

Optimal Mixture Approximation of the Product of Mixtures

Oliver C. Schrempf, Olga Feiermann, and Uwe D. Hanebeck
Intelligent Sensor-Actuator-Systems Laboratory
Institute of Computer Science and Engineering
Universität Karlsruhe (TH)
Karlsruhe, Germany
Email: {schrempf|feiermann}@ira.uka.de, Uwe.Hanebeck@ieee.org

Abstract—Gaussian mixture densities are a very common tool for describing arbitrarily structured uncertainties in various applications. Many of these applications have to deal with the fusion of uncertainties, an operation that is usually performed by multiplication of these densities. The product of Gaussian mixtures can be calculated exactly, but the number of mixture components in the resulting mixture increases exponentially. Hence, it is essential to approximate the resulting mixture with less components, to keep it tractable for further processing steps. This paper introduces an approach for approximating the exact product with a mixture that uses less components. The maximum approximation error can be chosen by the user. This choice allows to trade accuracy of the approximation for the number of mixture components used. This is possible due to the usage of a progressive processing scheme that calculates the product operation by means of a system of ordinary differential equations. The solution of this system yields the parameters of the desired Gaussian mixture.

I. INTRODUCTION

Many processing schemes in stochastic applications using Bayesian techniques like estimation and filtering have to deal with multiplication of probability density functions. Since arbitrary densities are often approximated by Gaussian mixtures, the need for a mixture representation of the product of mixtures arises. Calculating the product of two Gaussian mixtures of, say N and M components, yields a mixture with $N \cdot M$ components. It is obvious, that successive multiplication as used in recursive processing schemes increases the number of components to an intractable size. Hence, a procedure is required that keeps the number of mixture components after each multiplication step constant or at least manageable.

There are two main approaches known in literature today for keeping Gaussian mixtures tractable. The first approach is reestimation of the product density function's parameters based on samples taken from the exact product density function. The goal is to reestimate the function with less components. Most famous in this category are Parzen windows [1], the Expectation-Maximization Algorithm (EM) [2], or advanced methods for learning the parameters of Gaussian mixtures from samples as applied in [3]. A method especially designed for the approximation of the product of Gaussian mixtures based on sampling is given in [4]. It is known, that reestimation of Gaussian mixtures is an optimization problem with many local minima, making it tough for sample based approaches. Furthermore, the use in online processing applications forbids long search times or large sample sets.

The second approach for keeping the number of components of Gaussian mixtures tractable is to take the exact mixture resulting from the product and reduce the number of its components. These mixture reduction techniques are well known in the field of target tracking in random clutter. The exact Bayesian solution to this tracking problem yields Gaussian mixture densities growing exponentially in the number of components [5]. Many approaches to mixture reduction have been studied, like Nearest Neighbor (NN) approximation [5], [6] or Joint Probabilistic Data Association (JPDA) [5], [7]. NN reduces the mixtures associated with one target to its largest component, whereas JPDA merges the components of a mixture belonging to a target into one single Gaussian component. It is obvious, that these techniques result in an intolerable loss of information for the generic multiplication of Gaussian Mixtures. A more flexible approach is the mixture reduction algorithm of Salmond as reviewed in [8]. It successively merges neighboring components using update rules, which preserve the mean and covariance of the merged pairs.

No matter what approach is used to keep the density tractable, the resulting mixture contains less components and is therefore an approximation of the exact product density function. Hence, it must be ensured, that the resulting mixture is as close to the original mixture as possible. In contrast to the well known techniques mentioned above, we present an approach that approximates the mixture of the product density $f_p = f_1 \cdot f_2$ based on the parameters of the operand mixtures f_1 and f_2 while minimizing a distance measure between f_p and its approximation.

Our solution to the problem uses a progressive approximation scheme called Progressive Bayes [9]. In this approach, the algorithm starts with one factor of the product and converges to the desired product. This is achieved by introducing a progression parameter and transforming the problem into a system of ordinary differential equations which can be solved for example using Euler or Runge-Kutta methods [10]. During the process of solving, the algorithm is adding/removing components to keep the distance to the exact density at a minimum. This approach ensures that a predefined error threshold is not exceeded while keeping the number of used components small. In practice, significantly less than $N \cdot M$ components are required to produce approximations close to the original density.

The remainder of this paper is structured as follows. We first give a mathematical formulation of the problem solved in this paper. In section III we show how to apply the Progressive Bayes framework to this problem. This is followed by some examples in section IV. Concluding remarks are made in section V.

II. PROBLEM FORMULATION

We consider the product of two parametric density functions which are given as Gaussian mixtures

$$f_1(x) = \sum_{i=1}^M w_{i,1} \frac{1}{\sqrt{2\pi}\sigma_{i,1}} \exp\left\{-\frac{1}{2} \frac{(x - \mu_{i,1})^2}{\sigma_{i,1}^2}\right\}$$

and

$$f_2(x) = \sum_{j=1}^N w_{j,2} \frac{1}{\sqrt{2\pi}\sigma_{j,2}} \exp\left\{-\frac{1}{2} \frac{(x - \mu_{j,2})^2}{\sigma_{j,2}^2}\right\},$$

where f_1 consists of M and f_2 consists of N components. $w_{i,1}, w_{j,2}$ are the weights, $\mu_{i,1}, \mu_{j,2}$ the mean values, and $\sigma_{i,1}^2, \sigma_{j,2}^2$ the variances of the i 'th and j 'th component of f_1 and f_2 respectively.

It is possible to describe the product $f_p = f_1 \cdot f_2$ by means of a Gaussian mixture as well, since the components of f_1 and f_2 are Gaussian densities, which can be multiplied componentwise. The product of two Gaussian densities is again a Gaussian density with updated parameters μ and σ . This leads to f_p having $M \cdot N$ components, where the parameters of each component depend on the parameters of one component in f_1 and of one component in f_2 .

