Gaussian Mixture Reduction via Clustering

Dennis Schieferdecker
Algorithmics Group II
Institute for Theoretical Computer Sciences
University Karlsruhe (TH), Germany
schiefer@ira.uka.de

Marco F. Huber
Intelligent Sensor-Actuator-Systems Laboratory
Institute for Anthropomatics
University Karlsruhe (TH), Germany
marco.huber@ieee.org

Abstract – *Recursive processing of Gaussian mixture functions inevitably leads to a large number of mixture components. In order to keep the computational complexity at a feasible level, the number of their components has to be reduced periodically. There already exists a variety of algorithms for this purpose, bottom-up and top-down approaches, methods that take the global structure of the mixture into account or that work locally and consider few mixture components at the same time. The mixture reduction algorithm presented in this paper can be categorized as global top-down approach. It takes a clustering algorithm originating from the field of theoretical computer science and adapts it for the problem of Gaussian mixture reduction. The achieved results are on the same scale as the results of the current "state-of-the-art" algorithm PGMR, but, depending on the input size, the whole procedure performs significantly faster.*

Keywords: Gaussian mixture reduction, nonlinear optimization, clustering

1 Introduction

Gaussian mixture functions can be applied for a lot of different tasks. They are often used to model probability density functions (PDFs) in estimation algorithms such as for machine learning [3], speaker recognition [10] or target tracking [2], to name a few. As the set of all Gaussians forms a complete base system, i.e., every function can be approximated by a Gaussian mixture with arbitrary accuracy [9], they are a very potent tool for these applications.

A general drawback of these mixture functions is their tendency to grow at an exponential rate when processed in recursion. Therefore, the mixture has to be reduced after several computation steps before the number of components grows too large. This mixture reduction step should be computationally cheap and the reduced density should only have a small deviation from the original one.

Previous approaches dealing with the problem of Gaussian mixture reduction can be classified into two fields. First, there are some *bottom-up approaches* that start with a single Gaussian function and iteratively add additional components until the original mixture density is approximated appropriately. The PGMR algorithm [5] uses this approach. Then, there are a lot of *top-down* or *classical approaches*. They take the original Gaussian mixture density and iteratively decrease the number of mixture components, either by removing single "unimportant" components or by merging similar ones. These algorithms can be further divided into *local* and *global* methods. The joining and clustering algorithms by Salmond [12], West's algorithm [15] and IPRA by Scott and Szewczyk [14] belong to the former. They consider only few characteristics of the overall mixture function or even just individual mixture components independently from the remaining ones. Global methods, like Williams' algorithm [16], take the whole Gaussian mixture into account when selecting components for removal or merging. They usually provide better approximation results but also take much longer to finish. Runnalls [11] tries to incorporate elements from both the local and the global approach. He takes a complex global quality measure, the Kullback-Leibler divergence, and then tries to optimize a computational cheap, localized upper-bound, when removing or merging cluster components.

The approach presented in this paper can be classified as a top-down algorithm using global information. The original idea comes from the field of theoretical computer science. Here, *clustering* is one of the classic problems in machine learning and computational geometry. Given a number of fixed locations, so-called *sites*, and an integer L as input, one tries to find L other locations, *cluster centers*, so that a distance measure between the sites and the cluster centers is minimized. This idea is transfered to the problem of reducing Gaussian mixtures by interpreting each component of the original mixture as a site and each component of the reduced one as a cluster center. This is different to existing clustering approaches for Gaussian mixture reduction like Salmond's clustering algorithm. Instead of computing only one cluster per step for merging the associated components, we distribute *all* mixture components into multiple clusters in order to find an appropriate cluster representation of the (true) mixture. Computing a good reduced Gaussian mixture becomes equal to finding good cluster centers using an appropriate distance measure. By means of simulation, it is demonstrated that our proposed algorithm for Gaussian mixture reduction yields results with a quality on the same

scale as those obtained by the current "state-of-the-art" algorithm PGMR but, depending on the input size, it performs significantly faster.

The remainder of the paper is structured as follows: In the next section we formally define the Gaussian mixture reduction problem. Then, Section 3 describes our proposed algorithm and how a clustering technique can be used to reduce Gaussian mixtures. Subsequently in Section 4, we compare the performance of our algorithm and the quality of its results to existing algorithms. Finally, we give conclusions and provide an outlook to future work.

2 Problem Formulation

We assume that a density function

$$\tilde{f}(x;\underline{\tilde{\eta}}) = \sum_{i=1}^{N} \tilde{\omega}_i \cdot \mathcal{N}(x;\tilde{\mu}_i,\tilde{\sigma}_i^2)$$

of a random variable x is given in the form a Gaussian mixture with N components, non-negative weights $\tilde{\omega}_i$ with $\sum_i \tilde{\omega}_i = 1$ and $\mathcal{N}(x;\tilde{\mu},\tilde{\sigma}^2)$ denoting a Gaussian function with mean $\tilde{\mu}$ and variance $\tilde{\sigma}^2$. The parameters of the mixture density are represented by the parameter vector

$$\underline{\tilde{\eta}} = [\underline{\tilde{\eta}}_1^{\mathrm{T}}, \underline{\tilde{\eta}}_2^{\mathrm{T}}, \ldots, \underline{\tilde{\eta}}_N^{\mathrm{T}}]^{\mathrm{T}}, \quad \underline{\tilde{\eta}}_j^{\mathrm{T}} = [\tilde{\omega}_j, \tilde{\mu}_j, \tilde{\sigma}_j^2].$$

Given this density function and an integer value L, we want to compute a *reduced Gaussian mixture* density

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

with $L \ll N$ components that is close to the original density according to some distance measure $D(\tilde{f}(x), f(x))$. Its parameter vector $\underline{\eta}$ is defined accordingly to $\underline{\tilde{\eta}}$.

