Efficient Representation and Fusion of Hybrid Joint Densities for Clusters in Nonlinear Hybrid Bayesian Networks

Oliver C. Schrempf, Anne Hanselmann, and Uwe D. Hanebeck
Intelligent Sensor-Actuator-Systems Laboratory
Institute of Computer Science and Engineering
Universität Karlsruhe (TH), Germany
schrempf@ieee.org, hanselm@ira.uka.de, uwe.hanebeck@ieee.org

Abstract - ***Undirected cycles in Bayesian networks are often treated by using clustering methods. This results in networks with nodes characterized by joint probability densities instead of marginal densities. An efficient representation of these hybrid joint densities is essential especially in nonlinear hybrid networks containing continuous as well as discrete variables. In this article we present a unified representation of continuous, discrete, and hybrid joint densities. This representation is based on Gaussian and Dirac mixtures and allows for analytic evaluation of arbitrary hybrid networks without loosing structural information, even for networks containing clusters. Furthermore we derive update formulae for marginal and joint densities from a system theoretic point of view by treating a Bayesian network as a system of cascaded subsystems. Together with the presented mixture representation of densities this yields an exact analytic updating scheme.***

Keywords: Bayesian Networks, Joint Densities, Joint Density Fusion, Nonlinear Systems, Hybrid Systems, Cascaded Systems

1 Introduction

Bayesian networks are very popular in the field of information fusion what can be seen from the vast number of publications. To mention only a few examples, see [13], [12], and [7]. Bayesian networks in general are used for representing causal structures and probabilistic dependencies in complex systems. They belong to the family of graphical models. The models are represented by an acyclic graph structure. The nodes of such a graph represent random variables whereas edges depict dependencies between variables. These dependencies are modeled by means of conditional probabilities or conditional probability density functions. Most Bayesian network implementations found today use only discrete random variables. This approach by Pearl [9] is very famous and uses conditional probability tables (CPT) to describe the dependencies between variables. In his work, a method for continuous variables is mentioned as well, but it considered exclusively continuous variables with linear dependencies and Gaussian probability density functions. A first approach to purely continuous Bayesian networks with nonlinear dependencies was given in [1]. They use Gaussian mixtures to represent the more complex densities that occur when nonlinearities are considered.

A well known approach to hybrid Bayesian networks consisting of continuous as well as of discrete random variables is given in [4], where so called cg-potentials are used to describe the hybrid densities. The problem with this approach is that only mean and variance of the continuous variables are considered, which lets the algorithm fail to represent multimodal density functions that occur in complex hybrid networks, even if only linear dependencies are considered. An example for this failure is given in [10]. Another problem of the original cg-potentials is that the algorithm cannot deal with discrete children of continuous parent nodes. A first approach to solve this by variational approximations is given in [8]. An exact solution, that uses softmax conditional densities is given in [5]. Both of these approaches, however, can only deal with linear dependencies. In [10] and [11] we presented a method for modeling hybrid Bayesian networks with nonlinear dependencies having restrictions concerning discrete children of continuous parents. In this work the mixture approach of [1] was extended for the hybrid case.

For a given Bayesian network we are mainly interested in the marginal densities of the variable nodes given observations of other variables in the network. A popular algorithmic approach to this problem is the so called message passing algorithm presented by Pearl in [9]. It has been shown that this algorithm is an instance of the sum-product algorithm on factor graphs presented in [3]. This algorithm exploits the graph structure of the Bayesian network to pass information concerning observations or a-priori knowledge from neighbor to neighbor while updating the marginal densities. Although this approach is very popular, it uses awkward formulations for the density update problem. Hence, we derive more generic update formulae in this article.

A main restriction of the message passing or sum-product algorithm is that it only works for so called singly connected networks. This means the underlying graph must be free of cycles. Even though directed cycles are forbidden in Bayesian networks per se, the algorithm can further not deal with undirected cycles. This is due to the fact, that updates are also passed against the direction of the edges in the graph. This is also called a Bayesian backward step in contrast to the Bayesian forward step that calculates the update in direction of the edges. A standard approach to cope with cycles in Bayesian networks is to merge the nodes forming the cycle into a single node, which is called clustering. The easiest way for doing this is to replace the marginal densities of the variables in the cycle by their joint density. A more sophisticated approach to clustering is the so called junction tree described in [2]. This algorithm reorganizes the graph structure by means of moralization and triangulation operations. In the resulting graph, maximum cliques are identified as clustered nodes for a new singly connected graph. A drawback of this approach is that the original graph structure is hard to rediscover and the variables of the original graph appear in several clique nodes.

No matter what approach is used for clustering, the resulting nodes contain more than one random variable. This means, that the density of this node is represented by a joint probability density function. In the case of hybrid Bayesian networks with nonlinear dependencies this results in hybrid joint densities that cannot be fully described by first and second moments (mean and variance). Therefore, we will present a parametric representation of hybrid joint densities with arbitrary complexity.

The remainder of this paper is structured as follows. We first give a problem formulation in Section 2. In Section 3 we derive the general update rules for arbitrary density functions from a system theoretic point of view. Section 4 is dedicated to the representation of density functions and hybrid conditionals by means of mixtures. In Section 5 we then consider hybrid joint densities. Conclusions are given in Section 6

2 Problem Formulation

Cycles in Bayesian networks, as shown in the example on the left hand side of Figure 1, are usually removed by application of clustering methods. A clustered version of the example is shown on the right hand side of Figure 1.

