Progressive Gaussian Mixture Reduction

Marco F. Huber and Uwe D. Hanebeck
Intelligent Sensor-Actuator-Systems Laboratory
Institute of Computer Science and Engineering
Universität Karlsruhe (TH), Germany
Email: marco.huber@ieee.org, uwe.hanebeck@ieee.org

***Abstract*—For estimation and fusion tasks it is inevitable to approximate a Gaussian mixture by one with fewer components to keep the complexity bounded. Appropriate approximations can be typically generated by exploiting the redundancy in the shape description of the original mixture. In contrast to the common approach of successively merging pairs of components to maintain a desired complexity, the novel Gaussian mixture reduction algorithm introduced in this paper avoids to directly reduce the original Gaussian mixture. Instead, an approximate mixture is generated from scratch by employing homotopy continuation. This allows starting the approximation with a single Gaussian, which is constantly adapted to the progressively incorporated true Gaussian mixture. Whenever a user-defined bound on the deviation of the approximation cannot be maintained during the continuation, further components are added to the approximation. This facilitates significantly reducing the number of components even for complex Gaussian mixtures.**

Keywords: Gaussian mixture reduction, nonlinear optimization, homotopy continuation

I. Introduction

Thanks to their universal approximation property, Gaussian mixtures are a very convenient function system for representing probability densities. Particularly in estimation tasks like multi-target tracking [1], density estimation [2], [3], nonlinear filtering [4] or machine learning [5], Gaussian mixtures are employed for accurately representing multimodalities. However, recursive processing of Gaussian mixtures generally leads to an exponential growth of the number of mixture components. In order to keep the computational and memory requirements bounded, it is inevitable to control this growth.

Several methods were developed in recent years for reducing the amount of Gaussian components. Typically, the reduction is achieved by deleting components with low contribution to the overall mixture or by successively merging components with strong similarity. Depending on the measure applied for selecting components, Gaussian mixture reduction methods can be classified into two groups: *Local* algorithms only consider lower-order statistics of the mixture like mean and variance or completely disregard the overall effect when evaluating the similarity between components. Salmond's joining and clustering algorithms [6], [7] or West's algorithm [8] are part of this group. On the other hand, the measure employed in *global* methods like in [9], [10] considers all available information, i.e., shape information of the mixture, when selecting components for reduction.

Compared to local methods, the reduction results of global methods are typically more accurate, at the expense of a higher computational effort for reduction. One way to benefit from both reduction approaches is to evaluate a localized version of a global measure as done in [11], where a computationally cheap upper bound of the Kullback-Leibler divergence measure is minimized. However, still the common approach of starting the reduction with the complex Gaussian mixture and reducing it by successively merging pairs of Gaussians is applied. For obtaining a specific approximation quality, merging approaches often end up with a number of components that is still too large, as the inherent redundancy of the original mixture is exploited only in a greedy fashion.

To fully exploit the approximation potential of reduced-order Gaussian mixtures, the global Gaussian mixture reduction approach introduced in this paper employs a dual principle to existing algorithms. Instead of repeatedly removing mixture components, a Gaussian mixture is successively built up to approximate the original mixture with far less components. However, a priori determining the optimal number of components for maintaining a specific deviation to the original mixture is impossible. Thus, a homotopy continuation approach is employed, which starts with a single Gaussian density (for an introduction to homotopy continuation see e.g. [12]). During the continuation toward the original mixture, new Gaussian components are added to the approximate mixture by splitting existing components for providing better approximation capabilities. To control this growth in components, the deviation is constantly tracked by the squared integral distance measure and new components are only added in regions of emerging strong deviations when necessary.

For obtaining accurate results in an efficient way, the progress of the continuation is controlled by a predictor-corrector scheme. The continuation progresses faster whenever the changes generated by the gradually incorporated original mixture are marginal. On the other hand, strong changes slow down the progression such that accurately adapting the approximation is possible. Furthermore, for enabling an efficient implementation of the proposed reduction method, closed-form solutions for all necessary calculations are derived.

In the next section, the Gaussian mixture reduction problem is briefly introduced. The remainder of the paper is structured as follows: The formulation of the reduction problem as constrained optimization problem and the corresponding continuation solution algorithm is provided in Section III, while Section IV is concerned with the adaptation procedures for improving the accuracy and efficiency of the continuation algorithm. These adaptations comprise the speed of progression as well as structurally adapting the approximation by adding new components. The effectiveness of the proposed Gaussian mixture reduction algorithm in comparison to state-of-the-art algorithms is demonstrated by means of simulations in Section V. Finally, we give conclusions and an outlook to future work.

II. Problem Formulation

It is assumed that the true density function of the random variable x is represented by the Gaussian mixture

$$\tilde{f}(x) = \sum_{i=1}^{M} \omega_i \cdot \mathcal{N}\left(x; \mu_i, \sigma_i^2\right) \ ,$$

where ω_i are non-negative weighting coefficients with $\sum_i \omega_i = 1$ and $\mathcal{N}\left(x; \mu, \sigma^2\right)$ is a Gaussian density with mean μ and variance σ^2. For the sake of brevity and clarity only one-dimensional random variables are considered in this paper.

In typical estimation and fusion tasks, the number of mixture components M increases exponentially over time. Due to computational and memory limitations, this growing mixture cannot be processed for any significant time span. Even when $\tilde{f}(x)$ has a large number of components, the shape of the Gaussian mixture is often not that complex, e.g., a mode of the true density is represented by several Gaussians, whereas a single component would be adequate for approximating the mode. Thus, a Gaussian mixture with a considerable smaller number of components can typically be found by fusing locally shared information and by removing redundancy in $\tilde{f}(x)$.

The goal is now to find a *reduced Gaussian mixture*

$$f(x, \underline{\eta}) = \sum_{j=1}^{L} \omega_j^2 \cdot \mathcal{N}\left(x; \mu_j, \sigma_j^2\right) \ , \qquad (1)$$

with parameter vector

$$\underline{\eta} = [\underline{\eta}_1^{\mathrm{T}}, \underline{\eta}_2^{\mathrm{T}}, \ldots, \underline{\eta}_L^{\mathrm{T}}]^{\mathrm{T}} \ , \ \underline{\eta}_j = [\omega_j, \mu_j, \sigma_j]^{\mathrm{T}} \ ,$$

consisting of $L \ll M$ components that is close to the original mixture $\tilde{f}(x)$.[1] A distance measure $G\big(\tilde{f}(x), f(x, \underline{\eta})\big)$ is used for quantifying the deviation or the similarity between both mixtures, which in turn allows adapting the parameters $\underline{\eta}$, i.e., the weights, means, and variances, of $f(x, \underline{\eta})$ in order to minimize the deviation.