Gaussian mixture densities are described completely by their parameter vectors $\underline{\eta}$. Therefore, we subsequently refer to $\underline{\eta}$ as the whole mixture and to $\underline{\eta}_j$ as mixture component j of $\underline{\eta}$ for brevity's sake. To distinguish between a reduced mixture and the original one we indicate the parameters of the latter by a tilde.

When evaluating the quality of a reduced density $f(x)$ compared to the original density $\tilde{f}(x)$, the Integrated Squared Distance (ISD) [6] is often used as distance measure. We apply the normalized ISD

$$D(\tilde{f}(x), f(x)) = \frac{\int_{\mathbb{R}} \left(\tilde{f}(x) - f(x)\right)^2 \mathrm{d}x}{\int_{\mathbb{R}} \tilde{f}(x)^2 \mathrm{d}x + \int_{\mathbb{R}} f(x)^2 \mathrm{d}x} \qquad (1)$$

instead, since its constant range $D \in [0,1]$ is convenient for comparison. A different deviation measure such as the Kullback-Leibler (KL) divergence [7] could have also been applied. In fact, it would have been the natural deviation measure for mixture reduction in a maximum likelihood sense [11, 16]. The normalized ISD was chosen instead since it can be computed in closed-form, resulting in a faster algorithm, and because of its constant range.

3 Clustering Algorithm

The mixture reduction approach presented in this paper, Gaussian Mixture Reduction via Clustering (GMRC), is a three step algorithm as outlined by Algorithm 1.

Algorithm 1 Basic Algorithm: $\underline{\eta} \leftarrow \mathrm{GMRC}(\underline{\tilde{\eta}}, L)$

1: $\underline{\eta} \leftarrow \mathrm{Preprocessing}(\underline{\tilde{\eta}}, L)$
2: $\underline{\eta} \leftarrow \mathrm{Clustering}(\underline{\tilde{\eta}}, \underline{\eta})$
3: $\underline{\eta} \leftarrow \mathrm{Refinement}(\underline{\tilde{\eta}}, \underline{\eta})$

A preprocessing step takes the original Gaussian mixture $\underline{\tilde{\eta}}$ and computes a preliminary solution $\underline{\eta}$ with L components of the reduction problem. The mixture components $\underline{\eta}_j$ are then used as initial cluster centers for the clustering step. Here, the current solution is iteratively improved by a clustering approch, greedily optimizing the associations of mixture components $\underline{\tilde{\eta}}_i$ to cluster centers $\underline{\eta}_j$. Afterwards, a refinement step further improves the current result via numerical optimization.

Subsequently, we analyze each of the three steps in more detail. Later in the simulation section, we show that none of them are mandatory but that each of them contributes to the quality of the reduced Gaussian mixture.

3.1 Preprocessing Step

We apply Runnalls' algorithm [11] to compute an initial solution $\underline{\eta}$. This algorithm was chosen since it yields a good compromise between approximation quality and overall computation time. It applies a top-down approach where in each step the two mixture components $\underline{\eta}_i$ and $\underline{\eta}_j$ with the smallest dissimilarity are replaced by their moment-preserving merging $\underline{\eta}_m = [\omega_m, \mu_m, \sigma_m^2]^{\mathrm{T}}$ with

$$\begin{aligned} \omega_m &= \omega_i + \omega_j, \\ \mu_m &= \frac{1}{\omega_m} \cdot (\omega_i \mu_i + \omega_j \mu_j), \\ \sigma_m^2 &= \frac{1}{\omega_m} \cdot \left(\omega_i \sigma_i^2 + \omega_j \sigma_j^2 + \frac{\omega_i \omega_j}{\omega_m} (\mu_i - \mu_j)^2\right). \end{aligned} \qquad (2)$$

To determine the dissimilarity of two components $\underline{\eta}_i$ and $\underline{\eta}_j$, Runnalls employs the deviation measure

$$B(\underline{\eta}_i, \underline{\eta}_j) = \tfrac{1}{2}(\omega_m \log \sigma_m^2 - \omega_i \log \sigma_i^2 - \omega_j \log \sigma_j^2) \qquad (3)$$

with ω_m, σ_m^2 according to (2). This is an upper bound on the Kullback-Leibler divergence between a Gaussian mixture with the components $\underline{\eta}_i$ and $\underline{\eta}_j$ and a mixture with the Gaussian $\underline{\eta}_m$.
In the simulation section we will also focus on the impact of using different algorithms to find an initial solution, West's algorithm [15] and a random initialization, in particular.

3.2 Clustering Step

To improve Runnalls' initial solution, we reinterpret the task of finding a reduced mixture as clustering problem and adapt the well-known *k-means algorithm* for this task.

Clustering. In general, a clustering problem can be described by a set $\mathcal{S} \subset \mathbb{R}^d$ of N sites and an integer L. A clustering algorithm tries to find L centers $\mathcal{C}$ so that a deviation measure, i.e.,

$$D(\mathcal{S}, \mathcal{C}) = \sum_{s_i \in \mathcal{S}} \min_{c_j \in \mathcal{C}} ||s_i - c_j||^2$$

is minimized. The clustering is defined implicitly by the centers. A site s_i belongs to cluster j, if the distance to its cluster center $c_j \in \mathcal{C}$ is smaller than to any other cluster center c_k with $k \neq j$. Note that this problem is NP-hard and stays so, even for only two clusters [4].