The Gaussian parameters of each component in f_p can be calculated, using the well known Kalman filter rules

$$\sigma^2 = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2}$$

and

$$\mu = \frac{\mu_1 \sigma_2^2 + \mu_2 \sigma_1^2}{\sigma_1^2 + \sigma_2^2}.$$

The weight factors of each component of the Gaussian mixtures can be updated according to

$$w = \frac{w_1 w_2}{\sqrt{2\pi(\sigma_1^2 + \sigma_2^2)}} \cdot \exp\left\{-\frac{1}{2} \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}\right\}.$$

It is clear that this growth in the number of components is a problem for recursive processing which is heavily based on successive multiplication of densities like

$$\begin{aligned} f_2 &= f_1 \cdot g_1 \\ f_3 &= f_2 \cdot g_2 \\ f_4 &= f_3 \cdot g_3 \\ &\vdots \end{aligned}$$

Every time a new mixture g_i is multiplied with the resulting mixture from the preceding step, the number of components is increased by the factor of the number of components in g_i . In the worst case this leads to non tractable large mixtures.

Inspecting the parameters of products of Gaussian mixtures shows, that in many cases there are weights of negligible size. It also happens, that variances of single components become very large in comparison to the other variances, hence, the component becomes very flat and has no influence on the resulting mixture. This indicates, that the resulting density can be approximated by a Gaussian mixture with less components.

The goal is to find an approximation $f(x)$ of the product of two Gaussian mixtures that is close to the exact mixture $f_p(x)$. The deviation between $f(x)$ and $f_p(x)$ has to be quantified by a distance measure $G(f, f_p)$.

It must be possible for the user to define a maximum tolerance threshold for the deviation error. The algorithm has to find an approximation with less components than in f_p with an error smaller or equal to the error accepted by the user.

III. PROGRESSIVE PROCESSING

Our approach makes use of the Progressive Bayes framework introduced in [9], which was originally intended for non-linear estimation purposes like filtering and prediction. Since filtering and prediction are recursive processing approaches, which are based on multiplication of densities, it is possible to use the framework also for the basic multiplication of Gaussian mixtures.

In this section we will first give a short overview of the key idea behind the Progressive Bayes framework and how it can be applied to our problem. This is followed by a derivation, that is an alternative to the one published in [9].

Throughout this paper we will consider just the product of scalar Gaussian mixtures for brevity. None the less, formulae for the vector valued case are available.

A. Key Idea

In the first step, we introduce a new parameter γ into the term of the exact solution $f_p(x)$. γ affects f_p in the way, that for $\gamma = 0$ we have a function that can be approximated very easily and for $\gamma = 1$ we have the exact function f_p . How this is achieved will be shown in section III-B. The parameterized function will be named $\tilde{f}(x, \gamma)$.

Next we define a measure of deviation $G(\tilde{f}(x, \gamma), f(x, \underline{\eta}))$ between the parameterized exact density $\tilde{f}(x, \gamma)$ and the approximation density $f(x, \underline{\eta})$, where $\underline{\eta}$ is the parameter vector of the approximation density. As we are using Gaussian mixtures we have

$$\underline{\eta} = [\underline{\eta}_1, \underline{\eta}_2, \dots, \underline{\eta}_L]^T$$

for L components, with

$$\underline{\eta}_i = [w_i, \mu_i, \sigma_i^2]^T,$$

where w_i is the weight, μ_i is the mean and σ_i^2 is the variance of the i 'th component.

We will choose the parameterization in $\tilde{f}(x, \gamma)$ in such a way that for $\gamma = 0$ we have $G(\tilde{f}(x, \gamma), f(x, \underline{\eta})) = 0$. This means for $\gamma = 0$ we know the exact approximation with no deviation from $\tilde{f}(x, \gamma)$.

We then let γ approach 1 in a progressive way, while adjusting the parameters $\underline{\eta}$ of the approximation density to

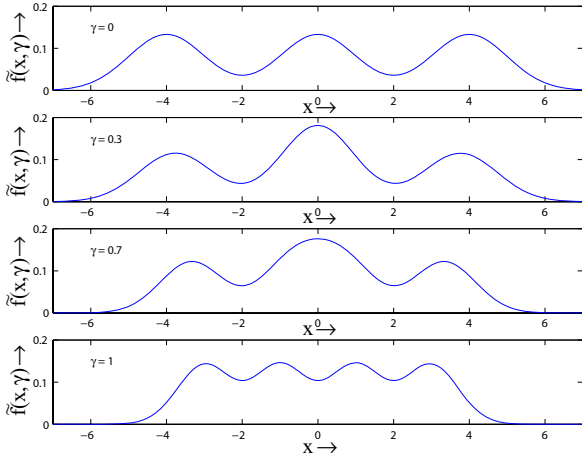


Fig. 1. The progression of $\tilde{f}(x, \gamma)$ goes from $f_2(x)$ to the product of $f_1(x)$ and $f_2(x)$ from figure 2.

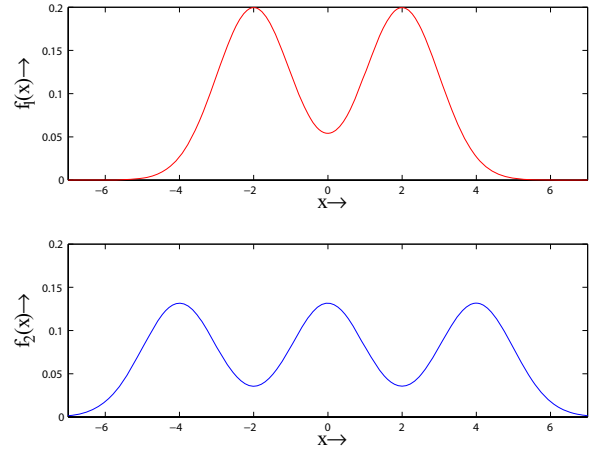


Fig. 2. Top: Gaussian mixture with two components. Bottom: Gaussian mixture with three components.

keep G at a minimum. Since for $\gamma = 1$ we have $\tilde{f}(x, \gamma) = f_p(x)$ and G remains at a minimum, we can guarantee that $f(x)$ is as close to $f_p(x)$ as possible.

If $G(\tilde{f}(x, \gamma), f(x, \underline{\eta}))$ becomes larger than the user defined threshold, structural adaptation in $f(x, \underline{\eta})$ by means of adding further components is applied.