III. Gaussian Mixture Reduction via Homotopy Continuation

The key idea is to reformulate the Gaussian mixture reduction problem as an optimization problem

$$\underline{\eta}_{\min} = \arg\min_{\underline{\eta}} G\big(\tilde{f}(x), f(x, \underline{\eta})\big) \qquad (2)$$
$$\text{w.r.t.} \ \ G\big(\tilde{f}(x), f(x, \underline{\eta}_{\min})\big) \leq G_{\max}$$

by minimizing a certain distance measure $G\big(\tilde{f}(x), f(x, \underline{\eta})\big)$ under the constraint that the deviation between $\tilde{f}(x)$ and $f(x, \underline{\eta})$ is less than an user-defined maximum value $G_{\max}$. Besides defining a maximum deviation it is also possible to additionally constrain the number of used components for $f(x, \underline{\eta})$.[2] Thus, the user is able to adjust the quality as well as the computational demand of the reduction by giving a limit on the allowed deviation and/or the number of components.

A. Progressive Processing

The optimization problem (2) is generally not convex, so that directly minimizing the deviation between both mixture densities results in getting trapped in an unappropriate local optimum. Furthermore, the optimal number of mixture components L is not known a priori for maintaining a deviation less than $G_{\max}$. To overcome these problems, the proposed reduction approach makes use of the Progressive Bayes framework introduced in [13], i.e., a specific type of homotopy continuation is applied in order to find the solution of (2) progressively.

In doing so, a so-called *progression parameter* $\gamma \in [0, 1]$ is used for parameterizing the original Gaussian mixture $\tilde{f}(x)$ in such a way that for $\gamma = 0$ the Gaussian mixture can be reduced directly, i.e., the exact solution of the optimization problem is known without deviation from $\tilde{f}(x)$. By incrementing the progression parameter, the effect of the original mixture is introduced gradually. This ensures a continuous transformation of the optimal solution of the initial optimization problem toward the desired original Gaussian mixture $\tilde{f}(x)$, by progressively adjusting the parameters $\underline{\eta}$ of $f(x, \underline{\eta})$ to keep $G\big(\tilde{f}(x), f(x, \underline{\eta})\big)$ at a minimum.

B. Parameterization

For that purpose, the *parameterized Gaussian mixture* $\tilde{f}(x, \gamma)$ is given by

$$\tilde{f}(x, \gamma) = \gamma \cdot \tilde{f}(x) + (1 - \gamma) \cdot \hat{f}(x) \ . \qquad (3)$$

Hence, we obtain

$$\tilde{f}(x, 0) = \hat{f}(x) \ \ \text{and} \ \ \tilde{f}(x, 1) = \tilde{f}(x) \ ,$$

where the Gaussian mixture $\hat{f}(x)$ should be chosen such that performing its reduction is straightforward. A very natural choice is a single Gaussian $\hat{f}(x) = \mathcal{N}(x; \mu, \sigma^2)$, whose mean μ and variance σ^2 correspond to the mean and variance of the original mixture $\tilde{f}(x)$, i.e.,

$$\mu = \sum_{i=1}^{M} \omega_i \cdot \mu_i \ , \ \sigma^2 = \sum_{i=1}^{M} \omega_i \cdot \left(\sigma_i^2 + \mu_i^2\right) - \mu^2 \ .$$

Using a single Gaussian density for capturing these first two moments automatically minimizes the Kullback-Leibler divergence or equivalently maximizes the entropy [14]. Starting the progression with this "simple" density allows directly determining the optimal solution, i.e., $f(x, \underline{\eta}) = \hat{f}(x)$ with

[1] Please note that squared weighting coefficients ω_j^2 are used to ensure that $f(x, \underline{\eta})$ remains a valid density function during the reduction process.

[2] Obviously, in case of an additional component constraint, the maximum deviation is not guaranteed to be maintained.

$\underline{\eta} = [1, \mu, \sigma]^{\mathrm{T}}$. This solution, i.e., the parameter vector $\underline{\eta}$, then tracks the original Gaussian mixture that is progressively modified by increasing γ.

As the initialization of the continuation indicates, the way the proposed mixture reduction approach operates is dual to existing algorithms. Instead of beginning with the complete original mixture, at first a less complex reduction or approximation task is solved. As it is shown in Section IV-B, splitting operations are used in order to add new Gaussian components to the initial single Gaussian when required. This ensures to achieve the maximum deviation value $G_{\max}$.

C. Distance Measure

For quantifying the deviation between $\tilde{f}(x,\gamma)$ and $f(x,\underline{\eta})$, several measures $G(\,\cdot\,)$ can be used. For convenience, the squared integral distance measure [15]

$$G\big(\tilde{f}(x,\gamma), f(x,\underline{\eta})\big) = \frac{1}{2}\int_{\mathbb{R}} \Big(\tilde{f}(x,\gamma) - f(x,\underline{\eta})\Big)^2 \,\mathrm{d}x \quad (4)$$

is chosen, since it can be evaluated analytically for Gaussian mixtures. However, the proposed approach is not restricted to this specific deviation measure. For instance, the Kullback-Leibler divergence [16] can also be used, especially as it is the ideal deviation measure for mixture reduction in a maximum likelihood sense [10], [11]. Due to the fact that it is impossible to evaluate this measure in closed form for Gaussian mixtures, numerical integration schemes have to be employed, which leads to increased computational costs.

In the following, we write $G(\underline{\eta},\gamma)$ shorthand for $G\big(\tilde{f}(x,\gamma), f(x,\underline{\eta})\big)$.

D. Progressive Minimization

To perform the progression of γ from 0 to 1, while keeping the distance measure at its minimum, the differential relation between γ and the parameter vector $\underline{\eta}$, i.e., the variation of $\underline{\eta}$ depending on the variation of γ, is required. Hence, the optimization problem (2) is transformed into a system of ordinary differential equations (ODE). In order to obtain these differential equations, the necessary condition of a minimum of $G(\underline{\eta},\gamma)$ has to be satisfied. Thus, derivatives of $G(\underline{\eta},\gamma)$ with respect to γ and $\underline{\eta}$ have to be zero, as $G(\underline{\eta},\gamma)$ is a function over γ and $\underline{\eta}$. Taking the partial derivative of $G(\underline{\eta},\gamma)$ with respect to the parameter vector $\underline{\eta}$ yields

$$\frac{\partial G(\underline{\eta},\gamma)}{\partial \underline{\eta}} = -\int_{\mathbb{R}} \Big(\tilde{f}(x,\gamma) - f(x,\underline{\eta})\Big)\, \underline{F}(x,\underline{\eta}) \,\mathrm{d}x \ , \quad (5)$$

where

$$\underline{F}(x,\underline{\eta}) = \frac{\partial f(x,\underline{\eta})}{\partial \underline{\eta}} \ .$$