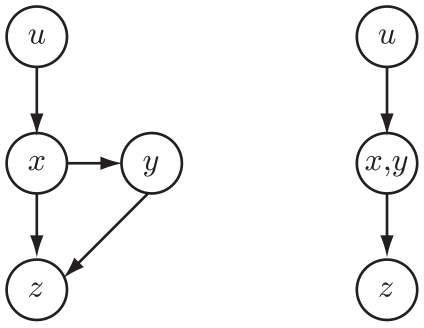

Figure 1: A cycle in the left network is removed by clustering the nodes x and y as depicted in the right network.

Clustered nodes have to be treated like all the other nodes in the network. They receive density functions from their neighbor nodes to update their own probability density and send information back. On the other hand, cluster nodes consist of more than one variable, hence, the density to describe this node is a joint density. Due to this, we cannot update the marginal densities of the variables in a cluster directly.

In the example shown in Figure 1, we do not update $f(x)$ and $f(y)$ directly. Instead we update $f(x, y)$. Since y depends on x, we have to consider the conditional density $f(y \,|\, x)$ when updating the joint density. Hence, we have

$$f(x, y) = f(y \,|\, x) f(x) \ .$$

It is very important, that the resulting joint density is of the same type as the marginal densities. Only in this case the updating algorithm can stay untouched. Further, we do not want to lose information on the structure of the density.

A simple example is the joint density shown in Figure 2. This is the joint density resulting from a Gaussian prior $f(x) = N(x, 0, 2)$ with $N(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\}$ and a nonlinear dependency $y = \sin(x) + v$ with additive noise v. It is clear from this example, that the resulting joint density cannot be represented by a single multivariate Gaussian without loosing information.

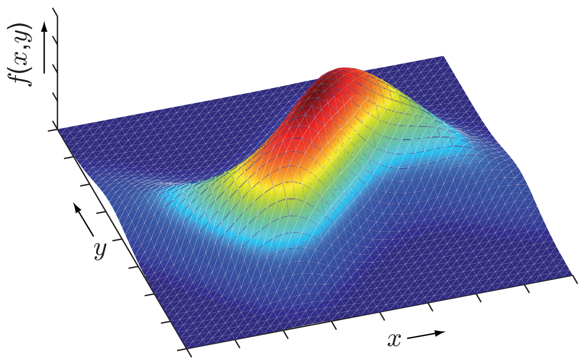

Figure 2: A joint density resulting from a Gaussian prior and a nonlinear dependency.

This kind of joint densities may be the result of clustering techniques that where applied to reduce cycles. Since these clustered cycles are cycles induced by dependencies, we have to take special care of these joint and conditional densities. Hence, the aim of this paper is to give an integrated view on joint and conditional densities that allows for a unified update approach in hybrid Bayesian networks.

Another issue with respect to this unified approach is the fusion of joint densities with only partly overlapping sets of random variables. We need to make sure, that this operation does not change the type of the density representation. Hence the representation of $f(x, y)$, $f(y, z)$, and $f(x, y, z) = f(x, y) f(y, z)$ must be the same, no matter if the overlapping variable y is continuous or discrete.

3 Update Rules

Before we give a representation for continuous and discrete density functions, we first derive the update rules for generic marginal densities in a more system theoretic approach. For this purpose we interpret a Bayesian network as a system of cascaded subsystems. Figure 3 shows a representation of the most general subsystem we have to consider. We use a block diagram representation with random variables on the edges and conditional densities in the blocks to emphasize the system theoretic viewpoint of this derivation. This block diagram representation is very close

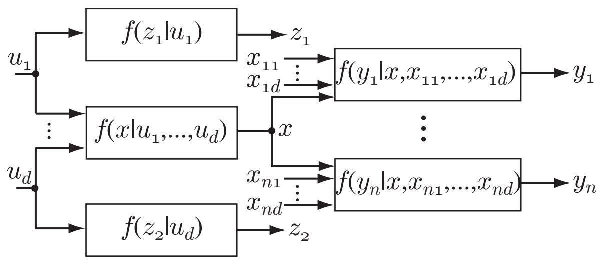

Figure 3: The most general subsystem of a cascaded system.

to the factor graph representation of Bayesian networks introduced in [3] and [6].

We are interested in the marginal density of x. This marginal density is governed by information coming from the left side of x denoted by $f^p(x)$ and information coming from the right side of x denoted by $f^e(x)$. Since $f^p(x)$ and $f^e(x)$ are independent, we can calculate the marginal density by

$$f(x) = cf^p(x)f^e(x) \ ,$$

where c is a normalizing constant. The density $f^p(x)$ coming from the left part is the result of a Bayesian forward or prediction step

$$f^p(x) = \int_{\mathbb{R}^d} f(x \,|\, u_1, \ldots, u_d) \prod_{i=1}^{d} f(u_i)\backslash_x \mathrm{d}u_1 \ldots \mathrm{d}u_d \ . \quad (1)$$

In this equation $f(u_i)\backslash_x$ is the marginal density of u_i where the information coming from x itself is ignored, since we want to update $f(x)$. Taking the full marginal density $f(u_i)$ into account would result in an double count of the information from x. We will denote this reduced density by

$$f(u_i)\backslash_x = \frac{f(u_i)}{f^e_x(u_i)}$$

to emphasize that this is the marginal density over u_i without any information coming from x. Due to the recursive update scheme it is not necessary to devide $f(u_i)$ by $f^e_x(u_i)$. It is sufficient to ignore it when updating $f(x)$.

