Superficial Gaussian Mixture Reduction

Marco F. Huber†, Peter Krauthausen‡, and Uwe D. Hanebeck‡

†AGT Group (R&D) GmbH, Darmstadt, Germany,
‡Intelligent Sensor-Actuator-Systems Laboratory (ISAS),
Institute for Anthropomatics, Karlsruhe Institute of Technology, Karlsruhe, Germany.
Marco.Huber@ieee.org, Peter.Krauthausen@kit.edu, Uwe.Hanebeck@ieee.org

Abstract: Many information fusion tasks involve the processing of Gaussian mixtures with simple underlying shape, but many components. This paper addresses the problem of reducing the number of components, allowing for faster density processing. The proposed approach is based on identifying components irrelevant for the overall density's shape by means of the curvature of the density's surface. The key idea is to minimize an upper bound of the curvature while maintaining a low global reduction error by optimizing the weights of the original Gaussian mixture only. The mixture is reduced by assigning zero weights to reducible components. The main advantages are an alleviation of the model selection problem, as the number of components is chosen by the algorithm automatically, the derivation of simple curvature-based penalty terms, and an easy, efficient implementation. A series of experiments shows the approach to provide a good trade-off between quality and sparsity.

1 Introduction

Gaussian mixtures as weighted sums of Gaussian densities are an often used function system in various information fusion applications, such as Bayesian filtering [AS72, HBH06], multi-target tracking [BSL95], density estimation [Sil98], or machine learning [CGJ96], just to name a few. Since the space of Gaussian densities forms a complete basis system, Gaussian mixtures can approximate every function with arbitrary accuracy [MS96]. Unfortunately, the number of Gaussian components of a Gaussian mixture tends to grow exponentially when processed recursively. To control this growth and thus, to bound computational and memory demands, Gaussian mixture reduction algorithms have to be applied continually.

In recent years, many reduction algorithms have been proposed. Most of them employ a top-down approach: Two or more components of the Gaussian mixture with strong similarity are merged or components that do not contribute much to the mixture are deleted. These operations are performed recursively in a greedy fashion. The reduction stops as soon as a user-defined threshold on the number of components is reached. To quantify the similarity between components, local distance measures such as the Mahalanobis distance as in [Wes93, Sal90] or global distance measures such as the integral squared distance

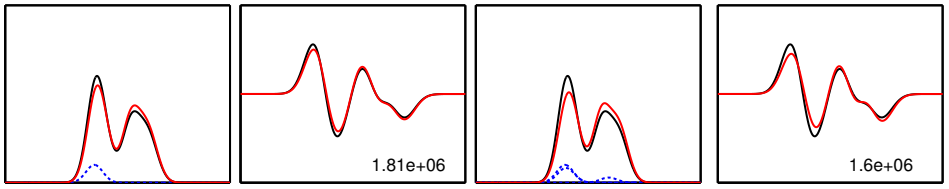

Figure 1: Gaussian mixture with 20 components (black) if one component (blue) is removed (a) and the pointwise second derivative of the resulting mixture (red) (b). The same mixture reduced to the 17 components (c) and the point-wise second derivative for the reduced density (red) (d).

as in [WM03, SH09] and the Kullback-Leibler divergence as in [Run07] are used. Compared to local approaches, the deviation from the original Gaussian mixture when applying global optimization is typically much lower, at the expense of a higher computational load. Merging components is performed in a moment-preserving way, i.e., the mean and the covariance of the mixture remain unchanged. Independent of the used distance measure, top-down approaches have two severe drawbacks. First, merging and deleting components is performed greedily without considering the global effect in successive reduction steps. The second drawback results from the user-defined threshold on the final number of components. From a statistical point of view, the choice of this threshold corresponds to a model selection. The reduction algorithm has to reduce the mixture even if the resulting model is inappropriate, which for example leads to oversmoothed modes. To overcome these drawbacks, a bottom-up approach named progressive Gaussian mixture reduction (PGMR) has recently been proposed [HH08]. Here, a Gaussian mixture is successively constructed to approximate the original mixture with far less components. Starting with a single Gaussian, new components are added at regions of strong deviation. Thus, the number of components is chosen automatically by the algorithm, which avoids the bias introduced by a predefined threshold. Unfortunately, PGMR is only able to construct mixtures with axis-aligned Gaussian components. It further requires a complex implementation.

In this paper, a global optimization approach named *superficial Gaussian mixture reduction* (SGMR) is proposed, which is based on minimizing the curvature of the reduced Gaussian mixture while keeping the integral squared distance (ISD) low. This top-down approach allows for the reduction of arbitrary Gaussian mixtures while preserving the mean. As illustrated in Fig. 1, the curvature captures the roughness in densities with an overall simple shape, i.e., with clear modes. Using the curvature it is possible to identify similar components globally and remove these components from the density. Carrying the idea that similar components may be dropped a step further, the trade-off between curvature and approximation error is minimized by merely optimizing weights, thus assigning zero weights to reduced components. This approach is computationally feasible and allows a simple and efficient implementation based on standard quadratic program (QP) solvers. Additionally, this weight-only optimization alleviates the model selection problem, as the final number of components is automatically derived from setting the trade-off between error and roughness. This hyperparameter may be automatically optimized too.