By setting (5) to zero we obtain

$$\int_{\mathbb{R}} \tilde{f}(x,\gamma)\underline{F}(x,\underline{\eta}) \,\mathrm{d}x = \int_{\mathbb{R}} f(x,\underline{\eta})\underline{F}(x,\underline{\eta}) \,\mathrm{d}x \ .$$

The partial derivative with respect to γ gives the desired system of ordinary first-order differential equations

$$\Bigg(\underbrace{\int_{\mathbb{R}} \underline{F}(x,\underline{\eta})\underline{F}(x,\underline{\eta})^{\mathrm{T}} \,\mathrm{d}x +}_{=:\mathbf{P}'(\underline{\eta})} \underbrace{\int_{\mathbb{R}} \Big(f(x,\underline{\eta}) - \tilde{f}(x,\gamma)\Big)\mathbf{M}(x,\underline{\eta}) \,\mathrm{d}x}_{=:\Delta\mathbf{P}(\underline{\eta},\gamma)}\Bigg)\frac{\partial \underline{\eta}}{\partial \gamma} = \int_{\mathbb{R}} \underline{F}(x,\underline{\eta})\frac{\partial \tilde{f}(x,\gamma)}{\partial \gamma} \,\mathrm{d}x \ ,$$

where

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

This can be written as

$$\mathbf{P}(\underline{\eta},\gamma) \cdot \dot{\underline{\eta}} = \underline{b}(\underline{\eta},\gamma) \ , \quad (6)$$

where the coefficients are given by

$$\mathbf{P}(\underline{\eta},\gamma) = \mathbf{P}'(\underline{\eta}) + \Delta\mathbf{P}(\underline{\eta},\gamma) \ , \quad (7)$$

$$\underline{b}(\underline{\eta},\gamma) = \int_{\mathbb{R}} \underline{F}(x,\underline{\eta})\frac{\partial \tilde{f}(x,\gamma)}{\partial \gamma} \,\mathrm{d}x \ . \quad (8)$$

Closed-form expressions for (7) and (8) are given in Appendix A and Appendix B, respectively. Due to the squared weights in (1) and the specific parameterization in (3), these expressions do significantly differ to those in [9], [13].

E. Solving the System of Ordinary Differential Equations

The system of ODEs (6) cannot be solved analytically in general. Thus, a numerical solution scheme has to be used. One option is to employ well-known ODE solvers like Runge-Kutta. However, for this specific case these methods often turned out to be numerically unstable. Instead, the numerical solver given in Algorithm 1 is proposed.

The algorithm starts with $\gamma = 0$ and thus with an optimal choice of the parameter vector $\underline{\eta}$ (see line 1-2). During the solution process, γ is gradually increased while $\underline{\eta}$ is simultaneously adjusted (line 5-8). Please note that solving the ODE in line 6 can be carried out directly, as γ is a fixed value and thus, merely a system of linear equations $\mathbf{P}\dot{\underline{\eta}} = \underline{b}$ has to be

Algorithm 1 Pseudo-code of the numerical solver for (6)

```
1: $\gamma \leftarrow 0$
2: $\underline{\eta} \leftarrow \underline{\eta}(\gamma = 0)$
3: $\Delta\gamma \leftarrow \gamma_{\min}$
4: **repeat**
5:     $\gamma \leftarrow \gamma + \Delta\gamma$
6:     $\dot{\underline{\eta}} \leftarrow \text{solve}\big(\mathbf{P}(\underline{\eta},\gamma), \underline{b}(\underline{\eta},\gamma)\big)$
7:     $\underline{\eta}_{\text{tmp}} \leftarrow \underline{\eta} + \Delta\gamma \cdot \dot{\underline{\eta}}$
8:     $[\underline{\eta},\ \gamma,\ \Delta\gamma] \leftarrow \text{Adaptation}\Big(\underline{\eta}_{\text{tmp}}, G_{\max}\Big)$
9: **until** $\gamma = 1$
```

solved, e.g., by employing LU factorization.

With the solution vector $\dot{\underline{\eta}}$ for a specific γ, a so-called *predictor-corrector scheme* can be realized, which is quite common in homotopy continuation [12], [17]. Here, the predictor is represented by line 7, where $\dot{\underline{\eta}}$ gives the direction for predicting $\underline{\eta}$, while the step size $\Delta\gamma$ gives the increment in prediction direction.

Typically, this prediction step causes an error governed by the current step size. For reducing the introduced error under the user-defined error bound $G_{\max}$, a correction or adaptation step is applied subsequently (line 8). In this paper, the term adaptation is used instead of correction, as not only a correction of $\underline{\eta}$ is performed after the prediction. In fact, new Gaussian components are introduced by splitting existing Gaussians, if the deviation between $\tilde{f}(x,\gamma)$ and $f(x,\underline{\eta})$ is still larger than $G_{\max}$. This procedure facilitates *adapting* $f(x,\underline{\eta})$ to emerging structural changes in $\tilde{f}(x,\gamma)$ during the progression. The methods used for adaptation are described in detail in the following section.

IV. Adaptations

A straightforward way to realize the adaptation is to keep the step size always at the minimum size $\gamma_{\min}$. This leads to a linear increment of γ. However, choosing an appropriate $\gamma_{\min}$ is critical, since one has to balance between a compensation of even marginal changes in $\tilde{f}(x,\gamma)$ and a fast progression, leading to a coarse error reduction at some parts of the progression.

A. Parameter and Step Size Adaptation

Since the distance measure has to be minimized for a specific γ, a Newton approach for determining the roots of (5) is applied [17]. This allows correcting $\underline{\eta}$ in order to compensate the introduced error. Furthermore, γ and the step size $\Delta\gamma$ are adjusted for controlling the speed of the progression. This can be done due to the fact that a fast convergence of the Newton approach indicates only a small error introduced by the prediction. Hence, the step size can be increased for the next progression step. The opposite case, where the Newton approach does not converge, indicates a large error. Thus, the prediction step can be reverted by setting γ to its former value and the step size can be decreased.

