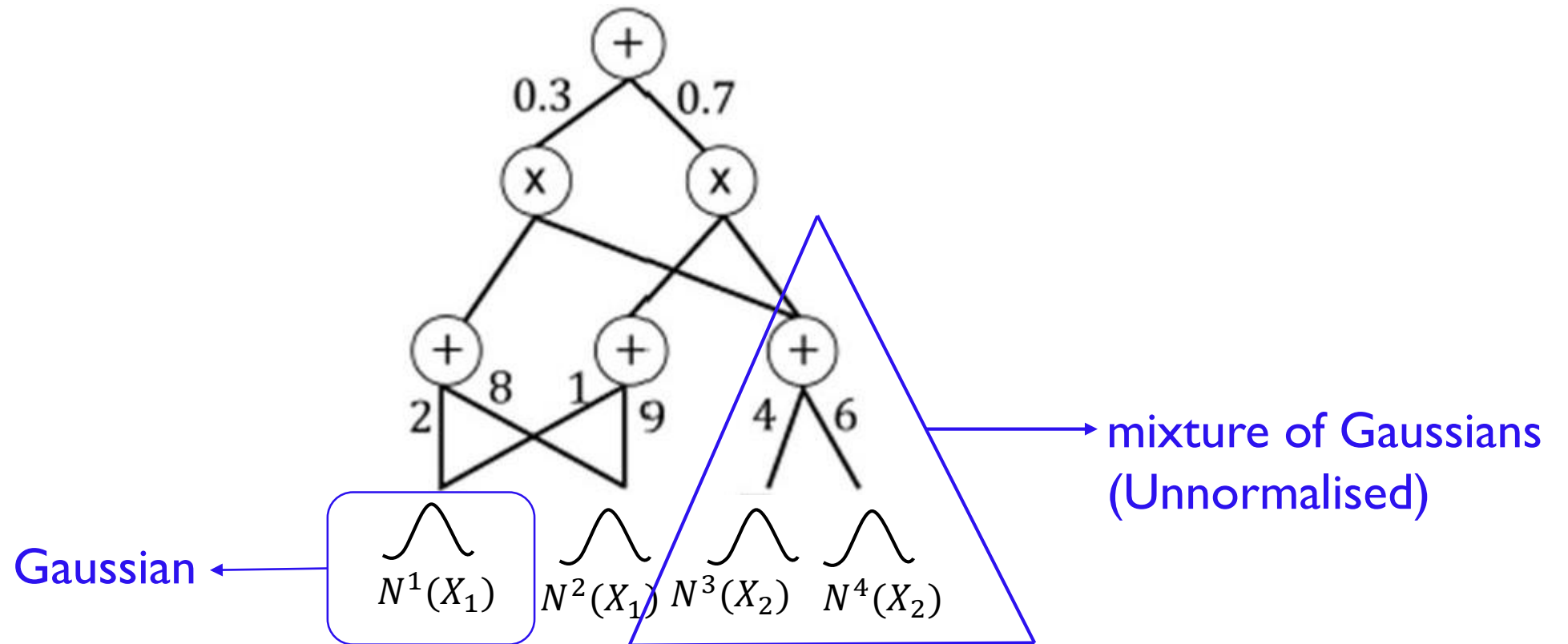
- Discrete {
- **Maximum Likelihood:** SGD (Poon & Domingos, 2011) **slow convergence, inaccurate**; EM(Perharz, 2015) **inaccurate**;
Signomial Programming (Zhao & Poupart, 2016)
 - **Bayesian Learning:** BMM(Rashwan et al., 2016) **accurate**;
Collapsed Variational Inference (Zhao et al., 2016) **accurate**

- Online parameter learning for continuous SPNs (Jaini et al.2016)
 - **extend oBMM to Gaussian SPNS**