The next section gives a brief introduction to the Gaussian mixture reduction problem. The rest of the paper is structured as follows: The theoretical background of the proposed

SGMR algorithm is described in Sec. 3. Here, an upper bound of the curvature is derived, which acts as a roughness penalty. In Sec. 4, limitations of the proposed approach are discussed. By means of numerical experiments, SGMR is compared to state-of-the-art mixture reduction algorithms in Sec. 5.

2 Problem Statement

Given is a random vector $\underline{x} \in \mathbb{R}^{\mathrm{N}}$ with probability density function $\tilde{f}(\underline{x})$. This density function is assumed to be represented as a Gaussian mixture

$$\tilde{f}(\underline{x}) = \underline{\tilde{\alpha}}^{\mathrm{T}} \underline{f}(\underline{x}) = \sum_{i=1}^{L} \tilde{\alpha}_i \cdot \mathcal{N}\left(\underline{x}; \underline{\tilde{\mu}}_i, \tilde{\mathbf{\Sigma}}_i\right) , \tag{1}$$

with $\underline{\tilde{\alpha}} = [\tilde{\alpha}_1 \dots \tilde{\alpha}_L]^{\mathrm{T}}$ and $\underline{f}(\underline{x}) = [\mathcal{N}(\underline{x}; \underline{\tilde{\mu}}_1, \tilde{\mathbf{\Sigma}}_1) \dots \mathcal{N}(\underline{x}; \underline{\tilde{\mu}}_L, \tilde{\mathbf{\Sigma}}_L)]^{\mathrm{T}}$. Here, L is the number of mixture components, $\tilde{\alpha}_i \geq 0$, $i = 1 \dots L$ are weights that sum up to one, and $\mathcal{N}\left(\underline{x}; \underline{\mu}, \mathbf{\Sigma}\right)$ is a Gaussian density with mean vector $\underline{\mu}$ and covariance matrix $\mathbf{\Sigma}$.

In typical estimation tasks such as recursive filtering, the number of mixture components L grows exponentially with the number of processing steps. To keep this growth bounded, it is necessary to compute a reduced Gaussian mixture with the number of components being significantly lower than L. Additionally, the deviation between the reduced mixture and the original mixture has to be as small as possible.

To solve this apparent conflict of goals, one can make use of the following observation: Although the number of components is large, the shape of the density functions in typical estimation tasks, is often rather simple, e.g. in multi-target tracking [BSL95] the number of modes is low and the smoothness is high. Especially in recursive filtering tasks, the density function of the hidden state often becomes unimodal or even Gaussian-like after a transient phase. Thus, a Gaussian mixture with a considerably smaller number of components can typically be found without causing a strong deviation from the original mixture.

In the following, reducing the number of components is achieved by adapting the weights $\tilde{\alpha}_j$ only, while the remaining parameters of the mixture, i.e., mean vectors and covariance matrices, remain untouched. The Gaussian mixture to be adapted is given by

$$f_{\underline{\alpha}}(\underline{x}) = \underline{\alpha}^{\mathrm{T}} \underline{f}(\underline{x}) = \sum_{j=1}^{L} \alpha_j \cdot \mathcal{N}\left(\underline{x}; \underline{\tilde{\mu}}_j, \tilde{\mathbf{\Sigma}}_j\right) , \tag{2}$$

with the vector of weights $\underline{\alpha} = [\alpha_1 \dots \alpha_L]^{\mathrm{T}}$. The proposed mixture reduction method will assign weights close to zero to redundant mixture components. These components can be easily removed in a subsequent processing step. The remaining components, however, compensate this loss in representing the shape of $\tilde{f}(\underline{x})$ just by weight adaption. For the sake of brevity and clarity, only uni- and bivariate mixtures are considered from now on.

3 Superficial Gaussian Mixture Reduction

The Gaussian mixture reduction problem is formulated as a weight optimization problem

$$\begin{aligned} \min_{\underline{\alpha}} \quad & D(\tilde{f}, f_{\underline{\alpha}}) + \lambda\, R(f_{\underline{\alpha}}) \\ \text{s.t.} \quad & \underline{1}^{\mathrm{T}}\, \underline{\alpha} = 1\,, \\ & \underline{0} \preceq \underline{\alpha}\,, \\ & \sum_{i=1:L} \tilde{\underline{\mu}}_i (\alpha_i - \tilde{\alpha}_i) = \underline{0}\,, \end{aligned} \tag{3}$$

where the parameter λ governs the trade-off between a distance $D(\tilde{f}, f_{\underline{\alpha}})$ of the true density $\tilde{f}$ to its reduction $f_{\underline{\alpha}}$ and a roughness penalty $R(f_{\underline{\alpha}})$, measuring the curvature of $f_{\underline{\alpha}}$. The constraints in the optimization problem assert the integration of the probability mass to one, the positivity of the density, and that $\tilde{f}$ and $f_{\underline{\alpha}}$ have identical means.