For obtaining this adaptation, the Newton approach

$$\mathbf{H}(\underline{\eta}_k,\gamma)\cdot\Delta\underline{\eta} = \frac{\partial G(\underline{\eta},\gamma)}{\partial\underline{\eta}}\Big|_{\underline{\eta}=\underline{\eta}_k} =: \underline{h}(\underline{\eta}_k,\gamma) \ , \qquad (9)$$

has to be applied. A closed-form expression of the gradient of the distance measure $\underline{h}(\underline{\eta}_k,\gamma)$ is given in Appendix C, while the Hessian

$$\mathbf{H}(\underline{\eta}_k,\gamma) = \frac{\partial^2 G(\underline{\eta},\gamma)}{\partial\underline{\eta}\,\partial\underline{\eta}^{\mathrm{T}}}\Big|_{\underline{\eta}=\underline{\eta}_k} \ ,$$

is identical to the matrix $\mathbf{P}$ in (7). $\Delta\underline{\eta} = \underline{\eta}_{k+1} - \underline{\eta}_k$ is determined by solving the system of linear equations (9), which yields the recursion

$$\underline{\eta}_{k+1} = \underline{\eta}_k + \Delta\underline{\eta} \ .$$

Algorithm 2 $[\underline{\eta},\gamma,\Delta\gamma] \leftarrow \mathrm{Adaptation}(\underline{\eta}_{\mathrm{tmp}}, G_{\max})$

```
 1: $\underline{\eta}_0 \leftarrow \underline{\eta}_{\mathrm{tmp}}$
 2: repeat
 3:     $\Delta\underline{\eta} \leftarrow \mathrm{solve}\left(\mathbf{H}(\underline{\eta}_k,\gamma), \underline{h}(\underline{\eta}_k,\gamma)\right)$
 4:     $\underline{\eta}_{k+1} \leftarrow \underline{\eta}_k + \Delta\underline{\eta}$
 5: until $k+1 = k_{\max}$ or $\Delta\underline{\eta} \rightarrow \underline{0}$
 6: if $\Delta\underline{\eta} \rightarrow 0$ then        // Newton method converged
 7:     $\Delta\gamma \leftarrow \mathrm{Increase}(\Delta\gamma)$
 8:     $\underline{\eta} \leftarrow \mathrm{StructuralAdaptation}(\underline{\eta}_{k+1}, G_{\max})$
 9: else
10:     $\underline{\eta} \leftarrow \underline{\eta}_{\mathrm{tmp}}$
11:     $\gamma \leftarrow \gamma - \Delta\gamma$
12:     $\Delta\gamma \leftarrow \mathrm{Decrease}(\Delta\gamma)$
13: end if
```

This recursion is initialized with $\underline{\eta}_0 = \underline{\eta}_{\mathrm{tmp}}$ (obtained at line 7 of Algorithm 1). In cases where this initial value is close to the true parameter vector, the method quickly converges, which can be detected by $\Delta\underline{\eta} \rightarrow \underline{0}$.

Algorithm 2 summarizes the correction method for $\underline{\eta}$, γ, and $\Delta\gamma$. Again, the system of linear equations in line 3 can be solved efficiently using LU factorization. In addition to controlling the convergence of the Newton approach, adapting $\underline{\eta}$ is aborted after a maximum number of steps $k_{\max}$ (see line 5). The structural adaptation performed in line 8 is described in detail in the following section.

B. Structural Adaptation

Performing the correction step does not guarantee that the maximum deviation $G_{\max}$ is maintained. This is especially the case when new modes emerge due to gradually incorporating the true Gaussian mixture. Here, the current number of components of the reduced mixture may not suffice to capture this structural change.

1) Normalized Distance Measure: For enabling a scale-invariant check of the deviation between $\tilde{f}(x,\gamma)$ and $f(x,\underline{\eta})$, the *normalized distance measure*

$$G_N(\underline{\eta},\gamma) = \frac{\int_{\mathbb{R}}\left(\tilde{f}(x,\gamma) - f(x,\underline{\eta})\right)^2 \mathrm{d}x}{\int_{\mathbb{R}}\tilde{f}(x,\gamma)^2\,\mathrm{d}x + \int_{\mathbb{R}} f(x,\underline{\eta})^2\,\mathrm{d}x} \qquad (10)$$

is employed. Compared to the distance (4), this measure is more convenient for specifying limits on the allowed deviation as it ranges between zero and one [13].

2) Component Splitting: Once $G_N(\underline{\eta},\gamma)$ is larger than $G_{\max}$, the progression is stopped and the number of components is increased. A straightforward way to introduce new mixture components is to split existing ones. In doing so, the most critical component, i.e., the component that is mainly responsible for the deviation, has to be identified by evaluating L individual distances

$$G_i(\underline{\eta},\gamma) = \int_{\mathbb{R}}\left(\tilde{f}(x,\gamma) - f(x,\underline{\eta})\right)^2 \cdot f_i(x,\underline{\eta}_i)\,\mathrm{d}x \ ,$$

where $i = 1, \dots, L$ and $f_i(x, \underline{\eta}_i) = \omega_i^2 \cdot \mathcal{N}(x; \mu_i, \sigma_i^2)$. These individual distances can be evaluated in closed form and the component with maximum distance is selected for splitting.

Several possibility arise for performing a split. They differ, e.g., in number of new components or the parameters of the new components. Simply reproducing the original component is not sufficient since the symmetry has to be broken to facilitate approximating the critical region of the true Gaussian mixture in different ways [18].

In this paper, splitting a component into two new Gaussians is used, since for two Gaussians a moment-preserving replacement can be easily guaranteed [11]. Therefore, a component $\omega^2 \cdot \mathcal{N}(x; \mu, \sigma^2)$ is replaced by $\omega_1^2 \cdot \mathcal{N}(x; \mu_1, \sigma_1^2)$ and $\omega_2^2 \cdot \mathcal{N}(x; \mu_2, \sigma_2^2)$, where

$$\begin{aligned} \omega^2 &= \omega_1^2 + \omega_2^2 \ , \\ \mu &= \bar{\omega}_1^2 \mu_1 + \bar{\omega}_2^2 \mu_2 \ , \\ \sigma^2 &= \bar{\omega}_1^2 \sigma_1^2 + \bar{\omega}_2^2 \sigma_2^2 + \bar{\omega}_1^2 \bar{\omega}_2^2 \left(\mu_1 - \mu_2\right)^2 \ , \end{aligned}$$

and $\bar{\omega}_1^2 = \omega_1^2 / (\omega_1^2 + \omega_2^2)$, $\bar{\omega}_2^2 = \omega_2^2 / (\omega_1^2 + \omega_2^2)$. Throughout the simulations the parameters

$$\begin{array}{lll} \omega_1^2 = 0.5\omega^2 & \mu_1 = 0.5\sigma + \mu & \sigma_1^2 = 0.75\sigma^2 \\ \omega_2^2 = 0.5\omega^2 & \mu_2 = -0.5\sigma + \mu & \sigma_2^2 = 0.75\sigma^2 \end{array}$$

are used.

3) Component Deletion: During the progression it also occurs that components of the reduced mixture become negligible and thus, contribute almost nothing to the approximation of $\tilde{f}(x, \gamma)$. These components can be identified by the ratio ω_i^2 / σ_i^2 being close to zero. Deleting them reduces the complexity of the reduced Gaussian mixture, which in turn avoids overfitting effects. Furthermore, in cases where a maximum number of components is specified by the user, deleting components facilitates splitting operations, especially when the current number of components in $f(x, \underline{\eta})$ is close to the maximum.