K-means Algorithm. The k-means algorithm is a popular clustering approach by Lloyd [8]. Its basic procedure can be summarized as follows:

1. Choose L initial cluster centers $\mathcal{C}$ at random from $\mathbb{R}^d$,
2. for each $i \in \{1, \ldots, L\}$, let cluster C_i be the set of sites in $\mathcal{S}$ closer to center c_i than to any other center c_j with $j \neq i$ f.a. $j \in \{1, \ldots, N\}$,
3. for each $i \in \{1, \ldots, L\}$, set c_i to be the center of mass of all sites in cluster C_i,
4. repeat steps 2 and 3 as necessaery.

Our clustering approach is outlined in Algorithm 2. It makes several modifications to the k-means algorithm, in particular concerning the applied deviation measures, as shown below.

Algorithm 2 Clustering Step: $\underline{\eta} \leftarrow \text{Clustering}(\underline{\tilde{\eta}}, \underline{\eta})$

1: $\mathcal{C} \leftarrow \text{InitialClustering}(\underline{\tilde{\eta}}, \underline{\eta})$
2: $\underline{\eta} \leftarrow \text{ComputeCenters}(\underline{\tilde{\eta}}, \mathcal{C})$
3: **for all** sites $\underline{\tilde{\eta}}_i \in \underline{\tilde{\eta}}$ **do**
4: **for all** clusters $C_j \in \{1, \ldots, L\}$ **do**
5: $\mathcal{C}_{\text{tmp}} \leftarrow$ reassociate site $\underline{\tilde{\eta}}_i$ to cluster C_j
6: $\underline{\eta}_{\text{tmp}} \leftarrow \text{ComputeCenters}(\underline{\tilde{\eta}}, \mathcal{C}_{\text{tmp}})$
7: $dist_j \leftarrow d(\underline{\tilde{\eta}}, \underline{\eta}_{\text{tmp}})$
8: **end for**
9: $\mathcal{C} \leftarrow$ reassociate site $\underline{\tilde{\eta}}_i$ to cluster C_j for $C_j \in \{1, \ldots, L\}$ with $dist_j$ minimal
10: $\underline{\eta} \leftarrow \text{ComputeCenters}(\underline{\tilde{\eta}}, \mathcal{C})$
11: **end for**

In terms of clustering, the parameter vector of the original Gaussian mixture $\underline{\tilde{\eta}}$ corresponds to the set of sites $\mathcal{S}$ and the the parameter vector of the reduced mixture $\underline{\eta}$ to the set of cluster centers $\mathcal{C}$. Accordingly, $\underline{\tilde{\eta}}_i$ and $\underline{\eta}_j$ correspond to one site s_i or one cluster center c_j, respectively.

InitialClustering. Before commencing the clustering process, each of the original mixture components in $\underline{\tilde{\eta}}$ has to be associated with one of the cluster centers in $\underline{\eta}$ that have been computed during the preprocessing step. This association is described by a cluster allocation $\mathcal{C}$ with

$$\mathcal{C}_{i,j} = \begin{cases} 1 & \text{component } \underline{\tilde{\eta}}_i \text{ is associated with cluster } C_j, \\ 0 & \text{otherwise.} \end{cases}$$

A mixture component $\underline{\tilde{\eta}}_i$ is associated to the center $c_j \mathrel{\widehat{=}} \underline{\eta}_j$ of the cluster $C_j \in \{1, \ldots, L\}$, for which the Kullback-Leibler divergence

$$\begin{aligned} D(\underline{\tilde{\eta}}_i, \underline{\eta}_j) &= \int_{\mathbb{R}} \tilde{\mathcal{N}}_i \log \frac{\tilde{\mathcal{N}}_i}{\mathcal{N}_j} \\ &= \frac{1}{2\sigma_j^2} \left((\tilde{\sigma}_i^2 - \sigma_j^2) + (\tilde{\mu}_i - \mu_j)^2 \right) + \log \frac{\sigma_j}{\tilde{\sigma}_i} \end{aligned}$$

between them, with $\mathcal{N}_i = \mathcal{N}(x; \mu_i, \sigma_i^2)$, is minimal. Thus, each component is associated with exactly one cluster center. Weights are omitted in the KL divergence we use, since only similarity of the general shape of the two Gaussians is to be compared.

ComputeCenters. After a cluster allocation $\mathcal{C}$ has been determined, it can be used to compute cluster centers. For each cluster $C_j \in \{1, \ldots, L\}$ the weight ω_j, the mean μ_j and the variance σ_j^2 of the sum of all mixture components $\underline{\tilde{\eta}}_i \in \{1, \ldots, N \mid \mathcal{C}_{i,j} = 1\}$ associated with this cluster are computed according to

$$\begin{aligned} \omega_j &= \sum_{\mathcal{C}_{i,j}=1} \tilde{\omega}_i \\ \mu_j &= \sum_{\mathcal{C}_{i,j}=1} \tilde{\omega}_i \tilde{\mu}_i \\ \sigma_j^2 &= \sum_{\mathcal{C}_{i,j}=1} \tilde{\omega}_i (\tilde{\sigma}_i^2 + \mu_i^2) / \omega_j - \mu_j^2 \end{aligned}$$

and used as cluster center $\underline{\eta}_j = [\omega_j, \mu_j, \sigma_j^2]^{\mathrm{T}}$.

Step 1 of the k-means algorithm corresponds to our preprocessing step and the first two lines of the clustering step. But instead of using a random initialization, we apply a more sophisticated initial solution calculated by means of Runnalls' algorithm.

