Increasing State Estimation Accuracy in the Inference Algorithm on a Hybrid Factor Graph Model

Mareike Stender[*], Mattis Hartwig[*†], Tanya Braun[‡], Ralf Möller[*]
[*]German Research Center for Artificial Intelligence, Lübeck
[†]singularIT GmbH, Leipzig
[‡]Computer Science Department, University of Münster, Münster

Abstract

We consider an intelligent agent that receives a continuous input from a signal-generating system and aims for estimating the discrete latent state of that system. The agent includes a switching linear dynamical system modeled as a hybrid factor graph for performing state estimation by determining the inference algorithm. We investigate the agent's performance in relation to two Gaussian mixture reduction methods restricting Gaussian mixture growing while message passing, namely, the naive pruning implemented in the past and the realization of the Kullback-Leibler (KL) discrimination based approach defined by Runnalls (2007). For evaluation, we use simulated data provided with various signal-to-noise ratios. Reviewing the metrics, the agent reaches an improvement in state estimation accuracy by using the KL discrimination based method with minimally increasing computational effort. The findings of our work let us take one step towards establishing intelligent systems for modeling real-world systems with switching behavior capable to analyze noisy data sets.

1 Introduction

Agents used in artificial intelligence often reach their capability limits due to data-heavy computations. To avoid overloading their processing capabilities, they may use approximations which may lead to a critical loss of information. In this context, we consider an agent investigating a real-world switching system to assist humans in planning actions in the future. The system generates a continuous signal based on its internal states. The agent receives this signal as continuous sensor input data. The agent then aims at identifying the discrete switches of the signal-generating system in the environment over time by estimating the system's latent state using the signal as evidence. The performance of the agent is measured based on accuracy and computation time. The mathematical model the agent uses for state estimations in this scenario is a hybrid factor graph that models a switching linear dynamical system (SLDS). The discrete states between which the system switches are modeled as categories of a discrete random variable and the continuous inputs as Gaussian distributions. Initially, for every discrete category in the discrete random variable, a single Gaussian distribution models the continuous latent state. While answering queries on the hybrid factor graph using an inference algorithm such as so called message passing, the initial Gaussians become Gaussian mixtures by multiplication and summation. In every time step, the number of Gaussian mixture components grows in an exponential way causing the agent to reach its capability limits and no longer be able to perform state estimation in a reasonable amount of time. To keep the agent's performance at a maximum, the exponential growth of the Gaussian mixtures has to be curbed by a way of Gaussian mixture reduction (GMR). Stender et al. (2021) use a naive pruning strategy to discard Gaussian mixture components of the Gaussian mixture that are presumed insignificant by deleting those Gaussian components with smallest weights. However, such a strategy can lead to fatal information loss as a Gaussian component containing a small weight could still carry an important amount of information in the Gaussian mixture. For example, a Gaussian component may contain a small weight and at the same time a small variance, implying a peaky Gaussian curve and thus a high information content in the Gaussian mixture. If, based on the small weight, this Gaussian component is removed from the mixture, the mixture loses a significant amount of information.

In this paper, we transfer the merging GMR method developed by Runnalls (2007) for multi-target tracking to the context of artificial intelligence, specifically an agent investigating a switching system. The merging GMR method is based on Kullback-Leibler (KL) discrimination, considering both the weights and the means and variances of the Gaussian mixture components. In particular, for deciding which pairs to merge, the dissimilarity between pairs of Gaussian components is calculated (Runnalls 2007). We investigate this method's suitability for maximizing the agent's performance regarding accuracy and computation time. We apply the merging GMR to the inference algorithm on the SLDS modeled in hybrid factor graphs, show the increase of accuracy and consider its computational effort in an example on simulated data.

The remainder of this paper is structured as follows: In Section 2, we cover related work regarding GMR. Afterwards, we briefly describe functionalities of hybrid factor graphs, the modeling of the SLDS based on the work of Stender et al. (2021), and the naive pruning approach. Section 4 deals with the implementation of Runnalls' merging GMR (Runnalls 2007) in hybrid factor graphs. We evaluate the benefit for the state estimation of the two GMR methods in a simulation example on the SLDS modeled in hybrid factor graphs in Section 5. We end this work with a summary of the findings, and conclude with future investigations.

2 Related Work

There are many approaches in the literature that address the challenge of reducing the number of Gaussian mixture components in Gaussian mixtures while preserving as much information as possible.