Throughout this derivation we further assume d to be the input degree of the corresponding subsystem. This number may vary from subsystem to subsystem.

The density $f^e(x)$ coming from the right part is the product of densities $f^e_j(x)$ received from each subsystem connected to x as result of a Bayesian backward or filter step

$$\begin{aligned} f^e(x) &= \prod_{j=1}^{n} f^e_j(x) \\ f^e_j(x) &= \int_{\mathbb{R}^{d+1}} f(y_j \,|\, x, x_{j1}, \ldots, x_{jd}) f^e(y_j) \\ &\quad \cdot \prod_{k=1}^{d} f^p(x_{jk}) \mathrm{d}y_j \, \mathrm{d}x_{j1} \ldots \mathrm{d}x_{jd} \end{aligned} \quad (2)$$

In the derivation of $f^e_j(x)$ all information coming from x itself is ignored. Hence, we omit $f^p(x)$ in $f^e_j(x)$.

For a specific observation $y_j = \hat{y}_j$ we set $f^e(y_j) = \delta(y_j - \hat{y}_j)$. Applying this to (2) results in

$$\begin{aligned} f^e_j(x) &= f^e_j(x \,|\, \hat{y}_j) \\ &= \int_{\mathbb{R}^{d+1}} f(y_j \,|\, x, x_{j1}, \ldots, x_{jd}) \delta(y_j - \hat{y}_j) \\ &\quad \cdot \prod_{k=1}^{d} f^p(x_{jk}) \mathrm{d}y_j \, \mathrm{d}x_{j1} \ldots \mathrm{d}x_{jd} \\ &= \int_{\mathbb{R}^{d}} f(\hat{y}_j \,|\, x, x_{j1}, \ldots, x_{jd}) \\ &\quad \cdot \prod_{k=1}^{d} f^p(x_{jk}) dx_{j1}, \ldots, dx_{jd} \ . \end{aligned}$$

For unobserved outputs y_j we set $f^e(y_j) = 1$. In general, observations for any variable are modeled by setting

$$f(x = \hat{x}) = f^p(x = \hat{x}) = f^e(x = \hat{x}) = \delta(x - \hat{x}) \ .$$

In the next section we will give representations of continuous, discrete, and hybrid joint densities that allow for solving the integrals in (1) and (2) analytically.

4 Mixture Representation for Densities and Conditional Densities

As pointed out in the problem formulation, the resulting structure of densities in cascaded nonlinear systems can become very complex. Hence, we apply mixture representations of the type

$$f(\,\cdot\,) = \sum_{i=1}^{M} \alpha_i g_i(\,\cdot\,)$$

to represent arbitrary densities, where $g_i(\,\cdot\,)$ is a probability density function and M is the number of components in this mixture. α_i are weighting factors with the property

$$\sum_{i=1}^{M} \alpha_i = 1 \ .$$

Since we want to consider joint densities, we use multivariate mixtures of the type

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{M} \alpha_i g_i(x_1, \ldots, x_n) \ .$$

For the sake of computability we use mixtures with axis-aligned components. This means, that each component $g_i(x_1, \ldots, x_n)$ can be decomposed by the product $g_i(x_1) \cdot \ldots \cdot g_i(x_n)$ with $g_i(x_j) \neq g_i(x_k)$ for all $x_j \neq x_k$. The full mixture

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{M} \alpha_i \prod_{j=1}^{n} f_{i,j}(x_j)$$

however, is not axis-aligned. This can be seen in the example of Figure 4. This is the contour plot of a mixture density

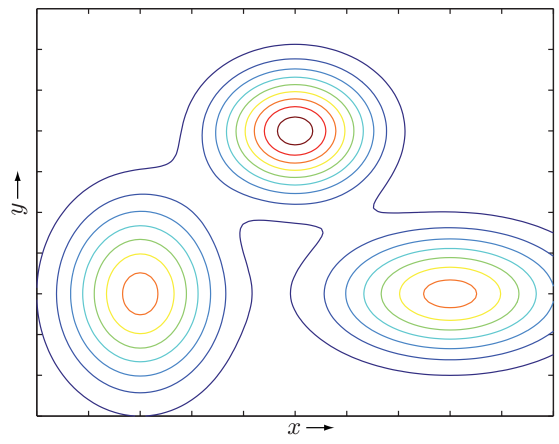

Figure 4: Contourplot of density function with four axis-aligned components.

with 3 components. Each of the three components is symmetric with respect to the x and the y axis. The resulting mixture however is not sysmetric with respect to the axes.