B. Parameterized True Density

We now show how to introduce the progression parameter γ into $f_p(x)$ in order to fulfill the constraints from section III-A. For the remainder of this paper we assume normalized weights for f_1 and f_2 , which means $\sum_{i=1}^M w_i = 1$ and $\sum_{j=1}^N w_j = 1$ respectively.

For $\gamma = 0$ it is essential, that $\tilde{f}(x, \gamma)$ is approximated with no error. Since $f_p(x)$ is the product of two Gaussian mixtures $f_1(x)$ and $f_2(x)$, we choose the parameterization in such a way, that for $\gamma = 0$ we have $\tilde{f}(x, \gamma) = f_2(x)$. So we can set $f(x, \underline{\eta}) = f_2(x)$ as well and $G(f(x, \gamma), f(x, \underline{\eta})) = G(f_2(x), f_2(x)) = 0$. The second constraint is, that for $\gamma = 1$ we have $\tilde{f}(x, \gamma) = f_p(x)$.

To accomplish these two constraints, γ has to be inserted only into the f_1 -part of f_p , hence, we define

$$\begin{aligned} \tilde{f}(x, \gamma) &= \tilde{f}_1(x, \gamma) f_2(x) \\ &= \sum_{i=1}^M w_{i,1} \exp \left\{ -\frac{1}{2} \frac{(x - \mu_{i,1})^2}{(\sigma_{i,1} \frac{1+\epsilon}{\gamma+\epsilon})^2} \right\} \\ &\quad \cdot \sum_{j=1}^N w_{j,2} \frac{1}{\sqrt{2\pi}\sigma_{j,2}} \exp \left\{ -\frac{1}{2} \frac{(x - \mu_{j,2})^2}{\sigma_{j,2}^2} \right\}, \end{aligned}$$

where ϵ is a small constant and $\gamma \in [0 \dots 1]$. For $\gamma = 0$ the f_1 part of the density is equal to 1, since the exponent is 0. Hence, $\tilde{f}(x, \gamma) = f_2$ for $\gamma = 0$. We omitted $\frac{1}{\sqrt{2\pi}\sigma_{i,1}}$ in the f_1 -part since it is only a normalizing factor.

We will start our approximation with the parameters from f_2 since this guarantees that $G = 0$ as shown above. For $\gamma = 1$ the parameterization factor $\frac{1+\epsilon}{\gamma+\epsilon}$ becomes 1 and “vanishes”. $\tilde{f}(x, \gamma) = f_p$ for $\gamma = 1$ and the exact density is reached.

Figure 1 shows the progression for $\tilde{f}(x, \gamma)$ in an example where f_1 has three components and f_2 has two components as depicted in figure 2. Please note that the first plot in figure 1 ($\gamma = 0$) is the same as $f_2(x)$ and the fourth plot shows the product of $f_1(x)$ and $f_2(x)$.

Please note further, that the exact product has six components, but it is easy to see, that it can be approximated with four components yielding just a very small error.

C. Parametric Adaptation

To execute the progression of γ from 0 to 1, while keeping a distance measure at its minimum, we transform the problem into a system of ordinary first-order differential equations, which we solve for γ in the interval $[0 \dots 1]$. Therefore we first have to define the distance measure between $\tilde{f}(x, \gamma)$ and its approximation $f(x, \underline{\eta})$. We choose a squared integral distance measure

$$G = \frac{1}{2} \int_{\mathbb{R}} \left(\tilde{f}(x, \gamma) - f(x, \underline{\eta}) \right)^2 dx.$$

The original paper [9] replaced $f(x, \underline{\eta})$ by a Taylor series expansion. We omit this replacement here to present an alternative derivation.

$G(\tilde{f}(x, \gamma), f(x, \underline{\eta}))$ is a function over γ and $\underline{\eta}$. Hence, the constraint for the minimum requires the derivative with respect to $\underline{\eta}$ and γ to be zero.

By taking the partial derivative of the distance measure G with respect to the parameter $\underline{\eta}$

$$\frac{\partial G}{\partial \underline{\eta}} = - \int_{\mathbb{R}} \left(\tilde{f}(x, \gamma) - f(x, \underline{\eta}) \right) \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} dx,$$

and setting the result to zero, we obtain

$$\int_{\mathbb{R}} \tilde{f}(x, \gamma) \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} dx = \int_{\mathbb{R}} f(x, \underline{\eta}) \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} dx.$$

The partial derivative with respect to γ is given by

$$\begin{aligned} & \int_{\mathbf{R}} \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} \frac{\partial \underline{\eta}}{\partial \gamma} \left(\frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} \right)^T dx \\ & + \int_{\mathbf{R}} f(x, \underline{\eta}) \frac{\partial^2 f(x, \underline{\eta})}{\partial \underline{\eta} \partial \underline{\eta}^T} \frac{\partial \underline{\eta}}{\partial \gamma} dx \\ & = \int_{\mathbf{R}} \frac{\tilde{f}(x, \gamma)}{\partial \gamma} \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} dx \\ & + \int_{\mathbf{R}} \tilde{f}(x, \gamma) \frac{\partial^2 f(x, \underline{\eta})}{\partial \underline{\eta} \partial \underline{\eta}^T} \frac{\partial \underline{\eta}}{\partial \gamma} dx, \end{aligned}$$

which results in

$$\begin{aligned} & \left(\int_{\mathbf{R}} \underline{F} \underline{F}^T dx + \int_{\mathbf{R}} (f(x, \underline{\eta}) - \tilde{f}(x, \gamma)) \mathbf{M} dx \right) \frac{\partial \underline{\eta}}{\partial \gamma} \\ & = \int_{\mathbf{R}} \underline{F} \frac{\partial \tilde{f}(x, \gamma)}{\partial \gamma} dx, \end{aligned} \quad (1)$$

where

$$\underline{F} = \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} \quad \text{and} \quad \mathbf{M} = \frac{\partial^2 f(x, \underline{\eta})}{\partial \underline{\eta} \partial \underline{\eta}^T}.$$