3.1 Distance Measure

A key requirement for any reduction algorithm is that the distance of the reduced $f_{\underline{\alpha}}$ to the true density $\tilde{f}$ is small over the entire state space. Therefore, the ISD is employed and reformulated as a function of the mixture weights $\underline{\alpha}$

$$D(\tilde{f}, f_{\underline{\alpha}}) = \frac{1}{2} \int_{\mathbb{R}^2} \left(\tilde{f}(\underline{x}) - f_{\underline{\alpha}}(\underline{x}) \right)^2 \mathrm{d}\underline{x} = \underline{\alpha}^{\mathrm{T}}\, \mathbf{D}\, \underline{\alpha} - 2\, \underline{d}^{\mathrm{T}} \underline{\alpha} + c\,, \tag{4}$$

with matrix $\mathbf{D} = \int \underline{f}(\underline{x})\, \underline{f}(\underline{x})^{\mathrm{T}}\, \mathrm{d}\underline{x}$ corresponding to the self-similarity of $f_{\underline{\alpha}}$, the vector $\underline{d}^{\mathrm{T}} = \tilde{\underline{\alpha}}^{\mathrm{T}}\, \mathbf{D}$ encoding the cross-similarity between $f_{\underline{\alpha}}$ and $\tilde{f}$, as well as constant $c = \tilde{\underline{\alpha}}^{\mathrm{T}}\, \mathbf{D}\, \tilde{\underline{\alpha}}$ corresponding to the self-similarity of $\tilde{f}$. The quadratic form in (4) is obtained by expanding the binomial, exploiting the linearity, and solving the obtained integrals. Note that the same vector of Gaussians $\underline{f}(\underline{x})$ is used for both $f_{\underline{\alpha}}$ and $\tilde{f}$.

3.2 Upper bound of Curvature

The roughness of $f_{\underline{\alpha}}$ is interpreted as the curvature κ of the probability density function's surface. Since the curvature [Car76] is signed and a function of the position on the surface, a quantification in terms of the integral squared curvature (ISC) is sought. For the sake of brevity, the notation $f_m := \frac{\partial}{\partial m} f(\underline{x})$ and $f_{xy} := \frac{\partial^2}{\partial x \partial y} f(\underline{x})$ is used for the derivatives at point $\underline{x}$. For a Gaussian mixture density $f(\underline{x}) = \underline{\alpha}^{\mathrm{T}}\, \underline{f}(\underline{x})$, $f_m^{(i)} = \frac{\partial}{\partial m} \mathcal{N}\left(\underline{x}; \underline{\mu}_i, \underline{\Sigma}_i\right)$ denotes the i-th component's partial derivative w.r.t. m and $\underline{f}_m$ the vector of all components' partial derivatives. The key idea is to derive an (approximate) upper bound of the squared curvature of the mixture density function. The derivation is based on the pointwise squared curvature $\kappa(\underline{x})$ for a probability density function f, for which an upper bound $\breve{\kappa}(\underline{x})$ is determined and integrated over the entire domain of $\underline{x}$, i.e., $R_{\underline{x}} = \int_{\mathbb{R}} \breve{\kappa}(\underline{x})^2\, \mathrm{d}\underline{x}$. For the

1-D and 2-D case, the following upper bounds are used

$$\breve{\kappa}(x)^2 := \left(\underline{\alpha}^{\mathrm{T}} \underline{f}_{xx}\right)^2 , \qquad \breve{\kappa}(\underline{x})^2 := \left(\underline{\alpha}^{\mathrm{T}} \underline{f}_{xx} - 2\,\underline{\alpha}^{\mathrm{T}} \underline{f}_{x}\,\underline{\alpha}^{\mathrm{T}} \underline{f}_{y}\,\underline{\alpha}^{\mathrm{T}} \underline{f}_{xy} + \underline{\alpha}^{\mathrm{T}} \underline{f}_{yy}\right)^2 .$$

For the weight optimization, the upper bound of the curvature is formulated as a quadratic form $R(\underline{\alpha}) = \underline{\alpha}^{\mathrm{T}}\, \mathbf{R}^{\underline{x}}\, \underline{\alpha}$. The elements of $\mathbf{R}^{\underline{x}}$ may be obtained as follows.

For the 1-D case, one may simplify the upper bound of the squared curvature [RS97]

$$\int_{\mathbb{R}} \left(\underline{\alpha}^{\mathrm{T}} \underline{f}_{xx}\right)^2 \,\mathrm{d}x = \int_{\mathbb{R}} \underline{\alpha}^{\mathrm{T}} \underline{f}_{xx}\, \underline{f}_{xx}^{\mathrm{T}} \underline{\alpha} \,\mathrm{d}x$$

and use the linearity of the integral to obtain the expression for the elements of $\mathbf{R}^{\underline{x}}$

$$\mathbf{R}^{\underline{x}}_{ij} = \int_{\mathbb{R}} f^{(i)}_{xx}\, f^{(j)}_{xx} \,\mathrm{d}x\,.$$