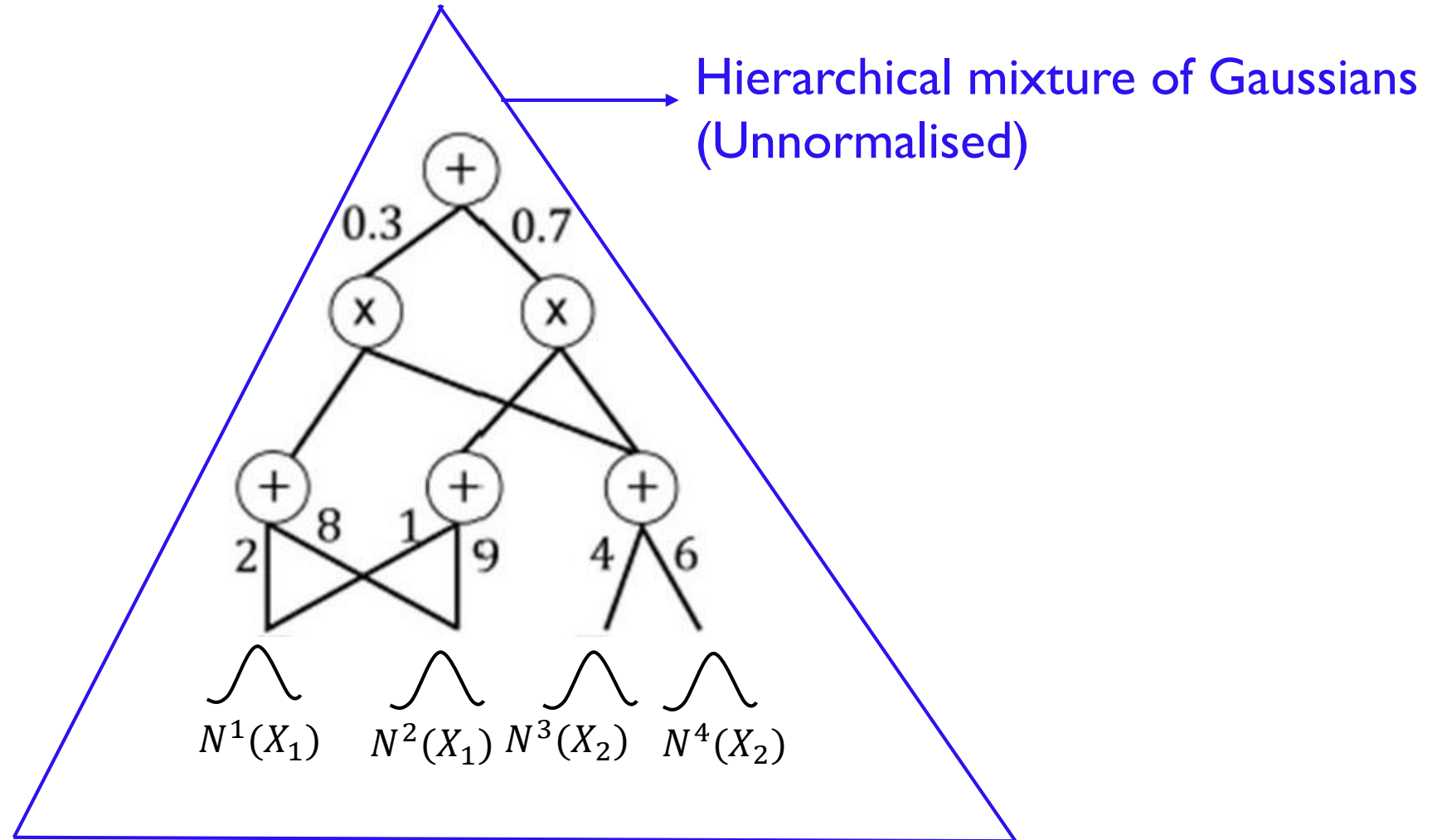
Continuous SPNS



Continuous SPNS



Continuous SPNS



Bayesian Learning of SPNs

Parameters : **weights**, **means** and **precisions** (inverse covariance matrices)