(1) is a system of ordinary first-order differential equations, which can be written as

$$\mathbf{P}(\underline{\eta}, \gamma) \frac{\partial \underline{\eta}}{\partial \gamma} = \underline{b}(\underline{\eta}, \gamma)$$

with

$$\begin{aligned} \mathbf{P}(\underline{\eta}, \gamma) &= \underbrace{\int_{\mathbf{R}} \underline{F} \underline{F}^T dx}_{\mathbf{P}_1} + \underbrace{\int_{\mathbf{R}} (f(x, \underline{\eta}) - \tilde{f}(x, \gamma)) \mathbf{M} dx}_{\Delta \mathbf{P}} \\ \underline{b}(\underline{\eta}, \gamma) &= \int_{\mathbf{R}} \underline{F} \frac{\partial \tilde{f}(x, \gamma)}{\partial \gamma} dx \end{aligned}$$

We now derive analytic expressions for $\mathbf{P}(\underline{\eta}, \gamma)$ and $\underline{b}(\underline{\eta}, \gamma)$.

1) *Analytical Expression for $\mathbf{P}(\underline{\eta}, \gamma)$* : $\mathbf{P}(\underline{\eta}, \gamma)$ is divided into two parts

$$\mathbf{P}(\underline{\eta}, \gamma) = \mathbf{P}_1(\underline{\eta}) + \Delta \mathbf{P}(\underline{\eta}, \gamma).$$

We give solutions for the parts separately. The first part is

$$\begin{aligned} \mathbf{P}_1(\underline{\eta}) &= \int_{\mathbf{R}} \underline{F}(x, \underline{\eta}) \underline{F}^T(x, \underline{\eta}) dx \\ &= \begin{bmatrix} \mathbf{P}^{(1,1)} & \mathbf{P}^{(1,2)} & \dots & \mathbf{P}^{(1,L)} \\ \mathbf{P}^{(2,1)} & \mathbf{P}^{(2,2)} & \dots & \mathbf{P}^{(2,L)} \\ \vdots & \vdots & \dots & \vdots \\ \mathbf{P}^{(L,1)} & \mathbf{P}^{(L,2)} & \dots & \mathbf{P}^{(L,L)} \end{bmatrix}. \end{aligned}$$

The individual three-by-three block matrices $\mathbf{P}^{(i,j)}$, for $i = 1, \dots, L$ and $j = 1, \dots, L$ are

$$P^{(i,j)} = \int_{\mathbf{R}} \frac{\partial f_i(x, \underline{\eta}_i)}{\partial \underline{\eta}_i} \frac{\partial f_j(x, \underline{\eta}_j)}{\partial \underline{\eta}_j} dx$$

with

$$\underline{\eta}_i = \begin{bmatrix} w_i \\ \mu_i \\ \sigma_i \end{bmatrix}$$

can be obtained analytically as

$$\mathbf{P}^{(i,j)} = \frac{1}{\sqrt{2\pi(\sigma_i^2 + \sigma_j^2)}} \exp\left(-\frac{1}{2} \frac{(\mu_i - \mu_j)^2}{\sigma_i^2 + \sigma_j^2}\right) \cdot \begin{bmatrix} P_{1,1}^{(i,j)} & P_{1,2}^{(i,j)} & P_{1,3}^{(i,j)} \\ P_{2,1}^{(i,j)} & P_{2,2}^{(i,j)} & P_{2,3}^{(i,j)} \\ P_{3,1}^{(i,j)} & P_{3,2}^{(i,j)} & P_{3,3}^{(i,j)} \end{bmatrix}$$

with

$$\begin{aligned} P_{1,1}^{(i,j)} &= 1 \\ P_{1,2}^{(i,j)} &= w_j \frac{\mu_i - \mu_j}{\sigma_i^2 + \sigma_j^2} \\ P_{1,3}^{(i,j)} &= w_j \sigma_j \frac{(\mu_i - \mu_j)^2 - (\sigma_i^2 + \sigma_j^2)}{(\sigma_i^2 + \sigma_j^2)^2} \\ P_{2,1}^{(i,j)} &= w_i \frac{\mu_j - \mu_i}{\sigma_i^2 + \sigma_j^2} \\ P_{2,2}^{(i,j)} &= w_i w_j \frac{\sigma_i^2 + \sigma_j^2 - (\mu_i - \mu_j)^2}{(\sigma_i^2 + \sigma_j^2)^2} \\ P_{2,3}^{(i,j)} &= w_i w_j \sigma_j \frac{(\mu_j - \mu_i)((\mu_i - \mu_j)^2 - 3(\sigma_i^2 + \sigma_j^2))}{(\sigma_i^2 + \sigma_j^2)^3} \\ P_{3,1}^{(i,j)} &= w_i \sigma_i \frac{(\mu_i - \mu_j)^2 - (\sigma_i^2 + \sigma_j^2)}{(\sigma_i^2 + \sigma_j^2)^2} \\ P_{3,2}^{(i,j)} &= w_i w_j \sigma_i \frac{(\mu_j - \mu_i)((\mu_i - \mu_j)^2 - 3(\sigma_i^2 + \sigma_j^2))}{(\sigma_i^2 + \sigma_j^2)^3} \\ P_{3,3}^{(i,j)} &= w_i w_j \sigma_i \sigma_j \cdot \frac{(\mu_i - \mu_j)^4 + 3(\sigma_i^2 + \sigma_j^2)(\sigma_i^2 + \sigma_j^2 - 2(\mu_i - \mu_j)^2)}{(\sigma_i^2 + \sigma_j^2)^4}. \end{aligned}$$

The expression for $\Delta \mathbf{P}$ is given by

$$\Delta \mathbf{P} = \int_{\mathbf{R}} (f(x, \underline{\eta}) - \tilde{f}(x, \gamma)) \mathbf{M} dx,$$

where

$$\mathbf{M} = \frac{\partial^2 f(x, \underline{\eta})}{\partial \underline{\eta} \partial \underline{\eta}^T} = \begin{bmatrix} \mathbf{M}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_2 & & \mathbf{0} \\ \vdots & & \ddots & \vdots \\ \mathbf{0} & \dots & \mathbf{0} & \mathbf{M}_L \end{bmatrix}.$$