Clustering Loop. The central part of the clustering step, lines 3 – 11, corresponds to steps 2 and 3 of the k-means algorithm. For each site s_i or component of the original mixture $\underline{\tilde{\eta}}_i$, the association to a cluster $C_j \in \{1, \ldots, L\}$ is determined that minimizes a deviation measure d (line 7). Ambiguities are resolved arbitrarily. The cluster centers are updated when each site has been associated with a new cluster. This is unlike to the k-means algorithm that updates the cluster centers when all sites have been processed.

We apply the normalized ISD (1) as deviation measure d, because it is the actual value we want to optimize. This is a further difference to the k-means algorithm, where a local deviation measure, the Euclidean distance between a site and a cluster center is used. Section 4 provides additional results for a local deviation measure (3), the upper bound on the KL divergence, also used in the preprocessing step.

The greedy approach taken by the algorithm ensures that in each step an allocation of a site to a cluster is determined that is at least as good as the current solution. Thus, if the algorithm is performed long enough, the result converges to a local optimum. But it also can be stopped at any time after one iteration of the outer loop has finished (line 11) and will yield a proper result. It is evident that the choice of the initial solution is important. Choosing a starting point for the clustering algorithm close to a good local optimum yields a faster convergence.

Since only one cluster association is changed during each execution of the inner loop, the computation of the new cluster centers and of the deviation measure can be accelerated by a considerable amount if only the parts that have been changed are recomputed.

In principal, clusters could become empty, i.e., no sites are associated to a cluster. But in our experience this happens only rarely, usually if the initial solution is chosen improperly and the number of sites N and clusters L is close together.

Note that the clustering step conserves the mean and the variance of the original Gaussian mixture, independently of the implementation of the preprocessing step.

Further Improvements. The clustering algorithm can be enhanced in several ways. These approaches either focus on improving quality of the results or on accelerating the compuation, which are in general conflicting goals. Some of these strategies can be combined to achieve an improvement for both aspects. Several of the possibilities listed below are evaluated later on their own and are combined in the simulation section.

The clustering quality can be improved by repeating the iteration over all mixture components in $\tilde{\underline{\eta}}$ several times. In fact, this is the usual procedure of the k-means clustering algorithm (see step 4 of the algorithm in Section 3). However, it is omitted in our initial algorithm because of the high computational cost of a single iteration and since the results are typically very good after only one iteration.

Since the algorithm greedily optimizes the associations of sites to clusters, one after the other, the order in which the sites are processed is important for the final results. Thus, a more sophisticated ordering, like considering sites with a higher weight first, might improve the results. So far, no special ordering is employed.

When iterating over the mixture components in $\tilde{\underline{\eta}}$, the components with a small weight $\tilde{\omega}_i$ below some threshold $\tilde{\omega}_{max}$ can be skipped to save processing time. Assuming that componentes with a small weight do not contribute much to the expected deviation, improving their cluster allocation does only yield a small gain in approximation quality. Thus, skipping them has only little impact on the quality of the clustering results.

As shown in the simulation section, the initial solution seems to have a considerable impact on the local optimum the remaining parts of the algorithm are able to find. Thus, applying restarts and choosing the best of the results is a viable strategy to improve the results unless restarts are too expensive.

3.3 Refinement Step

In general, a simple greedy merging of mixture components does not yield optimal results when trying to minimize a global distance measure between a Gaussian mixture $\tilde{f}(x)$ and a reduced one $f(x)$. Therefore, a refinement step is added to improve the quality of the reduced Gaussian mixture $f(x;\underline{\eta})$ with parameter vector $\underline{\eta}$. Here, we apply two optimization routines as described below. Both of them try to optimize the ISD between the original and the reduced Gaussian mixture instead of the normalized ISD since it is much easier to compute, yielding a faster algorithm.

Newton approach. At first, we try to optimize $\underline{\eta}$ by determining the roots of the derivative of the ISD

$$M(\tilde{f}(x;\tilde{\underline{\eta}}), f(x;\underline{\eta})) = \int_{\mathbb{R}} \left(\tilde{f}(x;\tilde{\underline{\eta}}) - f(x;\underline{\eta})\right)^2 \mathrm{d}x$$

between the original and the reduced Gaussian mixture with respect to $\underline{\eta}$. This is done with the Newton approach

$$\left.\frac{\partial^2 M}{\partial \underline{\eta}^2}\right|_{\underline{\eta}=\underline{\eta}_k} \cdot \Delta\underline{\eta} = -\left.\frac{\partial M}{\partial \underline{\eta}}\right|_{\underline{\eta}=\underline{\eta}_k}, \tag{4}$$

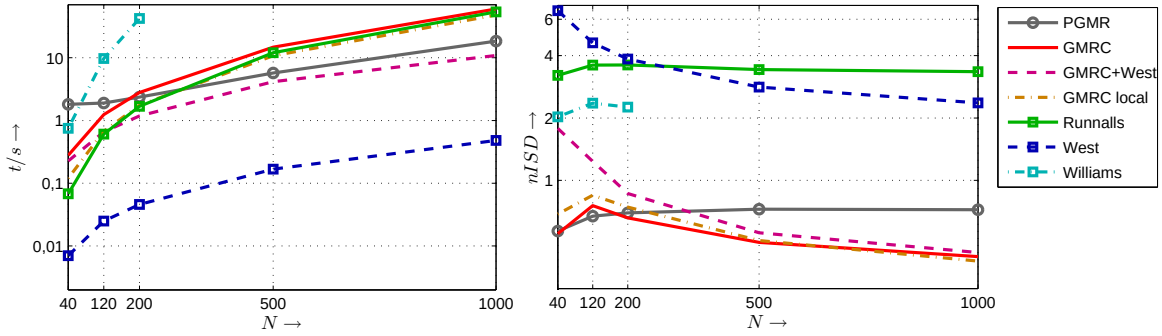
as described in [5]. The change $\Delta\underline{\eta} = \underline{\eta}_{n+1} - \underline{\eta}_n$ in the parameter vector $\underline{\eta}$ is determined by solving the system of linear equations (4). The iteration

$$\underline{\eta}_{n+1} = \underline{\eta}_n + \Delta\underline{\eta}$$

is initialized with $\underline{\eta}_0 = \underline{\eta}$ and repeated until $\Delta\underline{\eta} < 10^{-5}$ or at most 100 times, whichever happens first. In the latter case, the result of the last iteration step is returned.