Prior : $P(\mathbf{w}, \mu, \Lambda)$; product of **Dirichlets** and **Normal-Wisharts**

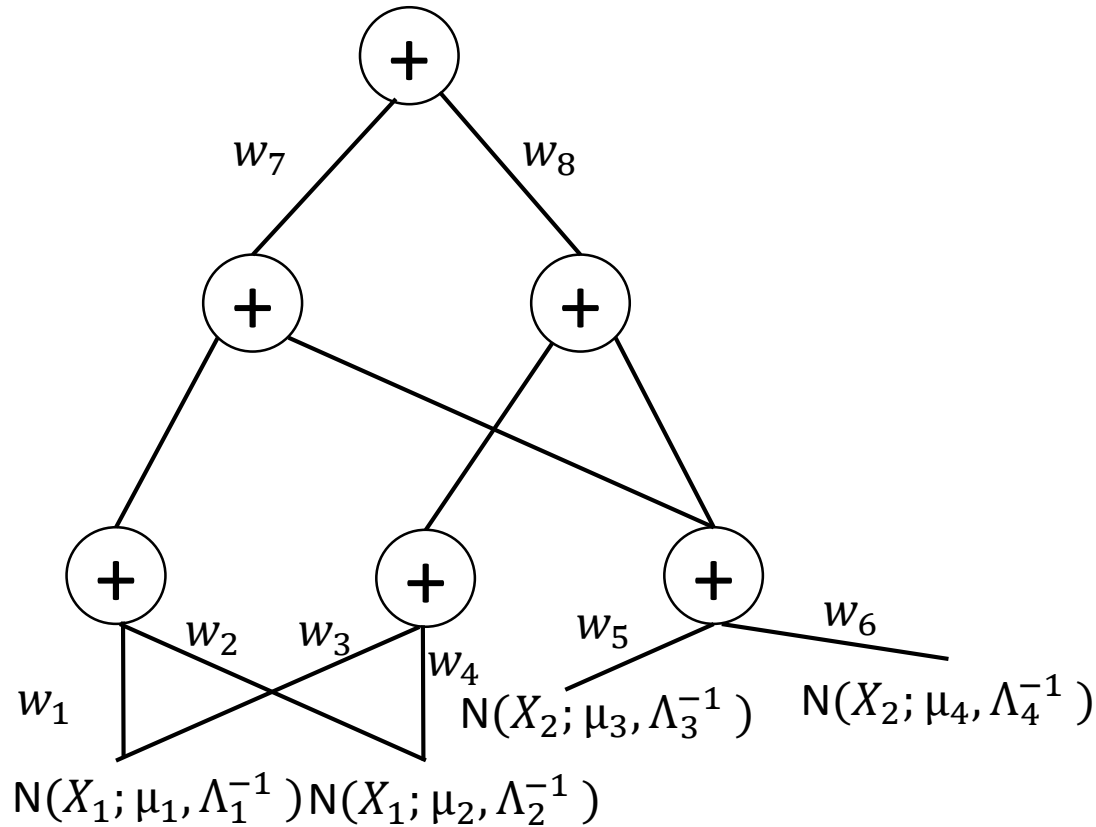
Likelihood :

$$P(\mathbf{x} ; \mathbf{w}, \mu, \Lambda) = SPN(\mathbf{x} ; \mathbf{w}, \mu, \Lambda)$$