Our aim in this article is to provide a unified view of joint and conditional densities. Hence, we would like to represent conditional density functions by means of mixtures, too. Since conditional densities in general are no density functions, we approximate the conditional density $f(x \,|\, u_1, \ldots, u_d)$ by means of a mixture

$$f(x \,|\, u_1, \ldots, u_d) \approx \sum_{i=1}^{M} \gamma_i f_i(x) \prod_{j=1}^{d} f_{i,j}(u_j)$$

with axis-aligned components. The update densities (1) and (2) can then be rewritten as

$$\begin{aligned} & f^p(x) \\ & = \int_{\mathbb{R}^d} f(x \,|\, u_1, \ldots, u_d) \prod_{i=1}^{d} f(u_i)\backslash_x \mathrm{d}u_1 \, \ldots \, \mathrm{d}u_d \\ & \approx \int_{\mathbb{R}^d} \sum_{i=1}^{M} \gamma_i f_i(x) \prod_{j=1}^{d} f_{i,j}(u_j) \prod_{k=1}^{d} f(u_k)\backslash_x \mathrm{d}u_1 \, \ldots \, \mathrm{d}u_d \\ & = \sum_{i=1}^{M} \gamma_i f_i(x) \prod_{j=1}^{d} \int_{\mathbb{R}} f_{i,j}(u_j) f(u_j)\backslash_x \mathrm{d}u_j \qquad (3) \end{aligned}$$

and

$$\begin{aligned} & f_j^e(x) \\ = & \int_{\mathbb{R}^{d+1}} f(y_j \,|\, x, x_{j1}, \ldots, x_{jd}) f^e(y_j) \\ & \cdot \prod_{k=1}^{d} f^p(x_{jk}) \mathrm{d}y_j \, \mathrm{d}x_{j1} \, \ldots \, \mathrm{d}x_{jd} \\ \approx & \int_{\mathbb{R}^{d+1}} \sum_{i=1}^{M} \gamma_i f_i(y_j) f_i(x) \prod_{k=1}^{d} f_{i,k}(x_{jk}) f^e(y_j) \\ & \cdot \prod_{k=1}^{d} f^p(x_{jk}) \mathrm{d}y_j \, \mathrm{d}x_{j1} \, \ldots \, \mathrm{d}x_{jd} \\ = & \sum_{i=1}^{M} \gamma_i f_i(x) \int_{\mathbb{R}} f_i(y_j) f^e(y_j) \mathrm{d}y_j \\ & \cdot \prod_{k=1}^{d} \int_{\mathbb{R}} f_{i,k}(x_{jk}) f^p(x_{jk}) \mathrm{d}x_{jk} \quad . \qquad (4) \end{aligned}$$

The solution of the integrals depends on the type of the density. Hence, we will give solvable representations for continuous, discrete, and hybrid joint densities in the next subsection.

We first take a look at purely continuous and purely discrete systems, followed by hybrid conditional densities, which leads directly to hybrid joints.

4.1 Gaussian Mixtures

We represent marginal densities over continuous variables by means of Gaussian mixtures. With this approach the approximation of the conditional density

$$f(x \,|\, u_1, \ldots, u_d) = \sum_{i=1}^{M} \gamma_i N_i(x, \mu_i, \sigma_i) \prod_{j=1}^{d} N_{i,j}(u_j, \mu_{ij}, \sigma_{ij})$$

is a mixture of axis-aligned Gaussian components. Since, this type of density representation is already covered in [1], we only give the resulting update densities for completeness. For keeping the formulae compact, we omit the parameters of the Gaussians if possible and have

$$\begin{aligned} f^p(x) &= \sum_{i=1}^{M} \gamma_i N_i(x) \prod_{j=1}^{d} \int_{\mathbb{R}} N_{i,j}(u_j) N(u_j)\backslash_x \mathrm{d}u_j \\ &= \sum_{i=1}^{M} \gamma_i N_i(x) \prod_{j=1}^{d} \underbrace{N_{i,j}^{u_j}\left(\mu_{ij}, \mu_{j\backslash_x}, \sqrt{\sigma_{ij}^2 + \sigma_{j\backslash_x}^2}\right)}_{c_{u_j}} \\ &= \sum_{i=1}^{M} \alpha_i N_i(x) \quad , \end{aligned}$$

where

$$\alpha_i = \gamma_i \prod_{j=1}^{d} c_{u_j}$$

and

$$f_j^e(x) = \sum_{i=1}^{M} \gamma_i N_i(x) \underbrace{\int_{\mathbb{R}} N_i(y_j) N^e(y_j) \mathrm{d}y_j}_{c_{y_j}} \cdot \prod_{k=1}^{d} \underbrace{\int_{\mathbb{R}} N_{i,k}(x_{jk}) N^p(x_{jk}) \mathrm{d}x_{jk}}_{c_{x_{jk}}} = \sum_{i=1}^{M} \alpha_i N_i(x)$$

where

$$\alpha_i = \gamma_i c_{y_j} \prod_{j=1}^{d} c_{x_{jk}} \ .$$

For reasons of brevity, we assumed here that $f^p(u_j) = N(u_j)$ is a single Gaussian component. In the general case $f^p(u_j)$ is a mixture of Gaussians. This is no problem, since the Integral and the sum of this mixture can be interchanged. The same holds for $f^e(y_j)$.

4.2 Dirac Mixtures

In the case of discrete random variables we use Dirac mixtures as density representation. This is a valid mapping of discrete events to a continuous scale.

One may argue, that the Dirac representation is not valid, since the product of two Dirac pulses on the same axis is not defined, but multiplication is a crucial operation for fusing densities.

We will now show that this is not necessarily correct, since we use Dirac series to represent the discrete Density. If we further calculate the normalized product of two mixtures, we can show that this product converges to the same solution that the standard vector approach yields.