The solutions for the three-by-three block matrices \mathbf{M}_i are given in figure 3.

$$\mathbf{M}_i = f(x, \underline{\eta}) \cdot \begin{bmatrix} 0 & \frac{x-\mu_i}{w_i \sigma_i^2} & \frac{(x-\mu_i)^2 - \sigma_i^2}{w_i \sigma_i^3} \\ \frac{x-\mu_i}{w_i \sigma_i^2} & \frac{(x-\mu_i)^2 - \sigma_i^2}{\sigma_i^4} & \frac{(x-\mu_i)^3 - 3\sigma_i^2(x-\mu_i)}{\sigma_i^5} \\ \frac{(x-\mu_i)^2 - \sigma_i^2}{w_i \sigma_i^3} & \frac{(x-\mu_i)^3 - 3\sigma_i^2(x-\mu_i)}{\sigma_i^5} & \frac{(x-\mu_i)^4 - 5\sigma_i^2(x-\mu_i)^2 + 2\sigma_i^4}{\sigma_i^6} \end{bmatrix}$$

Fig. 3. Three-by-three block matrix \mathbf{M}_i for $\Delta\mathbf{P}$.

2) *Expression for $\underline{b}(\underline{\eta}, \gamma)$* : The expression for $\underline{b}(\underline{\eta}, \gamma)$ from (1) is

$$\underline{b}(\underline{\eta}, \gamma) = \int_{\mathbf{R}} \frac{\partial \tilde{f}(x, \gamma)}{\partial \gamma} \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}} dx .$$

The partial derivation $\frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}}$ is a vector with the elements

$$\frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}_i} = f_i(x, \underline{\eta}_i) \begin{bmatrix} \frac{1}{w_i} \\ \frac{x-\mu_i}{\sigma_i^2} \\ \frac{(x-\mu_i)^2 - \sigma_i^2}{\sigma_i^3} \end{bmatrix} .$$

The partial derivation of $\tilde{f}(x, \gamma)$ with respect to γ is

$$\begin{aligned} \frac{\partial \tilde{f}(x, \gamma)}{\partial \gamma} &= \frac{\partial \tilde{f}_1(x, \gamma)}{\partial \gamma} \cdot f_2(x) \\ &= \sum_{i=1}^M w_{i,1} \frac{-(\gamma - \epsilon)(x - \mu_{i,1})^2}{(1 + \epsilon)^2 \sigma_{i,1}^2} \\ &\quad \cdot \exp \left\{ -\frac{1}{2} \frac{(x - \mu_{i,1})^2}{(\sigma_{i,1} \frac{1+\epsilon}{\gamma+\epsilon})^2} \right\} \cdot f_2(x) . \end{aligned}$$

Then we have

$$\underline{b}(\underline{\eta}, \gamma) = \begin{bmatrix} \underline{b}^{(1)}(\underline{\eta}, \gamma) \\ \underline{b}^{(2)}(\underline{\eta}, \gamma) \\ \vdots \\ \underline{b}^{(L)}(\underline{\eta}, \gamma) \end{bmatrix}$$

with

$$\underline{b}^{(i)}(\underline{\eta}, \gamma) = \int_{\mathbf{R}} \frac{\partial \tilde{f}_1(x, \gamma)}{\partial \gamma} f_2(x) f_i(x, \underline{\eta}_i) \begin{bmatrix} \frac{1}{w_i} \\ \frac{x-\mu_i}{\sigma_i^2} \\ \frac{(x-\mu_i)^2 - \sigma_i^2}{\sigma_i^3} \end{bmatrix} dx .$$

3) *Solving the Differential Equation*: We solve the system of ordinary first-order differential equations (ODE) on the interval $\gamma \in [0 \dots 1]$. Since this ODE cannot be solved directly we have to use numerical approaches, which solve stepwise on this interval. This can be accomplished with an appropriate solver, e.g. Euler backward method or a Runge-Kutta solver.

In every step, the algorithm checks if the approximation error is smaller than the user defined threshold.

4) *Structural Adaptation*: During the process of solving the differential equation the deviation between the parameterized exact density $\tilde{f}(x, \gamma)$ and the approximation density $f(x, \underline{\eta})$ may become higher than the threshold that was defined by the user.

This deviation error can be decreased by adding components to the approximation density $f(x, \underline{\eta})$. For this purpose we

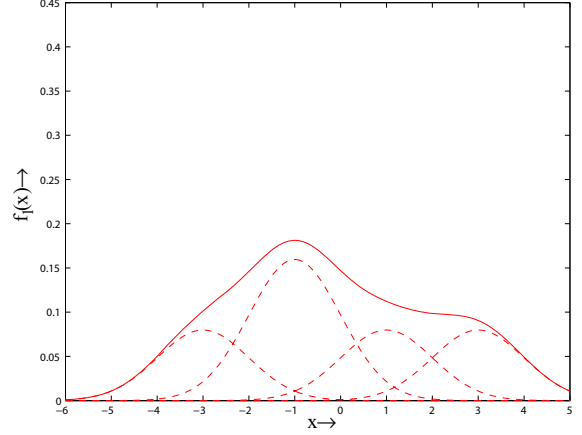


Fig. 4. A priori density $f_1(x)$ with large uncertainty. Dashed lines depict the components of this mixture.

split the component which yields the largest error into several Gaussian components as described in [9].

The user-defined error can be used in order to trade accuracy for speed of the algorithm.

IV. EXAMPLES

We now give examples of the application of the presented approach. To compare the resulting approximation to the exact function, we use the normalized distance measure

$$d = \frac{\int_{\mathbf{R}} (f_p(x) - f(x))^2 dx}{\int_{\mathbf{R}} (f(x))^2 dx + \int_{\mathbf{R}} (f_p(x))^2 dx} ,$$

which varies from 0 to 1. $d = 0$ indicates a perfect match and $d = 1$ is the maximum possible error. To give an error in percent we use $\sqrt{d} \cdot 100$.

A. Example 1

Consider a filtering experiment where we only have vague a priori knowledge with large uncertainty. Based on a noisy measurement we update this knowledge.