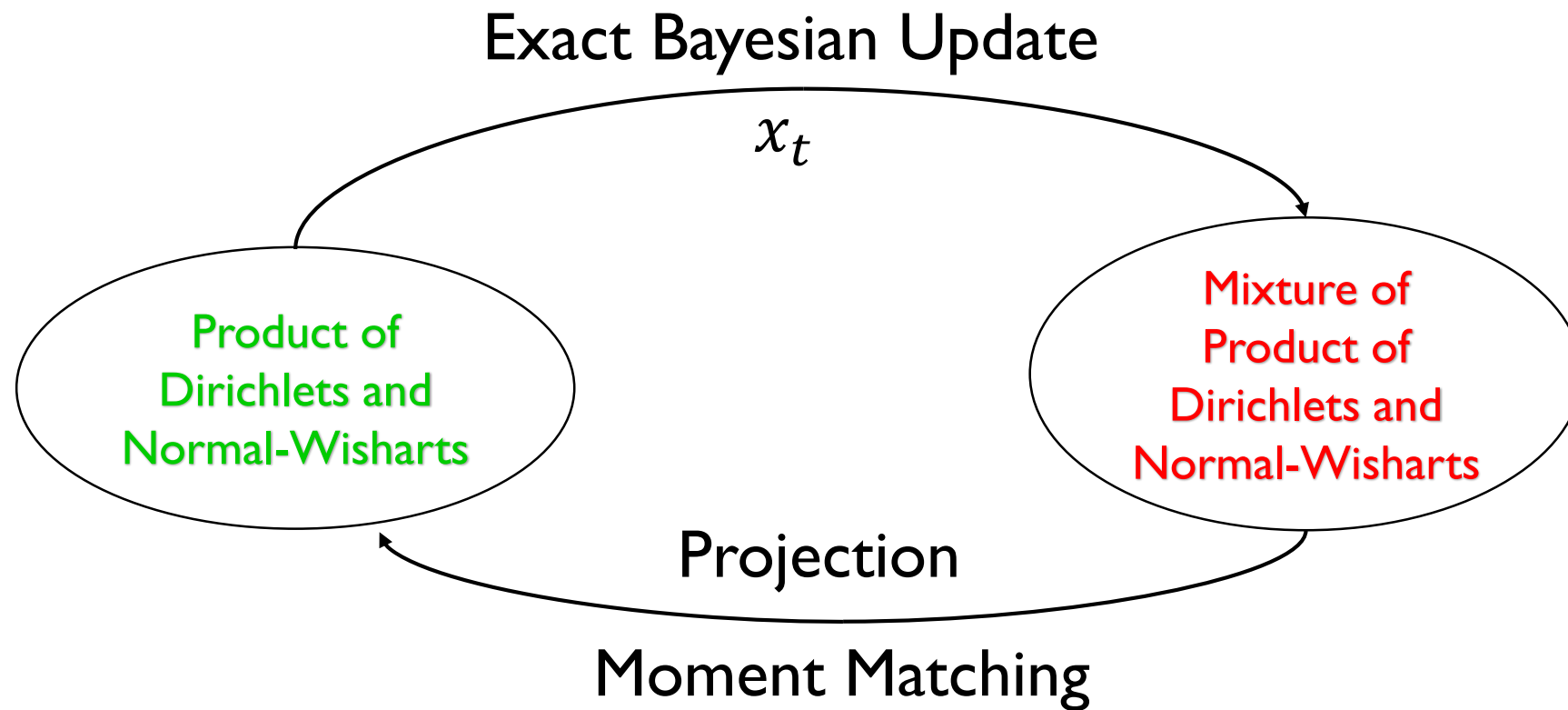
Posterior :

$$P(\mathbf{w}, \mu, \Lambda; \mathbf{data}) \propto$$

$$P(\mathbf{w}, \mu, \Lambda) \prod_n SPN(\mathbf{x}_n ; \mathbf{w}, \mu, \Lambda)$$



Bayesian Moment Matching Algorithm



Overall Algorithm

- Recursive computation of all constants
 - Linear complexity in the size of SPN
- Moment Matching
 - System of linear equations
 - Linear complexity in the size of SPN
- Streaming Data
 - Posterior update : constant time w.r.t. amount of data