4) Additional Correction Step: At first, structural adaptations by performing splitting and deletion of mixture components introduces an additional error. This error can be reduced by reapplying the Newton approach derived in Section IV-A.

V. Simulation Results

For demonstrating the effectiveness of the proposed progressive Gaussian mixture reduction (PGMR) algorithm, two different simulations are conducted. First, the effect of the deviation bound $G_{\max}$ on the reduction quality is highlighted. Additionally, PGMR is compared to state-of-the-art reduction methods by means of reducing randomly generated Gaussian mixtures. For improved readability, all deviation values and bounds are multiplied by a factor 100.

A. Deviation Bound

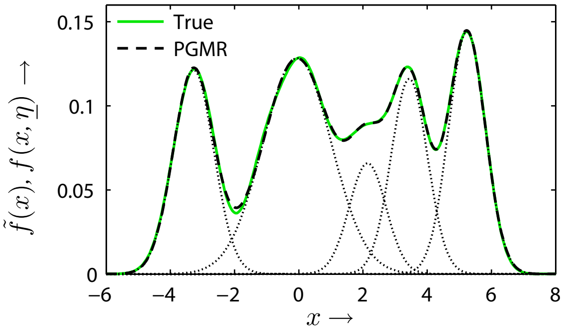

Figure 1. True Gaussian mixture (green, solid) and reduced Gaussian mixture (black, dashed) consisting of 5 components (black, dotted).

The true Gaussian mixture $\tilde{f}(x)$ consisting of $M = 10$ components, where the single Gaussians have weighting coefficients, means, and standard deviations according to

$$\begin{aligned} \omega &= [0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1] \ , \\ \mu &= [-3.5 \ -3 \ -1 \ 0 \ 0.5 \ 2 \ 3 \ 3.5 \ 5 \ 5.5] \ , \\ \sigma &= [0.6 \ 0.6 \ 0.6 \ 0.6 \ 0.7 \ 0.7 \ 1 \ 0.5 \ 0.5 \ 0.5] \ , \end{aligned}$$

is reduced by PGMR with varying deviation bound $G_{\max} \in \{0.5, 0.75, 2, 4\}$. In Table I, the used number of components L as well as the deviation between the true Gaussian mixture and its reduced version are listed. The normalized distance measure (10) is used for quantifying the deviation.

By increasing the maximum deviation value $G_{\max}$, the number of used components decreases as expected. This comes along with a reduced computation time since less structural adaptation operations have to be performed for more relaxed deviation limits. In Fig. 1, the true Gaussian mixture (red, solid) is depicted together with the reduced Gaussian mixture (blue, dashed) for $G_{\max} = 2$. Furthermore, the individual Gaussian components of $f(x, \underline{\eta})$ are also shown. Considering the five modes of the true mixture, one might expect that using also five mixture components would result into a precise approximation. This is almost true except of the second mode at $x \approx 0$, which cannot be fitted appropriately by a single Gaussian. Thus, a considerable improvement of the reduction quality is gained for $L = 6$. At this point, spending more components only gives marginal quality improvements.

Table I
Number of components and reduction quality for different maximum deviation values $G_{\max}$.

$G_{\max}$	0.5	0.75	2	4
Number of components	7	6	5	4
Deviation $G_N(\underline{\eta}, \gamma)$	0.217	0.234	0.917	3.228

As the last row in Table I indicates, the bound $G_{\max}$ is always maintained. The bound can be violated, when in addition to $G_{\max}$ a limit on L is imposed. For example, not allowing more than $L = 6$ for the bound $G_{\max} = 0.5$, results in a deviation $G_N(\underline{\eta}, \gamma) = 0.896$, which indeed is larger than the bound. However, in many practical applications, keeping L below a given maximum number of components is of paramount importance for assuring worst-case computation time. Thus, with PGMR the user can set preferences on either a maximum deviation or a maximum number of components.

B. Comparison with State-of-the-art Methods

Now, the PGMR algorithm is compared with two established reduction methods. Williams' reduction algorithm employs the squared integral measure (4) to evaluate at each reduction step which particular deletion of a component or merge of a pair of components yields the smallest dissimilarity from the true Gaussian mixture [10]. The second method is a local reduction algorithm proposed by M. West [8]. Here, at each reduction step the component with the smallest weight is merged with its nearest neighbor, where the weighted Mahalanobis distance [19] is used for determining the neighboring component.

For comparison purposes, the true Gaussian mixture consists of $M \in \{40, 80, 120, 160, 200\}$ components, where the parameters are drawn i.i.d. from uniform distributions over the intervals

$$\begin{aligned}\omega &\in [0.05, 0.5]\ ,\\ \mu &\in [0, 3]\ ,\\ \sigma &\in [0.09, 0.5]\ .\end{aligned}$$

For each number of components M, 20 Monte Carlo simulation runs are performed, where all reduction algorithms are forced to use $L = 10$ or less components. For PGMR the bound $G_{\max} = 1$ is selected.

Table II
DEVIATION AND TIME CONSUMPTION OF THE THREE METHODS FOR REDUCING A MIXTURE WITH A VARYING NUMBER OF COMPONENTS M

	Normalized Deviation G_N			Computation time in s		
M	PGMR	Williams	West	PGMR	Williams	West
40	0.620	0.861	5.533	4.171	1.370	0.009
80	0.545	0.949	4.796	3.871	5.588	0.017
120	0.548	0.962	3.944	5.424	14.531	0.028
160	0.589	0.898	4.174	3.957	30.627	0.036
200	0.638	1.029	3.806	4.777	57.923	0.044

In Table II, the average deviations and average computation times for all M are listed.[3] PGMR provides the best average deviation for each M. This is notable, as PGMR on average uses between four and five components, while Williams' and West's algorithm, always result in reduced mixtures with 10 components. In Fig. 2, the reduction results for a true mixture with $M = 200$ components are depicted. The corresponding progression is illustrated in Fig. 3. Thanks to the progressive processing, PGMR is capable of almost exactly capturing the shape of the true mixture, while Williams' algorithm fails in accurately approximating details of the shape as it can be clearly seen for the second mode. The grossness of West's method is even more significant, as it does not incorporate any shape information when merging components. However, in contrast to PGMR, both algorithms preserve the mean and variance of the original mixture (see Section VI).