In this example, a priori knowledge is given by a Gaussian mixture density $f_1(x)$ with four components. The parameters are given by

$$\begin{aligned} w_1 &= [0.2 \quad 0.4 \quad 0.2 \quad 0.2] \\ \mu_1 &= [-3 \quad -1 \quad 1 \quad 3] \\ \sigma_1 &= [1 \quad 1 \quad 1 \quad 1] , \end{aligned}$$

which results in the density shown in figure 4.

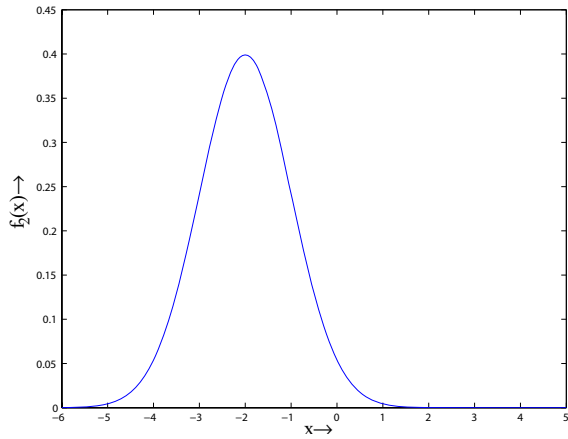


Fig. 5. Measurement density $f_2(x)$ with smaller uncertainty.

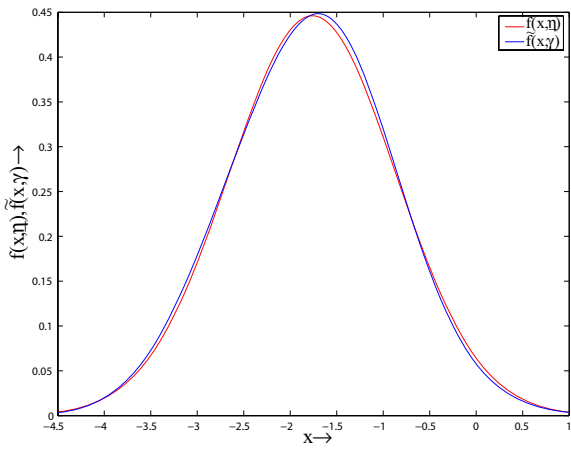


Fig. 6. The approximation of $\tilde{f}(x, \gamma)$ with one component for $f(x, \eta)$ yields an error of 1.64% .

The noisy measurement is less uncertain than the a priori knowledge and is presented by a single Gaussian density $f_2(x)$ with the parameters

$$\mu = -2$$

and

$$\sigma = 1$$

as shown in figure 5.

Please keep in mind, that the exact product of $f_1(x)$ and $f_2(x)$ has four components.

If we accept an approximation error of maximum 2%, the solver needs 18 steps and only one Gaussian component with the parameters

$$\mu = -1.7606$$

and

$$\sigma = 0.8944$$

is obtained. The approximation error at the end of the run is 1.64%. The comparison of the approximated density to the exact product density is shown in figure 6.

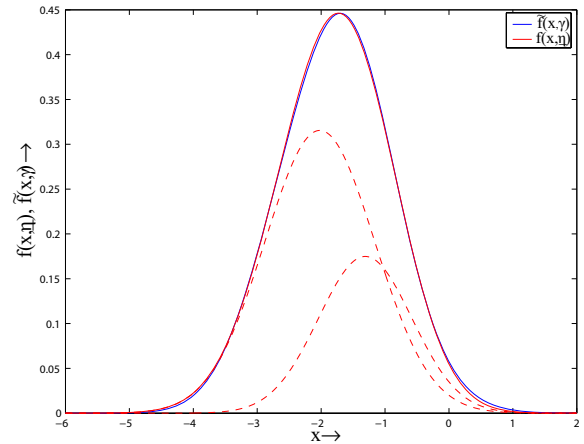


Fig. 7. The approximation of $\tilde{f}(x, \gamma)$ with two components for $f(x, \eta)$ yields an error of 0.66% . The dashed lines show the individual components of the approximation.

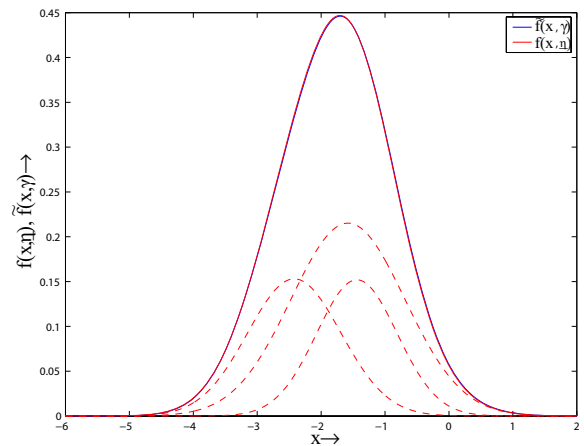


Fig. 8. The approximation of $\tilde{f}(x, \gamma)$ with three components for $f(x, \eta)$ yields an error of 0.064% . The dashed lines show the individual components of the approximation.

If we decrease the maximum accepted error to 1%, the algorithm has to perform a structural approximation once and we receive a mixture with

$$\begin{aligned} w &= [0.68 \quad 0.32] \\ \mu &= [-2.02 \quad -1.31] \\ \sigma &= [0.86 \quad 0.73] \end{aligned}$$

as parameters. The resulting approximation error is 0.66%. The solver needs 37 steps for this result. The resulting mixture is shown in figure 7.

Decreasing the maximum accepted error further to 0.5% leads to three components in the result. Their parameters are

$$\begin{aligned} w &= [0.28 \quad 0.48 \quad 0.24] \\ \mu &= [-2.42 \quad -1.58 \quad -1.44] \\ \sigma &= [0.73 \quad 0.89 \quad 0.63], \end{aligned}$$

as shown in figure 8. The solver needs 77 steps and gives an approximation error of 0.064%.

TABLE I
APPROXIMATION RESULTS FOR DIFFERENT USER SELECTED
APPROXIMATION ERROR THRESHOLDS.