Empirical Results

Avg. log-likelihood and standard error based on 10-fold cross-validation

Dataset	Flow Size	Quake	Banknote	Abalone	Kinematics	CA	Sensorless Drive
Attributes	3	4	4	8	8	22	48
oBMM (random)	-	-	-	-1.82 $\mp$ 0.19	-11.19 $\mp$ 0.03	-2.47 $\mp$ 0.56	1.58 $\mp$ 1.28
oEM (random)	-	-	-	-11.36 $\mp$ 0.19	-11.35 $\mp$ 0.03	-31.34 $\mp$ 1.07	-3.40 $\mp$ 6.06
oBMM (GMM)	4.80 $\mp$ 0.67	-3.84 $\mp$ 0.16	-4.81 $\mp$ 0.13	-1.21 $\mp$ 0.36	-11.24 $\mp$ 0.04	-1.78 $\mp$ 0.59	-
oEM (GMM)	-0.49 $\mp$ 3.29	-5.50 $\mp$ 0.41	-4.81 $\mp$ 0.13	-3.53 $\mp$ 1.68	-11.35 $\mp$ 0.03	-21.39 $\mp$ 1.58	-

oBMM performs better than oEM

Empirical Results

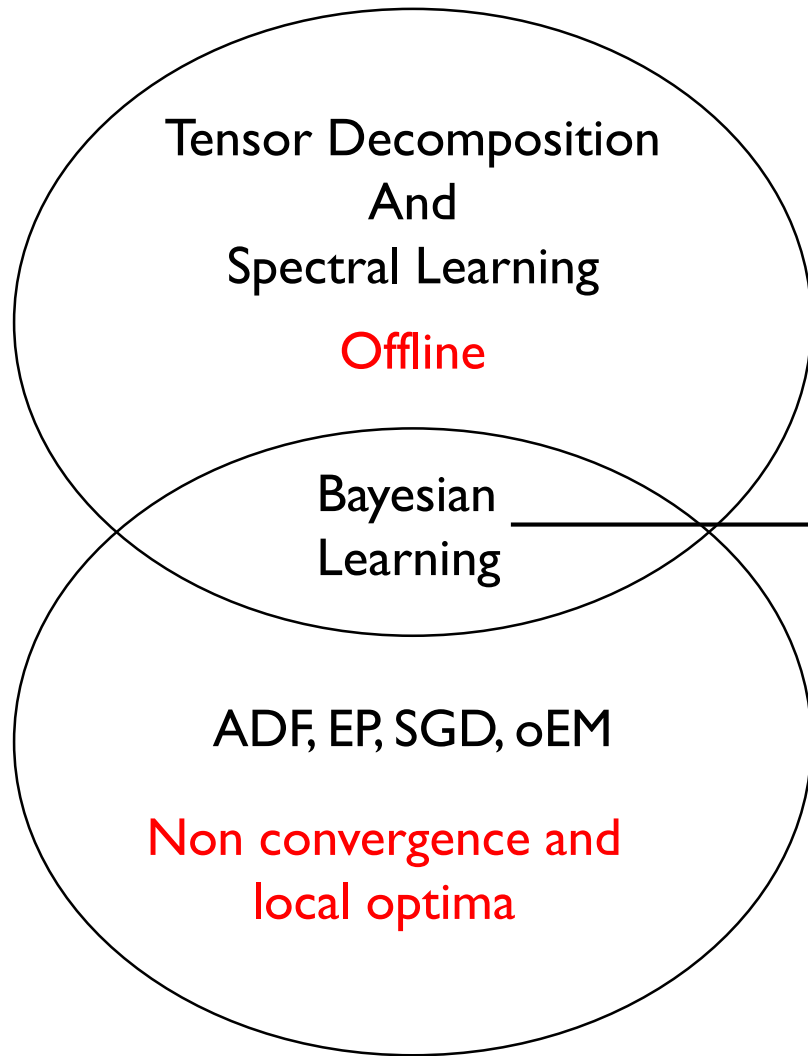
Avg. log-likelihood and standard error based on 10-fold cross-validation