Weight Optimization. After the Newton approach, we renormalize the weights according to $\sum_i \omega_i = 1$. Then, we perform a further improvement step by optimizing the weights $\underline{\omega} = [w_1, \ldots, \omega_L]^{\mathrm{T}}$ of the reduced Gaussian mixture. For this purpose, we determine the minimum of the ISD between the original and the reduced Gaussian mixture with regard to the weights $\underline{\omega}$. The system of linear

Figure 1: EXECUTION TIMES AND NORMALIZED ISDS FOR THE ANALYZED APPROXIMATION ALGORITHMS.



equations

$$\begin{bmatrix} \int_{\mathbb{R}} \mathcal{N}_1^2 \mathrm{d}x & & \\ & \ddots & \\ & & \int_{\mathbb{R}} \mathcal{N}_L^2 \mathrm{d}x \end{bmatrix} \underline{\omega} = \sum_{j=1}^{N} \tilde{\omega}_j \begin{bmatrix} \int_{\mathbb{R}} \tilde{\mathcal{N}}_j \mathcal{N}_1 \mathrm{d}x \\ \vdots \\ \int_{\mathbb{R}} \tilde{\mathcal{N}}_j \mathcal{N}_L \mathrm{d}x \end{bmatrix},$$

with $\mathcal{N}_i = \mathcal{N}(x; \mu_i, \sigma_i^2)$, has to be solved (for a full derivation see Appendix A). The resulting weights $\underline{w}$ are used as weights for the reduced Gaussian mixture $\underline{\eta}$.

Note that in general, the refinement step will not conserve the mean and the variance of the original Gaussian mixture without applying further modifications to the optimization routines such as Lagrangian multipliers.

4 Simulation Results

Our simulation setup consists of an office PC with an Intel Core2 Duo E8400 processor running OpenSUSE 11.0. The algorithms are implemented in Mathlab 7.7.0 (R2008b). All simulations are performed 1 000 times. For improved readability, the square root of the normalized ISD, multiplied by 100, is listed in the respective tables.

4.1 Comparison to Other Algorithms

We evaluate the performance of our proposed algorithm, GMRC, by comparing it to several recent other algorithms. In particular, we provide results for both West's and Williams' algorithms as representatives for the local and global top-down approaches as well as for the bottom-up approach PGMR. In addition, Runnalls' algorithm is also listed as it provides the preprocessing for our algorithm.

For an easier comparison to [5], we apply the same basic test cases. True Gaussian mixtures with $N \in \{40, 120, 200, 500, 1000\}$ components are used for evaluation. The mixture parameters are drawn uniformally at random from the intervals

$$\begin{aligned} \omega &\in [0.05, 0.5]\,, \\ \mu &\in [0, 3]\,, \\ \sigma &\in [0.09, 0.5]\ . \end{aligned} \tag{5}$$

Note that we did not evaluate Williams' algorithm for $N \in \{500, 1000\}$ due to the extremely long execution times.

The algorithms reduce the Gaussian mixtures to $L = 10$ components (or less in case of PGMR). In addition, for PGMR a maximum error with respect to the normalized ISD of $M_{max} = 1$ is selected. The resulting mean values of the execution time t and of the normalized ISD are shown in Fig. 1. More detailed results including standard deviations can be found in Tab. 4 in Appendix B [1].

As can be seen, West's algorithm is the fastest one by a large margin, but the quality of its results is mostly inferior to the other algorithms. The improvement of the approximation quality for larger values of N can be explained by our simulation setup where Gaussian mixtures with more components tend to produce "nicer" curves. Williams' algorithm produces better results but takes much too long to be of any practical value. PGMR and GMRC both deliver the best results among all of the studied algorithms with comparable approximation qualities. But whereas GMRC always reduces the Gaussian mixture to 10 components, PGMR only adds components until the normalized ISD is below a given threshold. For the smaller input sizes this results in about 7 components and for the largest input size in only 5 components on average. Therefore, the approximation quality of PGMR also seems to degrade for larger values of N, which acutally is not true. Regarding the execution times, GMRC is faster for small input sizes and PGMR performs better on larger inputs.

As can be further deduced from the simulational results, the leading term of the computational time complexity of PGMR seems to be $O(N^2)$, whereas for GMRC it is $O(N^3)$. The acutal execution times are in favor for GMRC up until about $N \approx 150$ components. Runnalls' algorithm also exhibits a time complexity of $O(N^3)$. Since our approach uses this algorithm during the preprocessing step, it is not surprising that GMRC also requires at least $O(N^3)$ time. Considering only the execution time of the clustering step and the refinement step by subtracting the running time of Runnalls' algorithm from the running time of GMRC, we see that the computational time complexity of these steps is the same as for PGMR, $O(N^2)$.