User threshold	2%	1%	0.5%
# of components	1	2	3
Approx. error	1.64%	0.66%	0.06%
# of solver steps	18	37	77

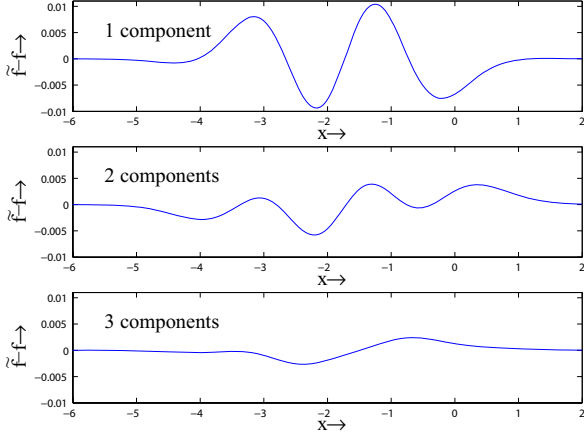


Fig. 9. The error $\tilde{f}(x, \gamma) - f(x, \eta)$ for the three approximations from example 1 with 1, 2, and 3 components.

The results of the experiment with the different user selected thresholds are given in table I. It shows how the user definable threshold can be used to control the tradeoff between approximation quality and speed of the algorithm.

Figure 9 shows the error $\tilde{f}(x, \gamma) - f(x, \eta)$ for the three resulting approximations from our experiment governed by the chosen threshold.

B. Example 2

As a second example we approximate the product of two more complicated mixtures. These densities are nearly the same as in section III-B. We just altered the weights and standard deviations. For f_1 we have the parameters

$$\begin{aligned} w_1 &= [0.5 \ 0.5] \\ \mu_1 &= [-2 \ 2] \\ \sigma_1 &= [2 \ 2] \end{aligned}$$

as shown in figure 10.

For f_2 we have the parameters

$$\begin{aligned} w_2 &= [0.4 \ 0.3 \ 0.3] \\ \mu_2 &= [-4 \ 0 \ 4] \\ \sigma_2 &= [0.9 \ 1 \ 1.2] \end{aligned}$$

as shown in figure 11. The algorithm approximates the product of f_1 and f_2 with three components

$$\begin{aligned} w &= [0.31 \ 0.46 \ 0.23] \\ \mu &= [-3.65 \ 0 \ 3.44] \\ \sigma &= [0.82 \ 1 \ 1.04] \end{aligned}$$

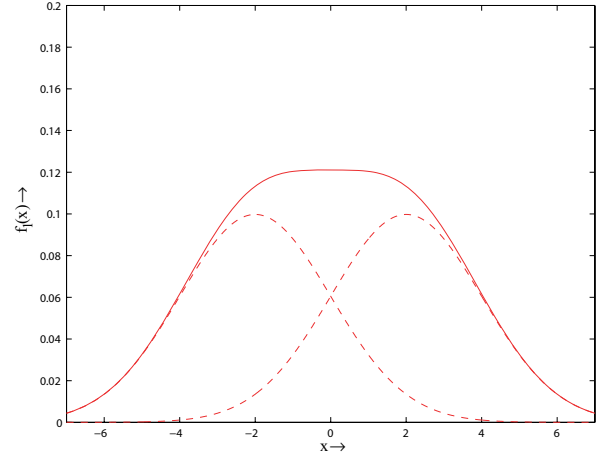


Fig. 10. Density f_1 for Example 2. The dashed lines show the individual components.

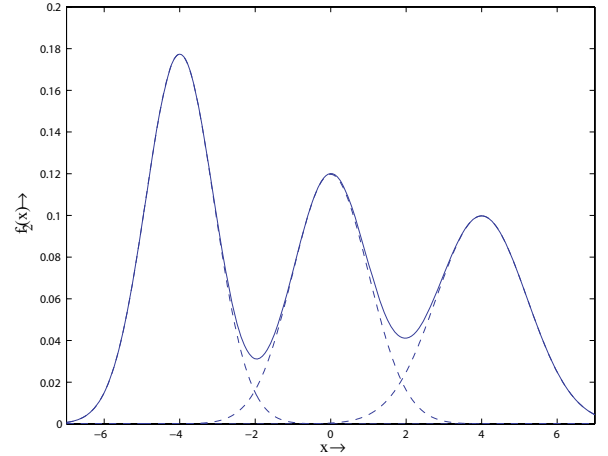


Fig. 11. Density f_2 for Example 2. The dashed lines show the individual components.

as shown in figure 12.

The resulting approximation error is 0.075585% and the solver needed 23 steps. The chosen maximum error for this experiment was 2%.

The exact product of these two densities consists of six components and is shown in figure 13. This is an example, where the approach finds a very accurate approximation with less components. In order to compare our algorithm to another approach, we chose the Salmond mixture reduction as described in [8]. The Clustering threshold was set to 0.09. We received a Gaussian mixture with 3 components as shown in figure 14. The error of the reduced version was 3.1344%.

V. CONCLUSIONS

In this paper a method for approximating the product of two Gaussian mixtures with a Gaussian mixture comprising less components than the exact product has been presented. The applied method guarantees that the approximation error between the product function and its approximation is smaller than a user predefined maximum error.

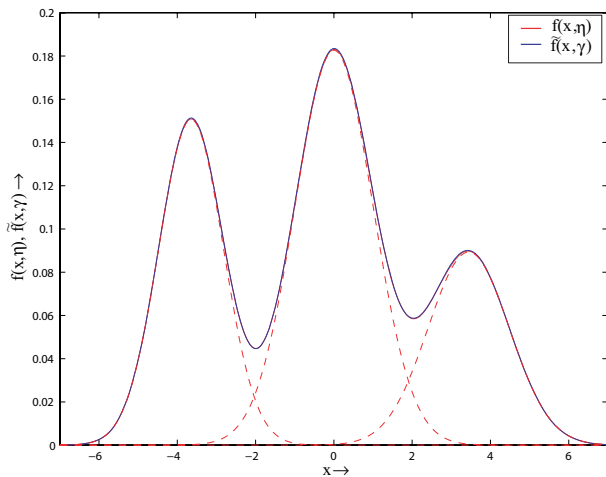


Fig. 12. The approximation of $f_1 \cdot f_2$. The dashed lines show the individual components. Due to the small approximation error of 0.075585% it is hard to discriminate between the exact density and its approximation.