Product of Dirac Mixtures

As an example we consider a binary random variable $\mathbf{x}$ and two density functions on this variable represented by the Dirac mixtures

$$f_1(x) = \alpha_1 \delta(x-1) + \beta_1 \delta(x-2)$$

and

$$f_2(x) = \alpha_2 \delta(x-1) + \beta_2 \delta(x-2) \ .$$

The most important part of the proof is, that we have to calculate the normalized product of these mixtures given by

$$f(x) = \frac{f_1(x) f_2(x)}{\int\limits_{\mathbb{R}} f_1(\xi) f_2(\xi) \mathrm{d}\xi} \ .$$

In our example this is

$$f(x) = \frac{[\alpha_1\delta(x-1) + \beta_1\delta(x-2)]\,[\alpha_2\delta(x-1) + \beta_2\delta(x-2)]}{\int\limits_{\mathbb{R}} [\alpha_1\delta(\xi-1) + \beta_1\delta(\xi-2)][\alpha_2\delta(\xi-1) + \beta_2\delta(\xi-2)]\mathrm{d}\xi}$$

We now substitute the Dirac delta functions by their definition as limit

$$\delta(x) = \lim_{\epsilon \to 0} d(x, \epsilon) \ .$$

where $d(x,\epsilon)$ is for example

$$d(x,\epsilon) = \frac{1}{\sqrt{2\pi}\epsilon} \exp\left\{ -\frac{1}{2} \frac{x^2}{\epsilon^2} \right\} \ .$$

It can then be shown, that

$$\lim_{\epsilon_1 \to 0, \epsilon_2 \to 0} \frac{f_1(x) f_2(x)}{\int_{\mathbb{R}} f_1(\xi) f_2(\xi) \mathrm{d}\xi} = \frac{\alpha_1\alpha_2}{\alpha_1\alpha_2 + \beta_1\beta_2} \delta(x-1) + \frac{\beta_1\beta_2}{\alpha_1\alpha_2 + \beta_1\beta_2} \delta(x-2) \ .$$

This is the same result we would expect from the standard matrix vector approach.

Update Rules for Discrete Variables

We now derive the update rules for the discrete case with Dirac mixture representation. For this purpose we substitute the generic density functions in (3) by Dirac mixture representations.

$$\begin{aligned} f^p(x) &= \sum_{i=1}^{M} \gamma_i f_i(x) \prod_{j=1}^{d} \int_{\mathbb{R}} f_{i,j}(u_j) f(u_j) \backslash_x \mathrm{d}u_j \\ &= \sum_{i=1}^{M} \gamma_i \left[\sum_{k=1}^{|\boldsymbol{x}|} \alpha_{ik} \delta(x-k) \right] \\ &\cdot \prod_{j=1}^{d} \int_{\mathbb{R}} \left[\sum_{l=1}^{|\boldsymbol{u_j}|} \alpha_{jl} \delta(u_j - l) \right] \left[\sum_{l=1}^{|\boldsymbol{u_j}|} \beta_{jl} \delta(u_j - l) \right] \mathrm{d}u_j \end{aligned} \tag{5}$$

$|\boldsymbol{x}|$ means the number of states of a discrete random variable $\boldsymbol{x}$. The normalized product of the integrand in (5) converges to

$$\sum_{l=1}^{|\boldsymbol{u_j}|} \frac{\alpha_{lj}\beta_{lj}}{\sum_{l=1}^{|\boldsymbol{u_j}|} \alpha_{lj}\beta_{lj}} \delta(u_j - l) \ ,$$

as shown in the proof above. Since this is the normalized product, we can say that the integral in (5) is the reciprocal of the normalizing factor, hence,

$$\int_{\mathbb{R}} \left[\sum_{l=1}^{|\boldsymbol{u_j}|} \alpha_{lj} \delta(u_j - l) \right] \left[\sum_{l=1}^{|\boldsymbol{u_j}|} \beta_{lj} \delta(u_j - l) \right] \mathrm{d}u_j = \sum_{l=1}^{|\boldsymbol{u_j}|} \alpha_{lj}\beta_{lj} \ .$$

This is a constant and we end up with

$$f^p(x) = \sum_{i=1}^{M} \alpha_i \left[\sum_{k=1}^{|\boldsymbol{x}|} \alpha_{ik} \delta(x-k) \right]$$

where

$$\alpha_i = \gamma_i \prod_{j=1}^{d} \sum_{l=1}^{|\boldsymbol{u_j}|} \alpha_{lj}\beta_{lj} \ .$$

With the same argument we substitute the generic densities in (4) and end up with

$$
\begin{aligned}
&f_j^e(x) \\
&= \sum_{i=1}^{M} \gamma_i f_i(x) \int_{\mathbb{R}} f_i(y_j) f^e(y_j) \mathrm{d}y_j \\
&\cdot \prod_{k=1}^{d} \int_{\mathbb{R}} f_{i,k}(x_{jk}) f^p(x_{jk}) \mathrm{d}x_{jk} \\
&= \sum_{i=1}^{M} \gamma_i \left[\sum_{l=1}^{|\boldsymbol{x}|} \alpha_{il} \delta(x-l) \right] \\
&\cdot \int_{\mathbb{R}} \left[\sum_{l=1}^{|\boldsymbol{y_j}|} \alpha_{il} \delta(y_j - l) \right] \left[\sum_{l=1}^{|\boldsymbol{y_j}|} \beta_{il} \delta(y_j - l) \right] \mathrm{d}y_j \\
&\cdot \prod_{k=1}^{d} \int_{\mathbb{R}} \left[\sum_{l=1}^{|\boldsymbol{x_{jk}}|} \alpha_{kl} \delta(x_{jk} - l) \right] \left[\sum_{l=1}^{|\boldsymbol{x_{jk}}|} \beta_{kl} \delta(x_{jk} - l) \right] \mathrm{d}x_{jk} \\
&= \sum_{i=1}^{M} \alpha_i \left[\sum_{l=1}^{|\boldsymbol{x}|} \alpha_{il} \delta(x-l) \right] \quad .
\end{aligned}
$$

where

$$
\alpha_i = \gamma_i \left(\sum_{l=1}^{|\boldsymbol{y_j}|} \alpha_{il} \beta_{il} \right) \prod_{k=1}^{d} \sum_{l=1}^{|\boldsymbol{x_{jk}}|} \alpha_{kl} \beta_{kl} \quad .
$$