For the 2-D case, the following approximation is used

$$\left(\underline{\alpha}^{\mathrm{T}} \underline{f}_{xx} - 2\,\underline{\alpha}^{\mathrm{T}} \underline{f}_{x}\,\underline{\alpha}^{\mathrm{T}} \underline{f}_{y}\,\underline{\alpha}^{\mathrm{T}} \underline{f}_{xy} + \underline{\alpha}^{\mathrm{T}} \underline{f}_{yy}\right)^2 \;\approx\; \left(\underline{\alpha}^{\mathrm{T}} \left[\underline{f}_{xx} - 2\,\underline{f}_{x}\,\underline{f}_{y}^{\mathrm{T}}\,\underline{f}_{xy} + \underline{f}_{yy}\right]\right)^2$$

where $\underline{\alpha}\,\underline{\alpha}^{\mathrm{T}}$ has been neglected. One obtains

$$\mathbf{R}^{\underline{x}}_{ij} = \int_{\mathbb{R}^2} \left(f^{(i)}_{xx} - 2\, f^{(i)}_{x}\, f^{(i)}_{y}\, f^{(i)}_{xy} + f^{(i)}_{yy}\right) \cdot \left(f^{(j)}_{xx} - 2\, f^{(j)}_{x}\, f^{(j)}_{y}\, f^{(j)}_{xy} + f^{(j)}_{yy}\right) \,\mathrm{d}\underline{x}\,. \quad (5)$$

For the 1-D curvature, the $\mathbf{R}^{\underline{x}}_{ij}$ may be calculated in closed form. Note for a 2-D probability density function, the curvature is not unique, as it is calculated from the minimum and maximum curvature in the principal directions at each point $\underline{x}$, which may be multiplied (Gaussian curvature) or averaged (mean curvature) [Car76]. The above upper bound of the integral squared mean curvature was obtained by dropping the denominator and positive summands. For arbitrary Gaussian mixture densities, the terms in (5) may only be calculated numerically or need to be approximated further. In the following algorithm, the property that the exact upper bound of the 1-D curvature and the approximate upper bound of the 2-D curvature of the probability density functions as well as the distance measure may be represented as quadratic forms will be exploited.

3.3 Algorithm

The overall algorithm of the SGMR comprises three parts: the *pre-processing* of the components of the quadratic forms and the hyperparameter, the weight optimization by the solution of (3) in form of a *QP* and a fast *post-optimization* of the already reduced set of weights from (3). The *pre-processing* consists of calculating the matrix $\mathbf{D}$ and vector $\underline{d}$ corresponding to the distance of the densities in $D(\tilde{f}, f_{\underline{\alpha}})$. The matrix $\mathbf{R}$ describing the curvature of the surface has to be calculated depending on the type of density. Subsequently, the QP may be composed, i.e.,

$$\underline{\alpha}^{\mathrm{T}}\, \mathbf{D}\, \underline{\alpha} \; - 2\, \underline{d}^{\mathrm{T}} \underline{\alpha} \; + \lambda\; \underline{\alpha}^{\mathrm{T}}\; \mathbf{R}\; \underline{\alpha} = \underline{\alpha}^{\mathrm{T}}\; \mathbf{Q}\; \underline{\alpha} - \underline{q}^{\mathrm{T}} \underline{\alpha}\,,$$

with

$$\mathbf{Q} := \mathbf{D} + \lambda\,\mathbf{R}\,, \qquad\qquad \underline{q} := 2\,\underline{d}\,.$$

The matrix $\mathbf{Q}$ of the quadratic form is symmetric. Furthermore, the matrices $\mathbf{D}$ and $\mathbf{R}$ are positive semi-definite since the inequalities

$$\underline{\alpha}^{\mathrm{T}}\,\mathbf{D}\,\underline{\alpha} = \underline{\alpha}^{\mathrm{T}}\left(\int \underline{f}\,\underline{f}^{\mathrm{T}}\,\mathrm{d}\underline{x}\right)\underline{\alpha} = \int \left(\underline{\alpha}^{\mathrm{T}}\,\underline{f}\right)^2\,\mathrm{d}\underline{x} \geq 0 \tag{6}$$

and

$$\underline{\alpha}^{\mathrm{T}}\,\mathbf{R}\,\underline{\alpha} = \underline{\alpha}^{\mathrm{T}}\left(\int T(\underline{f})\,T(\underline{f})^{\mathrm{T}}\,\mathrm{d}\underline{x}\right)\underline{\alpha} = \int \left[\underline{\alpha}^{\mathrm{T}}T(\underline{f})\right]^2\,\mathrm{d}\underline{x} \geq 0 \tag{7}$$

hold for all $\underline{\alpha} \in \mathbb{R}^L$. The differential operator T depends on the respective upper bound $\breve{\kappa}(\underline{x})^2$ used. Thus, $\mathbf{Q}$ is positive semi-definite and the optimization problem is a convex QP. Using the mass and positivity constraints, one obtains a QP that may be solved by any standard solver. For the experiments in this paper, the freely available `CVX` optimization library [GB08] was used.