Assa and Plataniotis (2018) identify and merge similar Gaussian components using an averaging method based on the Wasserstein distance (WD) with the goal of preserving the geometric shape of the mixture. Williams and Maybeck (2003) use the Integral Square Difference (ISD) cost measure in a cost-function based method for GMR. The ISD distance enables a closed form evaluation of the cost function of GMR. Runnalls (2007) applies an upper bound on the KL discrimination as a measure of similarity to decide which Gaussian components to merge to reduce the Gaussian mixture to the desired number of components. In comparison to the KL discrimination based reduction method, the WD-based method reaches higher similarity to the original Gaussian mixture but results in a higher computational complexity. The ISD-based approach of Williams and Maybeck (2003) leads to 30% greater dissimilarity between the original and merged Gaussian mixture compared to the KL discrimination based merging method (Runnalls 2007). Comparing upper bound measures based on ISD, Square Distance (SD), and KL discrimination, Valverde, Tortós, and Terzija (2012) draw the conclusion that the KL upper bound discrimination is most efficient in terms of computational demands and accuracy.

Due to the findings presented in the literature, we have decided to apply Runnalls' merging GMR method for preventing exponential growth of the Gaussian mixtures while passing messages over the hybrid factor graph.

3 Gaussian Mixture Reduction in the Hybrid Factor Graph Model of the SLDS

In this section, we give an insight into the agent's mathematical model used for performing state estimation over time. We briefly introduce the functionalities of factor graphs, deal with the modeling of an SLDS as a hybrid factor graph, and depict the naive pruning by Stender et al. (2021). The following description of the functionalities of factor graphs is based on the work of Loeliger et al. (2007).

Fundamentals on Factor Graphs Factor graphs belong to the set of probabilistic graphical models, which encode a joint probability density function (JPDF) over all variables in a model as a product of local functions, which depend only on a subset of variables (Kschischang, Frey, and Loeliger 2001), exploiting the conditional independence structure of the probability density function. Such models mainly fulfill the goal to determine the marginal probability density function (MPDF) of a subset of variables given the joint probability density function (JPDF) encoded by the

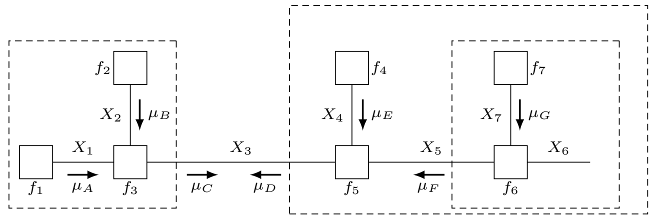

Figure 1: Graphical representation of Eq. (1) with $\overrightarrow{\mu}$ as forward and $\overleftarrow{\mu}$ backward messages on the edges towards X_3, where the sub-graphs are highlighted as dashed boxes.

model. Instead of solving the integral over all variables for computing MPDFs, we can apply message passing, which exploits the conditional independence structure of the probability density function, integrating variables in messages without ever realising the full joint.

In Fig. 1, a factor graph for the factorized JPDF

$$f(x_1, \ldots, x_7) = f_1(x_1)f_2(x_2)f_3(x_1, x_2, x_3)f_4(x_4) \cdot f_5(x_3, x_4, x_5)f_6(x_5, x_6, x_7)f_7(x_7) \quad (1)$$

of continuous random variables $X_1, \ldots, X_7$ is shown, where lowercase x_i refer to values each variable can take. The variables in factor graphs label the edges and the functions to apply to the variables – implying mathematical operations – are represented by nodes. The messages μ on the edges of the factor graph are determined by message passing. For setting up the message passing algorithm, we use the sum-product rule: For acyclic factor graphs, messages are computed from leaves inward, where the direction of the message is indicated by the definition and the mathematical operation of the used nodes.

The MPDF for the variable/edge X_3 in Fig. 1 is determined as the product of the forward message $\overrightarrow{\mu}_C(x_3)$ and the backward message $\overleftarrow{\mu}_D(x_3)$ on the edge X_3:

$$\overline{f}_3(x_3) \propto \overrightarrow{\mu}_C(x_3)\overleftarrow{\mu}_D(x_3) \quad (2)$$

The repeated application of the sum-product rule allows to combine nodes to sub-graphs (dashed boxes in Fig. 1) by eliminating variables.

Generally, the use of various types of distributions as messages is possible in factor graphs. Loeliger (2004) describes the basic nodes for the use of discrete or continuous (Gaussian) variables and enables the state estimation of either discrete or continuous latent system states.