Now, the normalized product can be used to fuse $f^p(x)$ and $f^e(x)$.

4.3 Hybrid Conditional Densities

In [10] we introduced a representation for approximated hybrid conditional densities. In the next section we will give a slightly reformulated version that is more intuitive.
For this purpose we first consider some special cases which will later be wrapped up to the general case. As an example, we will take a look at two dependent random variables $\boldsymbol{x}$ and $\boldsymbol{y}$ and their conditional density $f(y \,|\, x)$ that will be approximated by

$$
f(y \,|\, x) \approx \sum_{i=1}^{M} \gamma_i f_i(x) f_i(y) \quad .
$$

First we assume $\boldsymbol{x}$ and $\boldsymbol{y}$ to be discrete. Hence, we have

$$
f(y \,|\, x) \approx \sum_{i=1}^{M} \gamma_i \left[\sum_{j=1}^{|\boldsymbol{x}|} \alpha_{ij} \delta(x-j) \right] \left[\sum_{j=1}^{|\boldsymbol{y}|} \beta_{ij} \delta(y-j) \right] \quad .
$$

To explain the application of this approach we give the following example. Assume $\boldsymbol{x}$ and $\boldsymbol{y}$ to be binary and the CPT for $f(y \,|\, x)$ to be

	$\boldsymbol{y}=1$	$\boldsymbol{y}=2$
$\boldsymbol{x}=1$	0.3	0.7
$\boldsymbol{x}=2$	0.8	0.2

.

In our Dirac mixture approach we use the following parameters. We only need two components in the mixture. Hence, we set $M=2$. The parameters for the x and y part are

$$
\begin{array}{llll}
\alpha_{11} = 1 & \alpha_{12} = 0 & \beta_{11} = 0.3 & \beta_{12} = 0.7 \\
\alpha_{21} = 0 & \alpha_{22} = 1 & \beta_{21} = 0.8 & \beta_{22} = 0.2
\end{array}
$$

The γ parameters can be set to 1 since this is a conditional density that does not have to be normalized.
Next we consider the case that $\boldsymbol{x}$ is discrete and $\boldsymbol{y}$ is continuous, which leads to

$$
f(y \,|\, x) \approx \sum_{i=1}^{M} \gamma_i \left[\sum_{j=1}^{|\boldsymbol{x}|} \alpha_{ij} \delta(x-j) \right] N(y, \mu_i, \sigma_i) \quad .
$$

In this case we have

$$
M \bmod |\boldsymbol{x}| = 0 \quad ,
$$

because we have to consider all the possible states of $\boldsymbol{x}$. This means, for a binary $\boldsymbol{x}$ we have

$$
\begin{array}{cc}
\alpha_{11} = 1 & \alpha_{12} = 0 \\
\alpha_{21} = 0 & \alpha_{22} = 1 \\
\vdots & \vdots \\
\alpha_{M-1,1} = 1 & \alpha_{M-1,2} = 0 \\
\alpha_{M1} = 0 & \alpha_{M2} = 1
\end{array}
$$

The parameters for the Gaussian parts of the components have to be set accordingly.
Now we have the case that $\boldsymbol{x}$ is continuous and $\boldsymbol{y}$ is discrete, which yields

$$
f(y \,|\, x) \approx \sum_{i=1}^{M} \gamma_i N(x, \mu_i, \sigma_i) \left[\sum_{j=1}^{|\boldsymbol{y}|} \alpha_{ij} \delta(y-j) \right] \quad .
$$

For a continuous $\boldsymbol{x}$ it is extremely important to select the number of Gaussian components appropriately. This is due to the fact, that $f(y \,|\, x)$ in general is no density in x. Since we approximate this axis by means of Gaussian densities, we have to select an interval of relevance for x where we place the components. Without this restriction, an infinite number Gaussian components on this axis would be required.
Finally, we wrap up all the cases seen so far to a generic case. We consider $\boldsymbol{y}$ to depend on a set of continuous *and* discrete variables $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$. The type of $\boldsymbol{y}$ is left arbitrary, so we can write

$$
\begin{aligned}
&f(y|x_1, \ldots, x_m) \approx \\
&\sum_{i=1}^{M} \gamma_i f_i(y) \left(\prod_{k=1}^{n} \sum_{j=1}^{|\boldsymbol{x_k}|} \alpha_{ij} \delta(x_k - j) \right) \prod_{k=n+1}^{m} N(x_k, \mu_i, \sigma_i) \quad .
\end{aligned}
$$

$f_i(y)$ can then be substituted by a Gaussian or a Dirac mixture respectively.