The purpose of the *post-optimization* is an adaptation of the already reduced weights $\underline{\alpha}^+$ aimed at improving the accuracy, by neglecting the curvature and only minimizing the ISD w.r.t. $\underline{\alpha}^+$, where only the weights $\alpha_i^+ \geq \varepsilon$ resulting from the weight optimization are considered. In the experiments in Sec. 5, $\varepsilon = 1e^4$ is used. The resulting QP is

$$\begin{aligned}
\min_{\underline{\alpha}^+} \quad & \left(\underline{\alpha}^+\right)^{\mathrm{T}}\,\left(\mathbf{D}^+\right)\,\left(\underline{\alpha}^+\right) \,-\, 2\,\left(\underline{d}^+\right)^{\mathrm{T}}\left(\underline{\alpha}^+\right) \\
\text{s.t.} \quad & \underline{0} \preceq \underline{\alpha}^+\,, \\
& \underline{1}^{\mathrm{T}}\,\underline{\alpha}^+ = 1\,, \\
& \textstyle\sum_i \alpha_i^+\,\underline{\mu}_i^+ = \sum_i \tilde{\alpha}_i\,\tilde{\underline{\mu}}_i\,.
\end{aligned}$$

This optimization problem for the reduced weights consists of the quadratic form of the ISD as a target function and the positivity, mass, and mean constraint w.r.t. to the reduced mixture's components. This QP may be solved with any standard solver. The obtained weights $\underline{\alpha}^*$ will be reduced again by removing components with almost zero weights, i.e., $\alpha_i^* < \varepsilon$. The overall algorithm is given in Alg. 1. Note, that the hyperparameter λ in Alg. 1, which governs the trade-off between the distance $D(\tilde{f}, f_{\underline{\alpha}})$ and the curvature term $R(f_{\underline{\alpha}})$, needs to be determined by means of generic model selection algorithms, cf. [SS02, 7.8,16]. For small values of λ, the ISD will be weighted relatively higher than the curvature. This results in more components in the reduced mixture f and less approximation error. For large values of λ, the curvature will be weighted higher enforcing more reduction and approximation error.

4 Limitations

The computational complexity for both optimization steps is polynomial in the number of mixture components. The cost for the post-optimization is smaller, as only components

Algorithm 1 Superficial Gaussian Mixture Reduction

Input: $\tilde{f}$

Calculate distance terms $\mathbf{D}, \underline{d}$, roughness penalty matrix $\mathbf{R}$, and λ ▷ Preprocessing

Compose Quadratic Program QP($\mathbf{D}$, $\underline{d}$, $\mathbf{R}$, λ)

$\underline{\alpha}^{+} \leftarrow$ REDUCE(SOLVE QP($\mathbf{D}$, $\underline{d}$, $\mathbf{R}$, λ)) ▷ Weight Optimization

$\underline{\alpha}^{*} \leftarrow$ REDUCE(OPTIMIZEWEIGHTS($\mathbf{D}^{+}$, $\underline{d}^{+}$, $\underline{\alpha}^{+}$)) ▷ Post-Optimization

function OPTIMIZEWEIGHTS($\mathbf{D}^{+}$, $\underline{d}^{+}$, α^{+})

    Compose Quadratic Program QP($\mathbf{D}^{+}$, $\underline{d}^{+}$)

    $\underline{\alpha}^{++} \leftarrow$ REDUCE(SOLVE QP($\mathbf{D}^{+}$, $\underline{d}^{+}$))

end function

function REDUCE($\underline{\alpha}'$)

    $\underline{\alpha}'' \leftarrow \underline{\alpha}' \geq \varepsilon$

end function

Output: $f \sim \text{GMM}\{ \underline{\alpha}^{*}, \{\underline{\mu}_i, \mathbf{\Sigma}_i\}^{*} \}$ ▷ $L^{*} \ll L$

with $\alpha_i^{+} \geq \varepsilon$ are considered, which may be significantly fewer. The underlying major assumption of this approach is that $\tilde{f}$ consists of many components. As will be shown in the experiments, the quality of the results depends on the number of components to be reduced. Therefore, the reduction of $\tilde{f}$ with very few components–actually not needing a reduction–will result in a low quality reduction $f_{\underline{\alpha}}$ as only $\underline{\alpha}$ is optimized, but no component means or covariances. The initially given set of means is identical to the set of means used in $f_{\underline{\alpha}}$. Note that even though this reduces the theoretical reduction capability, any $\tilde{f}$ with an insufficient number of components will not require a reduction at all.