Dataset	Flow Size	Quake	Banknote	Abalone	Kinematics	CA	Sensorless Drive
Attributes	3	4	4	8	8	22	48
oBMM (random)	-	-	-	-1.82 $\pm$ 0.19	-11.19 $\pm$ 0.03	-2.47 $\pm$ 0.56	1.58 $\pm$ 1.28
oBMM (GMM)	4.80 $\pm$ 0.67	-3.84 $\pm$ 0.16	-4.81 $\pm$ 0.13	-1.21 $\pm$ 0.36	-11.24 $\pm$ 0.04	-1.78 $\pm$ 0.59	-
SRBM	-0.79 $\pm$ 0.01	-2.38 $\pm$ 0.01	-2.76 $\pm$ 0.01	-2.28 $\pm$ 0.01	-5.55 $\pm$ 0.02	-4.95 $\pm$ 0.01	-26.91 $\pm$ 0.03
GenMMN	-0.40 $\pm$ 0.01	-3.83 $\pm$ 0.01	-1.70 $\pm$ 0.03	-3.29 $\pm$ 0.10	-11.36 $\pm$ 0.02	-5.41 $\pm$ 0.14	-29.41 $\pm$ 1.19

oBMM is competitive with SRBM and GenMMN

Mixture Models

Consistent/Robust



Tensor Decomposition
And
Spectral Learning

Offline

Bayesian
Learning

ADF, EP, SGD, oEM

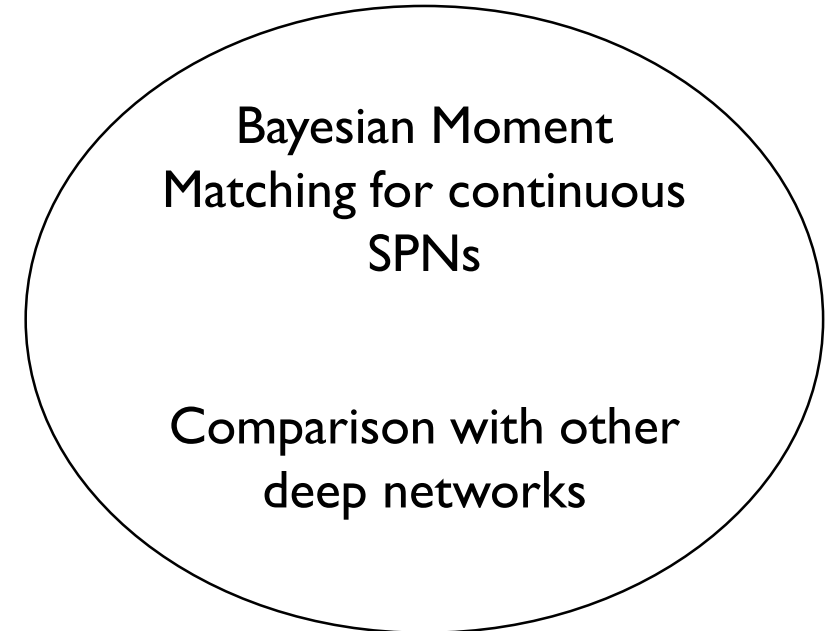
Non convergence and
local optima

Online

Can be distributed;
Practical problems

Bayesian Moment Matching
algorithm
Consistent?

SPNs



Bayesian Moment
Matching for continuous
SPNs

Comparison with other
deep networks

oBMM performs better
than oEM and
comparable to other
techniques

Conclusion and Future Work

Contributions :

- Online and distributed Bayesian Moment Matching algorithm
- Performs better than oEM w.r.t time and accuracy
- Extended it to continuous SPNs
- Comparative analysis with other deep learning methods

Future Work :

- Theoretical properties of BMM – consistent?
- Generalize oBMM to exponential family
- Extension to sequential data and transfer learning
- Online structure learning of SPNs

Thank you!