SLDS in Hybrid Factor Graphs For modeling switching systems and allowing for estimating discrete and continuous latent system states, discrete and continuous variables need to be combined in so called hybrid factor graphs. Stender et al. (2021) model the SLDS – a general model of a switching system with wide area of application – in hybrid factor graphs, which represents an expansion of the findings of Loeliger (2004). They combine discrete and continuous variables to so-called cluster messages, which consist of a discrete and a continuous part, and define hybrid nodes enabling message passing with hybrid cluster messages. The cluster message – representing the combined distribution of the discrete random variable H_t and the continuous normal distributed random variable X_t – is defined as a tuple of the parameters weights $w_{X_i}^{(l)}$, means $m_{X_i}^{(l)}$, and variances $V_{X_i}^{(l)}$, depicting l sums of n Gaussian components, where i represents the respective Gaussian component in the sum, and l is the category of the discrete variable H_t. The semantics of a cluster message is given by:

$$p_{X,H}(x,h) = \sum_{i,l} w_{X_i}^{(l)} \delta\left(h - h^{(l)}\right) \mathcal{N}\left(x; m_{X_i}^{(l)}, V_{X_i}^{(l)}\right), \quad (3)$$

$$\text{with } w_{X_i}^{(l)} \geq 0.$$

Index l refers to the category of the discrete variable H_t. Each discrete category l corresponds to a Gaussian mixture in the cluster message (Stender et al. 2021).

Figure 2 shows the SLDS modeled as hybrid factor graph, where blue dashed edges/nodes are discrete, and red edges/nodes are continuous variables/functions (Stender et al. 2021). The observed variable is depicted as $\hat{Y}_t$ and modeled as Gaussian. The name H_t denotes the discrete and the name X_t the continuous latent variable, which are combined into the so-called cluster message. The continuous part of the cluster message is modeled as a Gaussian mixture, which is defined as a tuple of the parameters weights w_{X_i}, means m_{X_i}, and variances V_{X_i}, representing a sum of n Gaussian components, where i represents the respective Gaussian component in the sum. The semantics of a Gaussian mixture is:

$$p_X(x) = \sum_{i=1}^{n} w_{X_i} \cdot \mathcal{N}(x; m_{X_i}, V_{X_i}), \text{ with } w_{X_i} \geq 0. \quad (4)$$

The node $p_{H_t,X_t|H_{t-1},X_{t-1}}$ contains the transition model for the discrete as well as the continuous part of the cluster message, the node $p_{Y_t|H_t,X_t}$ contains the measurement model – thus the confidence in the measurement – and the node highlighted as "=" is called *equality* node, which can be understood as a branching point that enables the use of a variable in more than two nodes (Loeliger 2004). The variables are all indexed by a time step t and thus make up a time slice. For an interval of time points $[t_1, t_2]$, the time slice can be instantiated by replacing t with values within $[t_1, t_2]$. Arrows highlight the direction of the arithmetic operation of nodes. Nevertheless, the factor graph is still an undirected graph.

Exponential Gaussian Mixture Growing over Time To achieve the agent's goal of estimating the discrete latent system state of the signal-generating system, the message passing algorithm as the inference algorithm is run on the hybrid factor graph. The agent copies the hybrid factor graph (Fig. 2) time slice and receives the continuous sensor input as a signal in the node $\hat{Y}_t$ in every time step t. By querying the inference algorithm for the distribution of the estimated latent states, the agent invokes the application of the sum-product rule from the leaves inwards and the calculation of the estimation of the discrete and continuous latent state on the edge H_t'', X_t''.

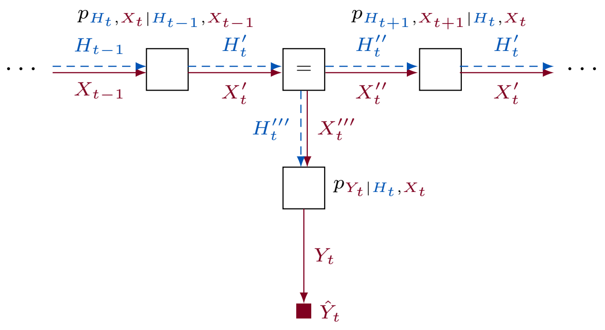

Figure 2: An SLDS modelled with a hybrid factor graph.

During message passing, due to the distributive property, the cluster message processing (multiplication and summation) results for each category l in a combination of every Gaussian component i_A in the Gaussian mixture of cluster message A with every Gaussian component i_B in the Gaussian mixture of cluster message B. Thus, the combined Gaussian mixtures in the cluster messages grow in an exponential way over time, i.e., with factor l in every time step t with l referring to the number of categories in the discrete random variable, resulting in increased computational costs for the agent. To ensure maximum performance of the agent, applying GMR methods is necessary to slow down the exponential growth of the Gaussian mixtures over time. The goal is to find an approximation of the original Gaussian mixture with fewer components while preserving as much information as possible and implement it as a so-called REDUCE function as part of the inference algorithm enabling a reduction induced by the agent at any time during message passing.