5 Hybrid Joint Densities

After having defined hybrid conditional densities, hybrid joint densities are a logical consequence. As already pointed out in the introduction, we have to consider joint

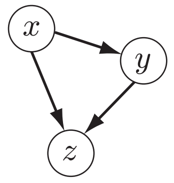

Figure 5: A cluster consisting of three variables.

densities, whenever clustering was applied to the original network.
We start with a short example of a cluster to explain how the joint is composed and how to represent this. The example in Figure 5 depicts a cluster of three variables $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$. The generic joint density in this example is

$$f(x,y,z) = f(x)f(y\,|\,x)f(z\,|\,x,y) \ .$$

Since we use mixture densities with axis aligned components for prior as well as for conditional densities we write the above as

$$f(x,y,z) = \\ \sum_{i=1}^{L}\alpha_i f_i(x)\cdot\sum_{j=1}^{M}\beta_j f_j(x)f_j(y)\cdot\sum_{k=1}^{N}\gamma_k f_k(x)f_k(y)f_k(z) \ .$$

This can be subsumed into one sum

$$f(x,y,z) = \sum_{i=1}^{L\cdot M\cdot N}\alpha'_i f_i(x)fi(y)f_i(z) \ .$$

From now on we will express these kind joint densities in the vector form

$$f(\underline{x}) = \sum_{i=1}^{M}\alpha_i f_i(x_1)\dots f_i(x_n) \ ,$$

where

$$\underline{x} = [x_1,\dots,x_n]^T$$

and

$$f_i(x_j)\neq f_i(x_k) \ \text{ for all } \ x_j\neq x_k \ .$$

We now give the idea of the update rules for systems with clusters. First of all we write the generic update rules (1) and (2) in vector form

$$f^p(\underline{x}) = \int_{\mathbb{R}^d} f(\underline{x}\,|\,\underline{u}_1,\dots,\underline{u}_d)\prod_{i=1}^{d} f^p(\underline{u}_i)\mathrm{d}\underline{u}_1\,\dots\,\mathrm{d}\underline{u}_d$$

and

$$\begin{aligned} f_j^e(\underline{x}) \ &= \ \int_{\mathbb{R}^{d+1}} f(\underline{y}_j\,|\,\underline{x},\underline{x}_{j1},\dots,\underline{x}_{jd})f^e(y_j) \\ &\quad\cdot\prod_{k=1}^{d} f^p(\underline{x}_{jk})\mathrm{d}\underline{y}_j\,\mathrm{d}\underline{x}_{j1}\,\dots\,\mathrm{d}\underline{x}_{jd} \ . \end{aligned}$$

The derivations are now similar to the considerations made above for purely continuous and discrete densities, besides the conditional densities used here. In the conditional densities for clusters

$$f(\underline{x}\,|\,\underline{u}_1,\dots,\underline{u}_d)$$

and

$$f(\underline{y}_j\,|\,\underline{x},\underline{x}_{j1},\dots,\underline{x}_{jd})$$

respectively, we have a set of variables depending on sets of variables. The sets $\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{y}}_j$ however, may contain variables, that did not depend on any of these $\underline{\boldsymbol{u}}_i$ or $\underline{\boldsymbol{x}}_{ji}$ in the original system, respectively. Hence, we have to consider the conditional densities of the original system for all members of the sets $\underline{\boldsymbol{x}}$ and $\underline{\boldsymbol{y}}_j$.
Comparing the update rules for systems with or without clusters, we can say, that due to the application of mixture densities with axis aligned components, everything boils down to multiplication or fusion of density functions. In the case of clusters, however, we have to fuse joint densities with nonempty intersections as pointed out in the problem formulation.

5.1 Fusion of Hybrid Joint Mixture Densities

We now derive the fusion of two hybrid joint mixture densities that overlap in some variables, but not in all. To guarantee the convergence of the discrete parts of the product (see 4.2), we derive the normalized product.
The normalized product of two arbitrary joint mixture densities is

$$\frac{\left(\sum\limits_{k_1=1}^{L}\alpha_{k_1}\prod\limits_{i=1}^{n} f_{k_1}(x_i)\right)\left(\sum\limits_{k_2=1}^{M}\beta_{k_2}\prod\limits_{j=l}^{m} f_{k_2}(x_j)\right)}{\int\limits_{\mathbb{R}^m}\left(\sum\limits_{k_1=1}^{L}\alpha_{k_1}\prod\limits_{i=1}^{n} f_{k_1}(\xi_i)\right)\left(\sum\limits_{k_2=1}^{M}\beta_{k_2}\prod\limits_{j=1}^{m} f_{k_2}(\xi_j)\right)\mathrm{d}\underline{\xi}} \ ,$$

where $1 < l < n < m$ and the overlapping variables are $x_l\dots x_n$. Since all functions $f(\,\cdot\,)$ used here are normalized densities, the denominator can be simplified by marginalization to

$$\int\limits_{\mathbb{R}^{n-l+1}}\left(\sum_{k_1=1}^{L}\alpha_{k_1}\prod_{i=l}^{n} f_{k_1}(\xi_i)\right)\left(\sum_{k_2=1}^{M}\beta_{k_2}\prod_{j=l}^{n} f_{k_2}(\xi_j)\right)\mathrm{d}\underline{\xi} \ .$$

Note that both products run from l to n, since all non-overlapping variables where marginalized to 1.
The numerator can be rearranged into one sum

$$\sum_{k=1}^{L\cdot M}\alpha'_k\prod_{i=1}^{l-1} f_k(x_i)\prod_{i=l}^{n} f_{k_1}(x_i)f_{k_2}(x_i)\prod_{i=n+1}^{m} f_k(x_i) \ .$$

To calculate the product of the densities for overlapping variables, we can use the approaches for Gaussians and Dirac Mixtures presented above. The normalization can then be combined with the weight α, since it is a constant. Finally, we end up with our well known mixture representation

$$\sum_{k=1}^{L\cdot M}\alpha'_k\prod_{i=1}^{m} f_k(x_i) \ ,$$

over all involved variables.