5 Experiments

In the following, SGMR is compared to six established reduction methods: The simplest reduction is a **pruning** of all but the components with the highest weights [Bla86]. The top-down and local reduction algorithm, denoted by **West**, employs the Mahalanobis distance and merge two components at each reduction step [Wes93]. **Salmond**'s approach is similar to West's approach, but merges complete clusters of mixture components of size two and more [Sal90]. A top-down and global reduction algorithm based on the ISD (4) is proposed by **Williams** [WM03], where two components are merged per step and irrelevant components are additionally deleted. **Runnalls**' algorithm offers a compromise of local and global reduction algorithms, as it considers a localized upper bound of the (global)

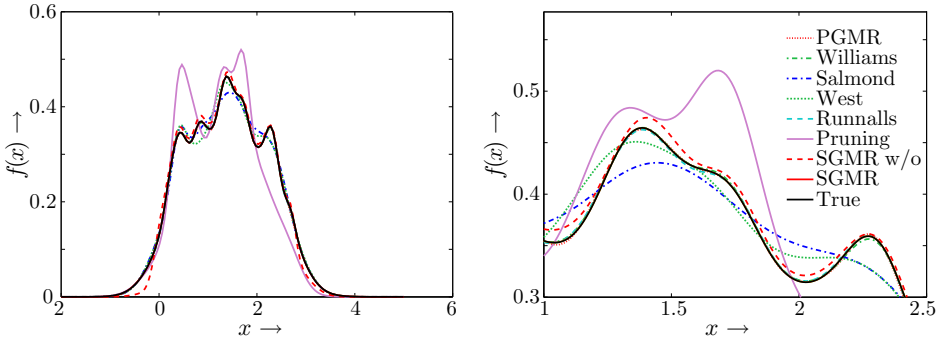

Figure 2: In (a), the true Gaussian mixture (black) and the reduced Gaussian mixture for the different reduction algorithms (blue, green - each with three styles), especially SGMR (red) without and with post-optimization (full, dashed) are given. In (b) a peak of the same Gaussian mixture and the respective reduced mixtures are focused. Note, that PGMR, Williams, Runnalls, and SGMR are almost identical to the true Gaussian mixture.

Kullback-Leibler divergence [Run07]. Merging is performed for two components. **PGMR** is a bottom-up approach employing the ISD (4) [HH08].

For SGMR, two variants are considered, one with post-optimization and one without post-optimization. The first four top-down approaches and the pruning method require a user-defined threshold on the number of components to which the given Gaussian mixture has to be reduced. Since SGMR reduces a Gaussian mixture in a completely different fashion and thus, to ensure a fair comparison, the number of components resulting from SGMR with post-optimization is used as threshold for these approaches. In order to quantify the reduction error, the normalized ISD

$$\bar{D}(\tilde{f}, f_{\underline{\alpha}}) = \sqrt{\frac{\int_{\mathbb{R}^N} \left(\tilde{f}(\underline{x}) - f_{\underline{\alpha}}(\underline{x})\right)^2 \mathrm{d}\underline{x}}{\int_{\mathbb{R}^N} \tilde{f}(\underline{x})^2 \,\mathrm{d}\underline{x} + \int_{\mathbb{R}^N} \tilde{f}_{\underline{\alpha}}(\underline{x})^2 \,\mathrm{d}\underline{x}}} \in [0,1] \tag{8}$$

is employed [HBR03]. It ranges between zero, which is the case if $\tilde{f}(\underline{x})$ and $f_{\underline{\alpha}}(\underline{x})$ are identical, and one, when both mixtures are absolutely non-overlapping. The algorithms are implemented in Matlab 7.8.0 (R2009a) and run on an office PC (Intel Core2 Duo P9600).

5.1 1-D Experiment

At first, univariate Gaussian mixtures with $L \in \{40, 80, 120, 160, 200\}$ components are used for evaluation. The mixture parameters are drawn uniformly at random from the intervals $\tilde{\alpha} \in [0.05, 0.5]$, $\tilde{\mu} \in [0, 3]$, and $\tilde{\sigma} \in [0.09, 0.5]$. For each number of components L, 50 Monte Carlo simulation runs are performed. For SGMR, the hyperparameter λ is set to 500 and the deletion threshold ϵ is $1e^{-4}$. The maximum error threshold of PGMR is set to 1%.

In Fig. 3 (top), the average reduction errors and the average computation times for all L are shown. It can be seen that SGMR provides the lowest reduction error. Closest to SGMR is Williams' algorithm, but this algorithm clearly suffers from its high computational demand. Salmond's and West's methods perform similarly. Both are very fast, but their approximation quality is the worst except for pruning. In terms of the reduction error, the results of Runnalls' method are in between of SGMR with and without post-optimization. But for an increasing number of components L in the original mixture it becomes computationally more expensive than both SGMR methods. Overall, SGMR provides the best trade-off between reduction error and computation time.

The reduction performance of SGMR improves with a larger number of components in the original mixture. As listed in Tab. 1, SGMR reduces to about 50% if the number of components of the original mixture is $L = 40$, while for $L = 200$ only 22% of the components remain. Since SGMR merely adapts the weights $\underline{\tilde{\alpha}}$, a larger number of components is advantageous for SGMR for a better exploitation of redundancies. This leads to a stronger reduction by a simultaneously lower reduction error. Furthermore, the comparison between SGMR and SGMR without post-optimization shows that the post-optimization always lowers both the reduction error and the number of components.

In Fig. 2, the reduction results for an exemplary Gaussian mixture with $L = 120$ components is depicted. SGMR, PGMR, Runnalls', and Williams' algorithm are capable of almost exactly capturing the shape of the original mixture, while West's and Salmond's algorithm show the tendency to oversmooth modes. The result of SGMR without post-optimization is in between. There is an obvious deviation to the original mixture, but the shape–especially the single modes–are captured very well. The inferior results of pruning can be explained by its simplicity. Components are only deleted on the basis of the value of their weights. No distance measure is used to quantify the loss of a component.