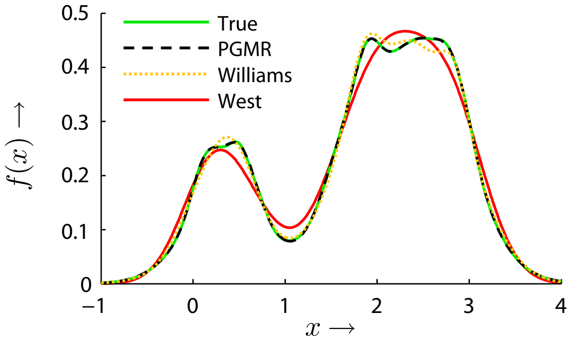

Figure 2. A 200 component Gaussian mixture (green, solid) and its reduced versions resulting from PGMR (black, dashed), Williams' algorithm (orange, dotted), and West's algorithm (red, solid).

The right hand columns of Table II indicate that the computation time of PGMR is approximately constant for all M, while it grows with M for the other algorithms. In case of West's algorithm, this growth is negligible, as the algorithm is generally computationally very efficient due to its local reduction characteristic.

The constant computation time of PGMR originates from the different way a mixture is reduced. Regardless of the number of components of $\tilde{f}(x)$, PGMR always starts with a single Gaussian. The computationally most expensive operations of PGMR are structural adaptations, i.e., extending the number of components. However, these operations are only performed if required and handling many components in $f(x, \underline{\eta})$ is systematically avoided. On the other side, Williams' and West's algorithm start the reduction with the complete original mixture and perform a greedy search involving all remaining components for identifying the next merging operation in each reduction step. This search basically has a quadratic complexity in case of Williams' algorithm[4] and a linear complexity for West's method.

VI. CONCLUSIONS AND FUTURE WORK

To achieve the goal of replacing a complex Gaussian mixture by one consisting of a minimal number of components with respect to a desired maximum reduction error, the classical approach of successively merging components is often inappropriate. In comparison, the novel Gaussian mixture reduction algorithm introduced in this paper provides significantly better reduction results. It was demonstrated that gradually incorporating the effect of the complex true Gaussian mixture during the progression facilitates accurate approximations as the reduced Gaussian mixture can be constantly adapted and, if required, its approximation capability at specific regions can be improved by adding new components. Compared to local reduction methods, the resulting reduced mixture is very close to the original since adapting the approximation is accomplished by a global optimization. Compared

[3] The computation times depend on a Matlab 7.5 implementation running on a PC with an Intel Core2 Duo 2.4 GHz processor.

[4] As suggested in [10], our implementation makes use of the fact that all terms for calculating the measure (4) can be pre-computed and stored. Because only a few terms change between several reduction steps, partially updating the stored terms leads to significant computational savings.

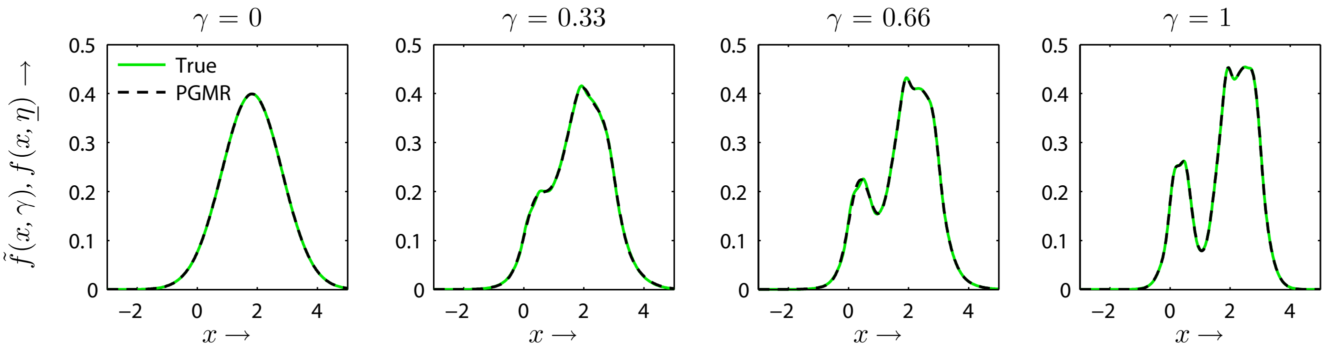

Figure 3. Progression for the Gaussian mixture depicted in Fig. 2. For several values of γ the parameterized mixture $\tilde{f}(x,\gamma)$ and the corresponding reduced mixture $f(x,\underline{\eta})$ are shown.

to other global approaches, redundancy in the original mixture is better exploited. Thus, the used number of components and the computational demand is significantly smaller.

Future work includes the extension to multivariate Gaussian mixtures. For the most part, this extension is straightforward, since all terms can still be expressed analytically. However, splitting components becomes more challenging as the degree of freedom in the splitting direction drastically grows.

Alternatively to splitting it is intended to provide adding of completely new components at regions of high deviation. By directly adding components, the problem of heuristically determining the splitting direction can be avoided and emerging deviations can be resolved in situ. However, accurately determining regions of high deviations is difficult, as this problem is similar to finding modes in a Gaussian mixture, which is well-known as a demanding optimization problem [20].

Thanks to the shape approximation of the proposed approach, the deviation between the moments of the original mixture and the reduced mixture is marginal. However, in order to provide an exact preservation of mean and variance, it is intended to incorporate the true moments as further constraints. By applying the Lagrangian multiplier approach, these constraints can then be maintained during the progression.

VII. Acknowledgements

This work was partially supported by the German Research Foundation (DFG) within the Research Training Group GRK 1194 “Self-organizing Sensor-Actuator-Networks”.

Appendix

A. Analytical Expression for $\mathbf{P}(\underline{\eta},\gamma)$

At first, the solution of the first summand in (7) is given, which is

$$\mathbf{P}'(\underline{\eta}) = \int_{\mathbb{R}} \underline{F}(x,\underline{\eta})\underline{F}(x,\underline{\eta})^{\mathrm{T}}\,\mathrm{d}x = \begin{bmatrix} \mathbf{P}^{(1,1)} & \mathbf{P}^{(1,2)} & \cdots & \mathbf{P}^{(1,L)} \\ \mathbf{P}^{(2,1)} & \mathbf{P}^{(2,2)} & \cdots & \mathbf{P}^{(2,L)} \\ \vdots & \vdots & & \vdots \\ \mathbf{P}^{(L,1)} & \mathbf{P}^{(L,2)} & \cdots & \mathbf{P}^{(L,L)} \end{bmatrix} .$$