Complexity

One may argue, that all these products of mixture sums lead to a very high number of components in the resulting mixture. With the presented approach however, we have some

features that keep the number of components on a tractable level.
First, the component growth is only true for the Gaussian components. For the Dirac mixtures this not true, since we always come back to components in the number of discrete states. This can also be seen directly from the product of discrete mixtures in section 4.2.
Second, we would like to point out, that the application of axis-aligned components in the conditional densities decreases the number of components in the forward step at the handover point from one subsystem to another, which can be seen in section 4. Due to this feature, the number of components does not accumulate from the input to the output of large cascaded systems.

6 Conclusions

In this article we presented a unified approach to nonlinear hybrid Bayesian networks. The goal of this approach is to unify the representation of continuous and discrete variables in the network in order to apply the same update mechanism for both cases.
To achieve this goal, we examined Bayesian networks from a system theoretic viewpoint. Our derivations show that a Bayesian network can be treated as a system of cascaded subsystems. Hence, we have to calculate just the densities from the output and the input of two connected systems and end up with a recursive updating scheme.
We further presented a method for modeling continuous, discrete, and hybrid joint densities by means of Gaussian and Dirac mixtures. Our approach makes no restrictions concerning discrete variables depending on continuous variables. In the same manner we can treat nodes containing joint densities that result from clusters to remove cycles in a network.
We did not address the problem of finding the model parameters for conditional densities. In the case of known dependencies and known uncertainty characteristics, however, the modeling of conditional densities is straightforward for the presented approach. For a given subsystem $y = h(x) + v$ with additive noise v for example, we only have to place axis-aligned Gaussian components along the curvature of $y = h(x)$. The number of components then governs the approximation quality. In the case of only given samples from the subsystem, the optimal approximation is still subject to research.
Our approach allows for exact evaluation. Hence, no convergence considerations as in sample based evaluation methods need to be made. Due to the preapproximation of conditional densities by means of mixture densities with axis-aligned components, the complexity in the number of components is kept at a constant level.

References

[1] E. Driver and D. Morrell. Implementation of Continuous Bayesian Networks Using Sums of Weighted Gaussians. In Besnard and Hanks, editors, *Proceedings of the 11^{th} Conference on Uncertainty in Artificial Intelligence*, pages 134–140, Montreal, Quebec, Canada, 1995.

[2] Michael I. Jordan. Graphical models. *Statistical Science*, 19(1):140–155, 2004.

[3] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, February 2001.

[4] Steffen L. Lauritzen and Nanny Wermuth. Graphical Models for Associations between Variables, some of which are Qualitative and some Quantitative. *The Annals of Statistics*, 17(1):31–57, March 1989.

[5] Uri Lerner, Eran Segal, and Daphne Koller. Exact Inference in Networks with Discrete Children of Continuous Parents. In *Proceedings of the 17^{th} Conference on Uncertainty in Artificial Intelligence*, pages 319–238, Seattle, Washington, USA, 2001.

[6] H.-A. Loeliger. An introduction to factor graphs. *Signal Processing Magazine, IEEE*, 21(1):28–41, 2004.

[7] Samuel McLaughlin, Vikram Krishnamurthy, and Robin J. Evans. Bayesian network model for data incest in a distributed sensor network. In *Proceedings of the 7^{th} International Conference on Information Fusion*, volume I, pages 606–613, Stockholm, Sweden, June 2004.

[8] Kevin Patrick Murphy. A Variational Approximation for Bayesian Networks with Discrete and Continuous Latent Variables. In K. Blackmond-Laskey and H. Prade, editors, *Proceedings of the 15^{th} Conference on Uncertainty in Artificial Intelligence*, pages 457–466, Stockholm, Sweden, 1999.

[9] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[10] O. C. Schrempf and U. D. Hanebeck. A New Approach for Hybrid Bayesian Networks Using Full Densities. In *Proceedings of 6^{th} Workshop on Computer Science and Information Technologies, CSIT 2004*, Budapest, Hungary, 2004.

[11] O. C. Schrempf and U. D. Hanebeck. Evaluation of Hybrid Bayesian Networks using Analytical Density Representations. In *Proceedings of the 16^{th} IFAC World Congress, IFAC 2005*, Prague, Czech Republic, 2005.

[12] Charles Sutton, Clayton Morrison, Paul R. Cohen, Joshua Moody, and Jafar Adibi. A Bayesian blackboard for information fusion. In *Proceedings of the 7^{th} International Conference on Information Fusion*, volume II, pages 1111–1116, Stockholm, Sweden, June 2004.

[13] Yongmian Zhang, Qiang Ji, and Carl Looney. Active Information Fusion for Decision Making Under Uncertainty. In *Proceedings of the 5^{th} International Conference on Information fusion, Anapolis, Maryland*, July 2002.