For 1-D mixtures, PGMR clearly is the best reduction algorithm. The reduction error is close to SGMR without post-optimization, but the number of components in the reduced mixture is significantly lower (see Tab. 1). However, a straightforward extension to multivariate mixtures is not possible as only axis-aligned Gaussian components can be utilized for representing the reduced mixture. For this reason, PGMR is not considered in the following 2-D experiment.

5.2 2-D Experiment

In this experiment, randomly generated bivariate Gaussian mixtures are considered. The weights $\underline{\tilde{\alpha}}$ and the elements c of the covariance matrices of the original mixture are drawn from the intervals $\tilde{\alpha} \in [0.05, 0.5]$ and $c \in [0.1, 1]$, respectively. 25% of the mean vectors are drawn from $\underline{\tilde{\mu}} \in [0, 0.75] \times [0, 1.5]$ and the remaining mean vectors are sampled from $\underline{\tilde{\mu}} \in [1.5, 3] \times [0, 1.5]$. Due to this placement of the Gaussian components, bimodality is forced in the true mixture. Again, 50 Monte Carlo simulation runs are performed for each number of components $L \in \{40, 80, 120, 160, 200\}$. The hyperparameter λ of SGMR is set to 0.04 and $\epsilon = 1e^{-4}$.

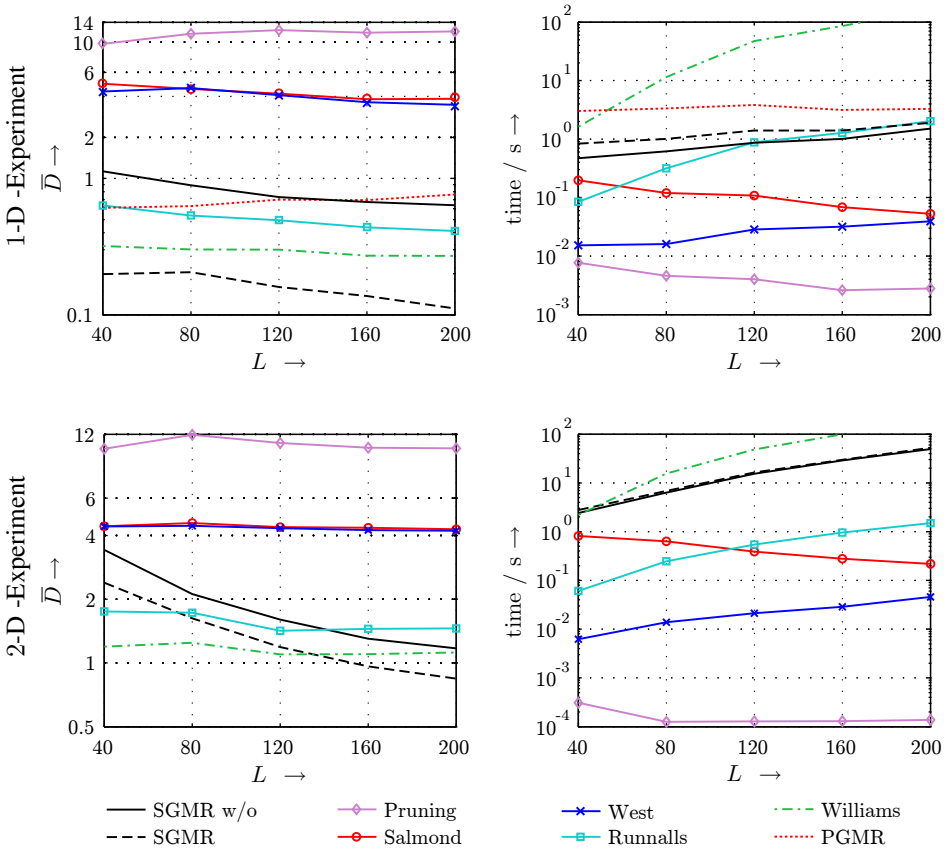

Figure 3: Reduction error (left) and time consumption (right) of different reduction algorithms for reducing 1-D and 2-D Gaussian mixtures with increasing number of components. The results are averages over 50 Monte Carlo runs. The average reduction error is multiplied by 100 for better readability.

Fig. 3 (bottom) gives the reduction error and the average computation time. In comparison to the 1-D experiment, it becomes more obvious that SGMR is the ideal method for reducing a Gaussian mixture with a large number of components. From $L = 160$ on, SGMR outperforms all other algorithms with respect to the reduction error. Furthermore, the benefit of the post-optimization is more significant than in the 1-D case. Besides a lower reduction error, the number of components can be reduced much stronger as shown in Tab. 1. This benefit comes with a low computational overhead.

In contrast to the 1-D experiment, the computation time of SGMR now significantly increases with the number of components. Most of the time is used for calculating the roughness penalty matrix $\mathbf{R}$, which requires numerical integration in the 2-D case. It is expected that an improved problem-adequate implementation of the numerical integration will reduce the computational demand drastically.