[1] Note that we apply the most recent version of PGMR. We have also discovered an inconsistency in the existing evaluation procedures that has been corrected. This leads to considerably different results than provided in a previous publication [5].

Since the time complexity of these results is not convincing, we also evaluated the impact of using West's algorithm as a faster preprocessing step for GMRC. As can be seen, the actual running time is always well below PGMR and the asymptotic time complexity of GMRC+West is the same as of PGMR, $O(N^2)$. For small input sizes, the approximation quality is inferior to both PGMR and GMRC but already better than any of the former top-down approaches. At about $N \approx 300$, the approximation quality of GMRC using West's algorithm reaches the results of PGMR and GMRC using Runnalls' algorithm.

In addition to the normal GMRC algorithm, we have also evaluated a variant called *GMRC local*. This variant employes the upper bound (3) on the KL divergence between a component of the original mixture and the reduced one as deviation measure d for the clustering step (see line 7 of Algorithm 2). The approximation quality is worse than for the normal algorithm but it improves for larger input sizes. Also, the standard deviation of its normalized ISD is a lot higher, being in the same region as the normalized ISD itself. When comparing the results to Tab. 2, we see that the quality of the results is only slightly better than if the clustering step would have been completely omitted. Thus, using this local deviation measure is not beneficial.

We want to ensure that our algorithm not only optimizes the value of the normalized ISD, but also provides an improvement with regards to other deviation measures. Therefore, we determined the KL divergence for the approximation results of our analyzed algorithms (using $N = 200$ and $L = 10$ components). The KLD was computed by means of numerical integration. The results can be found in Tab. 1. Again, PGMR and GMRC provide the best choices among all of the analyzed algorithms. Thus, we can state that even though GMRC applies the normalized ISD for its optimization process, the resulting mixtures also yield small values for the KL divergence as deviation measure.

4.2 Evaluation of GMRC steps

We have performed further simulations to evaluate the importance of each of the three steps of the GMRC algorithm. At first, we effectively skip the preprocessing step by using random cluster centers with parameters chosen from the same intervals as in (5). With this setup, we want to study the impact of the quality of the inital solution. The next simulation omits the whole clustering step and just applies our refinement routines to the solution found by Runnalls' algorithm. This is done to estimate the actual value of the clustering process. Finally, we study the effect of not using a refinement step. All simulations have been performed for $N = 200$ original components that are reduced to $L = 10$. The results of the simulations are summarized in Tab. 2. For comparison, the results of the complete GMRC algorithm and of Runnalls' algorithm are also provided.

As can be deduced from the results, the preprocessing step, namely Runnalls' algorithm, requires more than half of the overall running time whereas the refinement step only takes about 10% of the time. Apparently, the quality of the initial solution has a large influence on the overall results. Also, the refinement step appears to play a crucial part in obtaining good approximations. Without using the Newton approach and the weight optimization, the quality of the results is degraded by a large degree but it is still better than West's, Wiliams' and Runnalls' algorithms. As can be seen by omitting the clustering step, it is also responsible for the approximation quality. Without the clustering step the mean of the normalized ISD becomes about 15% larger and its standard deviation gets more than twice as large.

Thus, we conclude that, given a good initial solution, the clustering step pushes it further towards a good local optimum that is finally reached by the Newton approach. But if the initial solution is inappropriate, it is unable to improve the original solution by much.

4.3 Evaluation of Clustering Improvments

The impact of several improvements of the clustering step that have been proposed in Section 3.2 is studied here. The respective results are listed in Tab. 3. As in the previous sections, reductions from $N = 200$ to $L = 10$ components are evaluated.

At first, we regard the reallocation of sites to clusters, i.e., the outer loop of Algorithm 2 (lines 3 – 11), and repeat it 2 and 10 times, respectively, instead of only once. This has a large impact on the average running time but, in turn, the quality of the approximation improves as well. Unfortunately, this improvement is not enough to warrant the increased running time.

Then, we take the importance of single sites into account and only consider a fraction of all sites with the highest weights and try to reallocate them to a better cluster center. We use fractions of $W = 80\%$, 50% and 30%. Surprisingly, this has only a small impact on the normalized ISD. For $W = 80\%$ the approximation quality even im-

Table 1: KLD AND NORMALIZED ISD FOR THE APPROXIMATION RESULTS OF SEVERAL OF THE ALGORITHMS.

ALGORITHM	KLD $\times 10^{-3}$	normalized ISD
Runnals' algorithm	2.283 ± 0.920	3.606 ± 0.752
West's algorithm	3.002 ± 1.071	3.851 ± 0.790
Williams' algorithm	0.764 ± 0.465	2.257 ± 0.524
PGMR algorithm	0.388 ± 0.562	0.696 ± 0.204
GMRC algorithm	0.292 ± 0.494	0.658 ± 0.494

Table 2: RESULTS OF VARIANTS FOR GMRC, EACH WITH ONE STEP OF THE ALGORITHM SWITCHED OFF/ALTERED.

ALGORITHM	time t	normalized ISD
GMRC w. random init.	1.136 ± 0.045s	1.272 ± 1.561
GMRC w/o clustering	1.742 ± 0.043s	0.774 ± 0.872
GMRC w/o refinement	2.737 ± 0.036s	1.697 ± 0.432
Runnals' algorithm	1.678 ± 0.024s	3.606 ± 0.752
GMRC algorithm	2.793 ± 0.052s	0.658 ± 0.494

proves. Here, the refinement step probably reaches another local mimimum that is nearby.