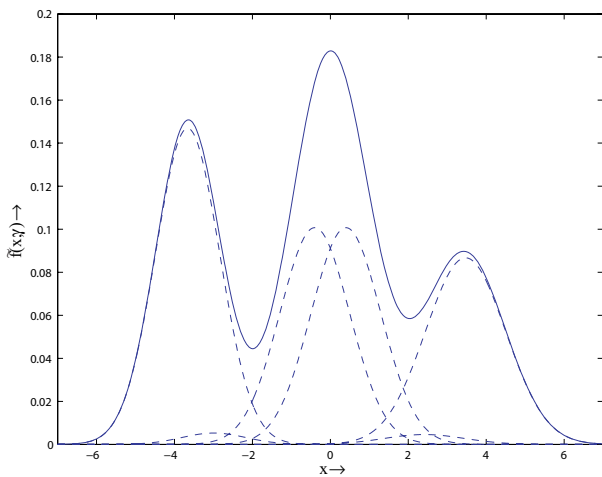


Fig. 13. The exact product of $f_1 \cdot f_2$. The dashed lines show the individual components.

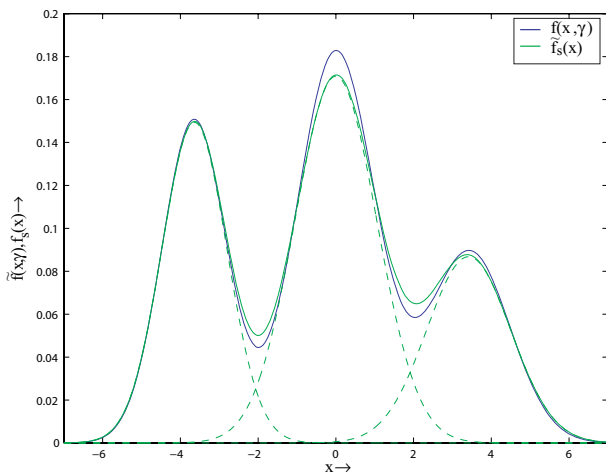


Fig. 14. Salmund Mixture Reduction in comparison to the exact product. The dashed lines show the individual components. The error is 3.1344%.

This re-approximation of the product of two Gaussian mixtures is extremely helpful for recursive processing schemes, since the complexity of resulting mixture densities can be kept at a tractable level.

Further applications are cascaded systems like continuous or hybrid nonlinear Bayesian Networks as given in [11], [12]. In these networks, densities are propagated from node to node, which involves multiplication with other densities. It is clear, that arbitrarily large networks lead to an enormous amount of components. Not to speak of dynamic networks of this kind.

Still an open question is, how the approach behaves in online processing. All experiments made so far were conducted with unoptimized Matlab code. It is expected that an implementation in C++ and optimization of the ODE solver leads to a real time capable application. Studies on this topic are currently conducted.

REFERENCES

- [1] E. Parzen, "On Estimation of a Probability Density Function and Mode," *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1065 – 1076, September 1962.
- [2] W. Poland and R. Shachter, "Mixtures of Gaussians and Minimum Relative Entropy Techniques for Modeling Continuous Uncertainties," in *Proceedings of the 9th Annual Conference on Uncertainty in Artificial Intelligence (UAI-93)*. San Francisco, CA: Morgan Kaufmann Publishers, 1993. [Online]. Available: <http://www.stanford.edu/~shachter/pubs/poland93.pdf>
- [3] A. Sanjeev and R. Kannan, "Learning Mixtures of Arbitrary Gaussians," in *STOC '01: Proceedings of the thirty-third annual ACM Symposium on Theory of computing*. ACM Press, 2001, pp. 247–257. [Online]. Available: <http://doi.acm.org/10.1145/380752.380808>
- [4] A. T. Ihler, E. B. Sudderth, W. T. Freeman, and A. S. Willsky, "Efficient Multiscale Sampling from Products of Gaussian Mixtures," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004. [Online]. Available: http://books.nips.cc/papers/files/nips16/NIPS2003_AA01.pdf
- [5] Y. Bar-Shalom and T. E. Fortmann, *Tracking and Data Association*, ser. Mathematics in Science and Engineering. Academic Press, 1998, vol. 179.
- [6] P. L. Bogler, *Radar Principles with Applications to Tracking Systems*. Wiley, New York, 1990.
- [7] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association," *Oceanic Engineering, IEEE Journal of*, vol. 8, no. 3, pp. 173–184, 1983.
- [8] L. Y. Pao, "Multisensor Multitarget Mixture Reduction Algorithms for Tracking," *AIAA Journal of Guidance, Control and Dynamics*, vol. 17, no. 6, pp. 1205–1211, 1994. [Online]. Available: citeseer.ist.psu.edu/pao94multisensor.html
- [9] U. D. Hanebeck, K. Briechle, and A. Rauh, "Progressive Bayes: A New Framework for Nonlinear State Estimation," in *Proceedings of SPIE*, vol. 5099, Orlando, Florida, 2003, pp. 256–267, aeroSense Symposium. [Online]. Available: http://isas.uka.de/downloads/hanebeck_spie2003_progbyes.pdf
- [10] J. R. Dormand and P. J. Prince, "A Family of Embedded Runge-Kutta Formulae," *J. Comput. Appl. Math.*, vol. 6, no. 1, pp. 19–26, 1980.
- [11] E. Driver and D. Morrell, "Implementation of Continuous Bayesian Networks Using Sums of Weighted Gaussians," in *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, Besnard and Hanks, Eds., Montreal, Quebec, Canada, 1995, pp. 134–140.
- [12] O. C. Schrepf and U. D. Hanebeck, "A New Approach for Hybrid Bayesian Networks Using Full Densities," in *Proceedings of 6th Workshop on Computer Science and Information Technologies, CSIT 2004*, Budapest, Hungary, 2004. [Online]. Available: <http://isas.uka.de/downloads/csit2004.pdf>