The individual 3×3 block matrices $\mathbf{P}^{(i,j)}$ for $i = 1,\ldots,L$ and $j = 1,\ldots,L$ are

$$\mathbf{P}^{(i,j)} = \int_{\mathbb{R}} \frac{\partial f_i(x,\underline{\eta}_i)}{\partial \underline{\eta}_i} \left(\frac{\partial f_j(x,\underline{\eta}_j)}{\partial \underline{\eta}_j}\right)^{\mathrm{T}} \mathrm{d}x = \omega_i \cdot \omega_j \cdot \mathcal{N}\big(\mu_i;\mu_j,\sigma_{i,j}^2\big) \cdot \begin{bmatrix} P_{1,1}^{(i,j)} & P_{1,2}^{(i,j)} & P_{1,3}^{(i,j)} \\ P_{2,1}^{(i,j)} & P_{2,2}^{(i,j)} & P_{2,3}^{(i,j)} \\ P_{3,1}^{(i,j)} & P_{3,2}^{(i,j)} & P_{3,3}^{(i,j)} \end{bmatrix} ,$$

with $f_i(x,\underline{\eta}_i) := \omega_i^2 \cdot \mathcal{N}(x;\mu_i,\sigma_i^2)$, $\sigma_{i,j}^2 := \sigma_i^2 + \sigma_j^2$ and

$$\begin{aligned}
P_{1,1}^{(i,j)} &= 4 \ , \\
P_{1,2}^{(i,j)} &= 2\omega_j \tfrac{\mu_i - \mu_j}{\sigma_{i,j}^2} \ , \\
P_{1,3}^{(i,j)} &= 2\omega_j\sigma_j \tfrac{(\mu_i-\mu_j)^2 - \sigma_{i,j}^2}{\sigma_{i,j}^4} \ , \\
P_{2,1}^{(i,j)} &= 2\omega_i \tfrac{\mu_j - \mu_i}{\sigma_{i,j}^2} \ , \\
P_{2,2}^{(i,j)} &= \omega_i\omega_j \tfrac{\sigma_{i,j}^2 - (\mu_i-\mu_j)^2}{\sigma_{i,j}^4} \ , \\
P_{2,3}^{(i,j)} &= \omega_i\omega_j\sigma_j \tfrac{(\mu_j-\mu_i)\cdot\left((\mu_i-\mu_j)^2 - 3\sigma_{i,j}^2\right)}{\sigma_{i,j}^6} \ , \\
P_{3,1}^{(i,j)} &= 2\omega_i\sigma_i \tfrac{(\mu_i-\mu_j)^2 - \sigma_{i,j}^2}{\sigma_{i,j}^4} \ , \\
P_{3,2}^{(i,j)} &= \omega_i\omega_j\sigma_i \tfrac{(\mu_i-\mu_j)\cdot\left((\mu_i-\mu_j)^2 - 3\sigma_{i,j}^2\right)}{\sigma_{i,j}^6} \ , \\
P_{3,3}^{(i,j)} &= \omega_i\omega_j\sigma_i\sigma_j \tfrac{(\mu_i-\mu_j)^4 + 3\sigma_{i,j}^2\left(\sigma_{i,j}^2 - 2(\mu_i-\mu_j)^2\right)}{\sigma_{i,j}^8} \ .
\end{aligned}$$

The expression for $\Delta\mathbf{P}(\underline{\eta},\gamma)$ is given by

$$\Delta\mathbf{P}(\underline{\eta},\gamma) = \int_{\mathbb{R}} \left(f(x,\underline{\eta}) - \tilde{f}(x,\gamma)\right) \mathbf{M}(x,\underline{\eta})\,\mathrm{d}x \ , \quad (11)$$

where

$$\mathbf{M}(x,\underline{\eta}) = \frac{\partial^2 f(x,\underline{\eta})}{\partial\underline{\eta}\,\partial\underline{\eta}^{\mathrm{T}}} = \begin{bmatrix} \mathbf{M}^{(1)} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}^{(2)} & & \mathbf{0} \\ \vdots & & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{M}^{(L)} \end{bmatrix} ,$$

with 3×3 block matrices

$$\mathbf{M}^{(i)} = 2 \cdot f_i(x, \underline{\eta}_i) \cdot$$
$$\begin{bmatrix} \frac{1}{\omega_i^2} & \frac{x-\mu_i}{\omega_i \sigma_i^2} & \frac{(x-\mu_i)^2-\sigma_i^2}{\omega_i \sigma_i^3} \\ \frac{x-\mu_i}{\omega_i \sigma_i^2} & \frac{(x-\mu_i)^2-\sigma_i^2}{2\sigma_i^4} & \frac{(x-\mu_i)^3-3\sigma_i^2(x-\mu_i)}{2\sigma_i^5} \\ \frac{(x-\mu_i)^2-\sigma_i^2}{\omega_i \sigma_i^3} & \frac{(x-\mu_i)^3-3\sigma_i^2(x-\mu_i)}{2\sigma_i^5} & \frac{(x-\mu_i)^4-5\sigma_i^2(x-\mu_i)^2+2\sigma_i^4}{2\sigma_i^6} \end{bmatrix} .$$

Thus, solving (11) corresponds to calculating the zeroth up to the forth moment of the Gaussian mixtures $f(x, \underline{\eta}) \cdot f_i(x, \underline{\eta}_i)$ and $\tilde{f}(x, \gamma) \cdot f_i(x, \underline{\eta}_i)$, similarly as shown in the following for $\underline{b}(\underline{\eta}, \gamma)$.

B. Analytical Expression for $\underline{b}(\underline{\eta}, \gamma)$

The expression for the vector

$$\underline{b}(\underline{\eta}, \gamma) = \int_{\mathbb{R}} \underline{F}(x, \underline{\eta}) \frac{\partial \tilde{f}(x, \gamma)}{\partial \gamma} \, \mathrm{d}x$$

consists of the vector of partial derivatives $\underline{F}(x, \underline{\eta})$, which comprises the elements

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

Together with the scalar function

$$\frac{\partial \tilde{f}(x, \gamma)}{\partial \gamma} = \tilde{f}(x) - \hat{f}(x) \ ,$$

the i-th element of $\underline{b}(\underline{\eta}, \gamma)$ is given by

$$\underline{b}_i(\underline{\eta}_i, \gamma) = \int_{\mathbb{R}} \left(\tilde{f}(x) - \hat{f}(x) \right) f_i(x, \underline{\eta}_i) \begin{bmatrix} \frac{2}{\omega_i} \\ \frac{x-\mu_i}{\sigma_i^2} \\ \frac{(x-\mu_i)^2-\sigma_i^2}{\sigma_i^3} \end{bmatrix} \mathrm{d}x \quad (12)$$
$$= \underbrace{\begin{bmatrix} \frac{2}{\omega_i} & 0 & 0 \\ -\frac{\mu_i}{\sigma_i^2} & \frac{1}{\sigma_i^2} & 0 \\ \frac{\mu_i^2-\sigma_i^2}{\sigma_i^3} & -\frac{2\mu_i}{\sigma_i^3} & \frac{1}{\sigma_i^3} \end{bmatrix}}_{=:\mathbf{B}_i} \cdot \begin{bmatrix} \mathrm{E}_{\tilde{\mathcal{F}}^i}\{1\} - \mathrm{E}_{\hat{\mathcal{F}}^i}\{1\} \\ \mathrm{E}_{\tilde{\mathcal{F}}^i}\{x\} - \mathrm{E}_{\hat{\mathcal{F}}^i}\{x\} \\ \mathrm{E}_{\tilde{\mathcal{F}}^i}\{x^2\} - \mathrm{E}_{\hat{\mathcal{F}}^i}\{x^2\} \end{bmatrix} .$$