Finally, we combine both improvement strategies and perform several iterations on a reduced amount of all sites. In each iteration the fraction of sites that is considerd, has been reevaluated again. The results indicate that using 10 iterations still takes inappropriately long but a combination of only 2 iterations and a smaller fraction of sites yields a good compromise between a decrease in running time and an improvement in approximation quality.

5 Conclusions and Future Work

The GMRC algorithm represents a top-down approach for Gaussian mixture reduction taking into account global information of the whole density function. It incorporates knowledge from theoretical computer science into this field of research. Our current implementation of the GMRC algorithm places itself ahead of previous top-down approaches like West and Williams and poses an interesting alternative to the PGMR algorithm.

Due to its modularity, different algorithms can be easily applied for the individual steps of GMRC. In doing so, different requirements (computational complexity, approximation quality, complexity of implementation, ...) on the reduction task can be met. By using Runnalls' algorithm during the preprocessing, similar results in terms of approximation quality as PGMR are achieved and the algorithm is considerably faster for input sizes of up to $N \approx 150$ components. By using West's algorithm instead, the running times are always well below PGMR and starting at $N \approx 300$ the approximation quality becomes similar or even better.

We are currently investigating further approaches to obtain good initial approximations on a faster timescale. Here, the preprocessing of the *k-means++* algorithm as described in [1] is already looking very promising by delivering appropriate solutions very fast. We are also in the process of extending our clustering algorithm for multivariate Gaussian mixture density functions to evaluate realistic problems. In addition, an adaptive variant of GMRC that uses a variable number of reduced mixture components L is also currently being developed and already shows promising results.

Replacing the clustering step by other effective clustering approaches such as *neural clustering* [13] which has a lower computational complexity than the k-means algorithm could provide interesting insights and further strengthen the modularity of GMRC by providing an additional option.

Table 3: RESULTS FOR VARIANTS OF GMRC, USING DIFFERENT IMPROVEMENT STRATEGIES OR COMBINATIONS.

ALGORITHM	time t	normalized ISD
GMRC w. 2 iterations	3.820 ± 0.046s	0.591 ± 0.407
GMRC w. 10 iterations	12.160 ± 0.050s	0.562 ± 0.361
GMRC w. 30%	2.051 ± 0.040s	0.680 ± 0.571
GMRC w. 50%	2.259 ± 0.041s	0.652 ± 0.536
GMRC w. 80%	2.566 ± 0.037s	0.637 ± 0.484
GMRC w. 30%, 2 it.	2.361 ± 0.041s	0.654 ± 0.504
GMRC w. 50%, 2 it.	2.771 ± 0.040s	0.619 ± 0.477
GMRC w. 80%, 2 it.	3.395 ± 0.042s	0.607 ± 0.455
GMRC w. 30%, 10 it.	4.851 ± 0.041s	0.633 ± 0.487
GMRC w. 50%, 10 it.	6.930 ± 0.042s	0.618 ± 0.460
GMRC w. 80%, 10 it.	10.090 ± 0.086s	0.594 ± 0.420
GMRC algorithm	2.793 ± 0.052s	0.658 ± 0.494

6 Acknowledgements

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

A Detailed Computations

Weight Optimization. A system of linear equations has to be solved in order to compute the weight optimization in Section 3.3. It is derived as follows.

To obtain weights $\underline{\omega}$ for the reduced Gaussian mixture $\underline{\eta}$ minimizing the ISD between the original and the reduced mixture density, we have to compute the first derivative of the ISD with respect to these weights

$$\begin{aligned}
\frac{\partial D}{\partial \underline{\omega}} =& \frac{\partial}{\partial \underline{\omega}} \left(\frac{1}{2} \int_{\mathbb{R}} \Big(\sum_{i=1}^{L} \omega_i \mathcal{N}_i - \sum_{j=1}^{N} \tilde{\omega}_i \tilde{\mathcal{N}}_j \Big)^2 \mathrm{d}x \right) \\
=& \int_{\mathbb{R}} \Big(\sum_{i=1}^{L} \omega_i \mathcal{N}_i - \sum_{j=1}^{N} \tilde{\omega}_j \tilde{\mathcal{N}}_j \Big) \cdot \Big[\mathcal{N}_1, \ldots, \mathcal{N}_L \Big]^{\mathrm{T}} \mathrm{d}x \\
=& \int_{\mathbb{R}} \sum_{i=1}^{L} \omega_i \mathcal{N}_i \sum_{k=1}^{L} \mathcal{N}_k \underline{e}_k \mathrm{d}x - \int_{\mathbb{R}} \sum_{j=1}^{N} \tilde{\omega}_j \tilde{\mathcal{N}}_j \sum_{k=1}^{L} \mathcal{N}_k \underline{e}_k \mathrm{d}x \\
=& \sum_{i=1}^{L} \sum_{k=1}^{L} \omega_i \underline{e}_k \int_{\mathbb{R}} \mathcal{N}_i \mathcal{N}_k \mathrm{d}x - \sum_{j=1}^{N} \tilde{\omega}_j \int_{\mathbb{R}} \sum_{k=1}^{L} \tilde{\mathcal{N}}_j \mathcal{N}_k \underline{e}_k \mathrm{d}x,
\end{aligned}$$

with $\underline{e}_k$ denoting the k^{th} unit vector and $\mathcal{N}_i = \mathcal{N}(x; \mu_i, \sigma_i^2)$. This term can be further transformed into a matrix notation

$$\frac{\partial D}{\partial \underline{\omega}} = \begin{bmatrix} \int_{\mathbb{R}} \mathcal{N}_1^2 \mathrm{d}x & & \\ & \ddots & \\ & & \int_{\mathbb{R}} \mathcal{N}_L^2 \mathrm{d}x \end{bmatrix} \underline{\omega} - \sum_{j=1}^{N} \tilde{\omega}_j \begin{bmatrix} \int_{\mathbb{R}} \tilde{\mathcal{N}}_j \mathcal{N}_1 \mathrm{d}x \\ \vdots \\ \int_{\mathbb{R}} \tilde{\mathcal{N}}_j \mathcal{N}_L \mathrm{d}x \end{bmatrix}.$$