1-D -Experiment	L	SGMR w/o	PGMR	SGMR & all others
	40	21.66	7.34	20.54
	80	30.88	7.42	29.5
	120	37.18	7.58	35.86
	160	42.86	6.78	41.8
	200	45.68	6.7	44.98

2-D -Experiment	L	SGMR w/o	SGMR & all others
	40	18.92	16.68
	80	26.76	22.88
	120	40.32	33.32
	160	52.72	42
	200	61.8	43.16

Table 1: Number of components in the reduced Gaussian mixture for different reduction algorithms. The results are averages over 50 Monte Carlo runs.

6 Conclusion

In this paper, a curvature-based reduction algorithm for Gaussian mixtures was presented. The key idea is the formulation of the reduction problem as an optimization problem. The optimization balances the integral squared distance between true and reduced density with the reduction in approximate shape curvature. The arising problem is solved for the weights of the Gaussian mixture only, i.e., reduced components are assigned a weight of zero, allowing a formulation as a quadratic program. The main contributions are the alleviation of the model selection problem, as the number of components is chosen by the algorithm automatically and an easy as well as efficient implementation. The experiments show the high quality and low number of components of the approach's results.

As future work, it remains to derive an analytic upper bound for the N-dimensional curvature and to improve the numerical calculations. For very large reduction problems, more efficient algorithms could be obtained from exploiting the locality of Gaussian mixtures.

Acknowledgment Marco Huber would like to thank the Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB, Karlsruhe, Germany, for its support during the early stages of this work.

References

[AS72] Daniel L. Alspach and Harold W. Sorenson. Nonlinear Bayesian Estimation using Gaussian Sum Approximation. *IEEE Transactions on Automatic Control*, 17(4):439–448, August 1972.

[Bla86] Samuel S. Blackman. *Multiple-Target Tracking with Radar Applications*. Norwood, MA: Artech House, 1986.

[BSL95] Yaakov Bar-Shalom and Xiao-Rong Li. *Multitarget-multisensor Tracking: Principles and Techniques*. YBS Publishing, Storrs, CT, 1995.

[Car76] Manfredo Do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, NJ, 1976.

[CGJ96] David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. Active Learning with Statistical Models. *Arxiv preprint cs.AI/9603104*, 1996.

[GB08] Michael C. Grant and Stephen P. Boyd. Graph implementations for nonsmooth convex programs. In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008.

[HBH06] Marco Huber, Dietrich Brunn, and Uwe D. Hanebeck. Closed-Form Prediction of Nonlinear Dynamic Systems by Means of Gaussian Mixture Approximation of the Transition Density. In *Proceedings of the 2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2006)*, pages 98–103, Heidelberg, Germany, September 2006.

[HBR03] Uwe D. Hanebeck, Kai Briechle, and Andreas Rauh. Progressive Bayes: A New Framework for Nonlinear State Estimation. In *Proceedings of SPIE, AeroSense Symposium*, volume 5099, pages 256–267, Orlando, Florida, May 2003.

[HH08] Marco F. Huber and Uwe D. Hanebeck. Progressive Gaussian Mixture Reduction. In *Proceedings of the 11th International Conference on Information Fusion (Fusion 2008)*, pages 1–8, Cologne, Germany, July 2008.

[MS96] Vladimir Maz'ya and Gunther Schmidt. On approximate approximations using Gaussian kernels. *IMA J. Numer. Anal.*, 16:13–29, 1996.

[RS97] Jim Ramsay and Bernard Silverman. *Functional Data Analysis*. Springer Series in Statistics. Springer, New York, Berlin, Heidelberg, 1997.

[Run07] Andrew R. Runnalls. Kullback-Leibler Approach to Gaussian Mixture Reduction. *IEEE Transactions on Aerospace and Electronic Systems*, 43(3):989–999, July 2007.

[Sal90] David J. Salmond. Mixture reduction algorithms for target tracking in clutter. In *Proceedings of SPIE Signal and Data Processing of Small Targets*, volume 1305, pages 434–445, October 1990.

[SH09] Dennis Schieferdecker and Marco F. Huber. Gaussian Mixture Reduction via Clustering. In *Proceedings of the 12th International Conference on Information Fusion (Fusion)*, pages 1536–1543, Seattle, Washington, July 2009.

[Sil98] Bernard W. Silverman. *Density Estimation for Statistics and Data Analysis*. Monographs on Statistics and Applied Probability ; 26. CRC Press, Boca Raton, 1998.

[SS02] Bernhard Schölkopf and Alexander Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive computation and machine learning series. MIT Press, Cambridge, Massachusetts, 2002.

[Wes93] Mike West. Approximating Posterior Distributions by Mixtures. *Journal of the Royal Statistical Society: Series B*, 55(2):409–422, 1993.

[WM03] Jason L. Williams and Peter S. Maybeck. Cost-Function-Based Gaussian Mixture Reduction for Target Tracking. In *Proceedings of the Sixth International Conference of Information Fusion (Fusion 2003)*, volume 2, pages 1047–1054, Cairns, Australia, July 2003.