Thus, $\underline{b}(\underline{\eta}, \gamma)$ can be efficiently calculated using matrix-vector calculus, where the vector comprises the zeroth up to the second moment of the densities $\tilde{\mathcal{F}}^i(x) = \tilde{f}(x) \cdot f_i(x, \underline{\eta}_i)$ and $\hat{\mathcal{F}}^i(x) = \hat{f}(x) \cdot f_i(x, \underline{\eta}_i)$. All moments can be determined in closed form and $\mathrm{E}_{\tilde{\mathcal{F}}^i}$ as well as $\mathrm{E}_{\hat{\mathcal{F}}^i}$ are the corresponding expected value operators.

C. Analytical Expression for $\underline{h}(\underline{\eta}, \gamma)$

The gradient $\underline{h}(\underline{\eta}, \gamma)$ of the squared integral distance measure comprises the elements

$$\underline{h}_i(\underline{\eta}, \gamma) = -\int_{\mathbb{R}} \left(\tilde{f}(x, \gamma) - f(x, \underline{\eta}) \right) \cdot \frac{\partial f(x, \underline{\eta})}{\partial \underline{\eta}_i} \, \mathrm{d}x \ , \quad (13)$$

for $i = 1, 2, \ldots, L$, which are quite similar to (12). Hence, (13) can also be written in matrix-vector notation

$$\underline{h}_i(\underline{\eta}, \gamma) = \mathbf{B}_i \cdot \begin{bmatrix} \mathrm{E}_{\mathcal{F}^i}\{1\} - \mathrm{E}_{\tilde{\mathcal{F}}^i}\{1\} \\ \mathrm{E}_{\mathcal{F}^i}\{x\} - \mathrm{E}_{\tilde{\mathcal{F}}^i}\{x\} \\ \mathrm{E}_{\mathcal{F}^i}\{x^2\} - \mathrm{E}_{\tilde{\mathcal{F}}^i}\{x^2\} \end{bmatrix} \ ,$$

with $\mathcal{F}^i(x) = f(x, \underline{\eta}) \cdot f_i(x, \underline{\eta}_i)$, $\tilde{\mathcal{F}}^i(x) = \tilde{f}(x, \gamma) \cdot f_i(x, \underline{\eta}_i)$.

References

[1] Y. Bar-Shalom and X.-R. Li, *Multitarget-multisensor Tracking: Principles and Techniques*. YBS Publishing, Storrs, CT, 1995.

[2] V. Hasselblad, "Estimation of Parameters for a Mixture of Normal Distributions," *Technometrics*, vol. 8, no. 8, pp. 431–444, Aug. 1966.

[3] E. Parzen, "On Estimation of a Probability Density Function and Mode," *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1065–1076, Sep. 1962.

[4] D. L. Alspach and H. W. Sorenson, "Nonlinear Bayesian Estimation using Gaussian Sum Approximation," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 439–448, Aug. 1972.

[5] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active Learning with Statistical Models," *Journal of Artificial Intelligence Research*, vol. 4, pp. 129–145, 1996.

[6] D. J. Salmond, "Mixture reduction algorithms for target tracking," in *IEE Colloquium on State Estimation in Aerospace and Tracking Applications*, London, UK, Dec. 1989, pp. 7/1–7/4.

[7] ——, "Mixture reduction algorithms for target tracking in clutter," in *Proceedings of SPIE Signal and Data Processing of Small Targets*, vol. 1305, Oct. 1990, pp. 434–445.

[8] M. West, "Approximating Posterior Distributions by Mixtures," *Journal of the Royal Statistical Society: Series B*, vol. 55, no. 2, pp. 409–422, 1993.

[9] O. C. Schrempf, O. Feiermann, and U. D. Hanebeck, "Optimal Mixture Reduction of the Product of Mixtures," in *Proceedings of the 8th International Conference on Information Fusion (Fusion 2005)*, vol. 1, Philadelphia, Pennsylvania, Jul. 2005, pp. 85–92.

[10] J. L. Williams and P. S. Maybeck, "Cost-Function-Based Gaussian Mixture Reduction for Target Tracking," in *Proceedings of the Sixth International Conference of Information Fusion*, vol. 2, 2003, pp. 1047–1054.

[11] A. R. Runnalls, "Kullback-Leibler Approach to Gaussian Mixture Reduction," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 3, pp. 989–999, Jul. 2007.

[12] E. L. Allgower and K. Georg, *Numerical Continuation Methods: An Introduction*, ser. Springer Series in Computational Mathematics. Springer-Verlag, 1990.

[13] U. D. Hanebeck, K. Briechle, and A. Rauh, "Progressive Bayes: A New Framework for Nonlinear State Estimation," in *Proceedings of SPIE, AeroSense Symposium*, vol. 5099, Orlando, Florida, May 2003, pp. 256–267.

[14] D. E. Catlin, *Estimation, Control, and the Discrete Kalman Filter*, 1st ed., ser. Applied Mathematical Sciences. New York: Springer-Verlag, 1989, vol. 71.

[15] A. J. Izenman, "Recent developments in nonparametric density estimation," *Journal of the American Statistical Association*, vol. 86, no. 413, pp. 205–224, Mar. 1991.

[16] S. Kullback and R. A. Leibler, "On Information and Sufficiency," *Annals of Mathematical Statistics*, vol. 22, no. 2, pp. 79–86, 1951.

[17] O. C. Schrempf, D. Brunn, and U. D. Hanebeck, "Dirac Mixture Density Approximation Based on Minimization of the Weighted Cramér-von Mises Distance," in *Proceedings of the 2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2006)*, Heidelberg, Germany, Sep. 2006, pp. 512–517.

[18] M. J. Beal, "Variational Algorithms for Approximate Bayesian Inference," Ph.D. dissertation, Gatsby Computational Neuroscience Unit, University College London, 2003.

[19] P. Mahalanobis, "On the generalised distance in statistics," *Proceedings of the National Institute of Science of India*, vol. 12, no. 1, pp. 49–55, 1936.

[20] M. A. Carreira-Perpinán, "Mode-Finding for Mixtures of Gaussian Distribution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1318–1323, Nov. 2000.