Now, we can compute the roots of this derivative with respect to $\underline{\omega}$ by solving the system of linear equations,

$$\begin{bmatrix} \int_{\mathbb{R}} \mathcal{N}_1^2 \mathrm{d}x & & \\ & \ddots & \\ & & \int_{\mathbb{R}} \mathcal{N}_L^2 \mathrm{d}x \end{bmatrix} \underline{\omega} = \sum_{j=1}^{N} \tilde{\omega}_j \begin{bmatrix} \int_{\mathbb{R}} \tilde{\mathcal{N}}_j \mathcal{N}_1 \mathrm{d}x \\ \vdots \\ \int_{\mathbb{R}} \tilde{\mathcal{N}}_j \mathcal{N}_L \mathrm{d}x \end{bmatrix}.$$

The solution is a set of optimized weights $\underline{\omega}$ for the reduced Gaussian mixture density $\underline{\eta}$.

B Detailed Results

Table 4: EXECUTION TIMES AND NORMALIZED ISDS FOR THE ANALYZED APPROXIMATION ALGORITHMS.

	$N = 1000$		$N = 500$		$N = 200$	
ALGORITHM	time t	normalized ISD	time t	normalized ISD	time t	normalized ISD
Runnalls	47.248 ± 0.375s	3.347 ± 0.693	10.813 ± 0.575s	3.425 ± 0.656	1.678 ± 0.024s	3.606 ± 0.752
West	0.482 ± 0.038s	2.367 ± 0.367	0.168 ± 0.018s	2.820 ± 0.496	0.046 ± 0.001s	3.851 ± 0.790
Williams	–	–	–	–	42.333 ± 0.335s	2.257 ± 0.524
PGMR	18.438 ± 5.126s	0.721 ± 0.182	5.717 ± 2.817s	0.725 ± 0.182	2.356 ± 1.372s	0.696 ± 0.204
GMRC	59.302 ± 0.681s	0.428 ± 0.186	14.763 ± 0.256s	0.501 ± 0.256	2.793 ± 0.052s	0.658 ± 0.494
GMRC+West	10.839 ± 0.233s	0.448 ± 0.195	4.146 ± 0.490s	0.558 ± 0.322	1.181 ± 0.044s	0.863 ± 0.694
GMRC local	49.218 ± 0.488s	0.407 ± 0.407	11.033 ± 0.138s	0.512 ± 0.526	1.762 ± 0.042s	0.742 ± 0.853

	$N = 120$		$N = 40$	
ALGORITHM	time t	normalized ISD	time t	normalized ISD
Runnalls	0.606 ± 0.012s	3.603 ± 0.770	0.068 ± 0.002s	3.214 ± 0.926
West	0.025 ± 0.001s	4.611 ± 1.040	0.007 ± 0.000s	6.598 ± 1.765
Williams	9.704 ± 0.115s	2.360 ± 0.540	0.750 ± 0.011s	2.026 ± 0.639
PGMR	1.896 ± 1.276s	0.671 ± 0.217	1.805 ± 1.585s	0.569 ± 0.247
GMRC	1.240 ± 0.038s	0.754 ± 0.653	0.279 ± 0.027s	0.552 ± 0.723
GMRC+West	0.658 ± 0.037s	1.235 ± 1.041	0.226 ± 0.030s	1.780 ± 1.704
GMRC local	0.675 ± 0.034s	0.847 ± 0.997	0.120 ± 0.027s	0.687 ± 1.104

References

[1] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

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

[3] D. Cohn, Z. Ghahramani, and M. Jordan. Active Learning with Statistical Models. *Arxiv preprint cs.AI/9603104*, 1996.

[4] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering Large Graphs via the Singular Value Decomposition. *Machine Learning*, 56(1):9–33, 2004.

[5] M. Huber and U. Hanebeck. Progressive Gaussian Mixture Reduction. In *11th International Conference on Information Fusion, 2008*, pages 1–8, 2008.

[6] A. J. Izeman. Recent Developments in Nonparametric Density Estimation. *Journal of the American Statistical Association*, 86(413):205–224, 1991.

[7] S. Kullback and R. Leibler. On Information and Sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[8] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

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

[10] R. Rose and D. Reynolds. Text independent speaker identification using automatic acoustic segmentation. In *International Conference on Acoustics, Speech, and Signal Processing, 1990*, pages 293–296, 1990.

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

[12] D. Salmond. Mixture reduction algorithms for target tracking in clutter. In *Proceedings of SPIE*, volume 1305, page 434. SPIE, 1990.

[13] J. Schubert. Clustering belief functions based on attracting and conflicting metalevel evidence using potts spin mean field theory. *Information Fusion*, 5(4):309–318, 2004.

[14] D. Scott and W. Szewczyk. From Kernels to Mixtures. *Technometrics*, 43(3):323–335, 2001.

[15] M. West. Approximating posterior distributions by mixtures. *Journal of the Royal Statistical Society: Series B*, 55:409–409, 1993.

[16] J. Williams and P. Maybeck. Cost-function-based gaussian mixture reduction for target tracking. In *Proceedings of the Sixth International Conference on Information Fusion, 2003*, volume 2, 2003.