Naive Pruning Stender et al. (2021) use as the REDUCE function a naive pruning method, which discards Gaussian components of the Gaussian mixture presumed insignificant depending on the weight of the Gaussian component. Formally, given a Gaussian mixture with n Gaussian components and for preserving k (with $k < n$) Gaussian components, the naive pruning method searches for the Gaussian components with the smallest weights and discards them until only k Gaussian components remain in the Gaussian mixture. Afterwards, the weights of the remaining Gaussian components are re-normalized. On the one hand, discarding Gaussian components in this naive fashion has the advantage of being fast. However, on the other hand it can be a crude approximation and thus lead to a huge loss of information. Specifically, if relevant Gaussian components, which includes those with small variance, are discarded because of their minimal weight, the loss of information increases.

As mentioned before, the REDUCE function can be called at any time in the inference algorithm. Stender et al. (2021) use the naive pruning method as the REDUCE function and apply it in every time step t to the cluster messages after their combination on the edge H_t'', X_t'', because the edge connects the time steps t and the combination in the equality node lets the number of Gaussian components in the Gaussian mixtures grow in an exponential way.

Even though an improvement in computation time is achieved using this method by Stender et al. (2021), they still accept a non-negligible loss of information due to the approximation. The objective of our work is to present an improved REDUCE function modeled as a KL discrimination based merging GMR, decreasing the loss of information.

4 Implementation of an Improved Gaussian Mixture Reduction

In this section, we describe our improved merging GMR based on the work by Runnalls (2007) using KL discrimination. In particular, we show how to implement the KL discrimination based merging GMR as part of the REDUCE function for the inference algorithm on the hybrid factor graph in Fig. 2, before evaluating our new approach in the next section.

The purpose of replacing the naive pruning method with the merging method lies in minimizing the loss of information during GMR at the expense of a limited amount of computation time, for which the merging method poses as the most promising from the literature. The desired size of the Gaussian mixture is defined by k, analogous to the naive pruning method. In the merging method, the objective is to keep the KL discrimination between the original n-component Gaussian mixture and the merged k-component (with $k < n$) Gaussian mixture as small as possible. Since there is no closed-form expression for the KL discrimination of one Gaussian mixture to another, Runnalls (2007) makes use of an upper bound on the KL discrimination. This results in the dissimilarity measure $B(i, j)$ in one dimension as

$$\frac{2B(i,j)}{w_i + w_j} = \log\left[w_{i|ij} \left(\frac{V_i}{V_j}\right)^{w_{j|ij}} + w_{j|ij} \left(\frac{V_j}{V_i}\right)^{w_{i|ij}} + w_{i|ij} w_{j|ij} \frac{(m_i - m_j)^2}{V_i^{w_{i|ij}} V_j^{w_{j|ij}}} \right] \tag{5}$$

with

$$w_{i|ij} = \frac{w_i}{w_i + w_j} \quad \text{and} \quad w_{j|ij} = \frac{w_j}{w_i + w_j} \tag{6}$$

between two Gaussian components in the Gaussian mixture, which can be understood as the cost of merging two Gaussian components into one. The indices i and j of the weights, means, and variances w, m, V depict the moments of the $i-$th and $j-$th component of the Gaussian mixture to be reduced. Two Gaussian components are merged as follows:

$$w_{ij} = w_i + w_j \tag{7}$$

$$m_{ij} = w_{i|ij} m_i + w_{j|ij} m_j \tag{8}$$

$$V_{ij} = w_{i|ij} V_i + w_{j|ij} V_j + w_{i|ij} w_{j|ij} (m_i - m_j)^2 \tag{9}$$

For the reduction to k Gaussian components, the two Gaussian components with the smallest dissimilarity will be merged iteratively according to Eqs. (7) to (9) until the size

Function 1: Gaussian Mixture Reduction

```
Input: Gaussian mixture with $n$ Gaussian components, number $k$
 1: while $n > k$: do
 2:     for $i$ in Gaussian Mixture($n$) do
 3:         for $j$ in Gaussian Mixture($n$) do
 4:             Compute $B(i, j)$        ▷ Equations (5) and (6)
 5:         end for
 6:     end for
 7:     Search pair of components with smallest $B(i, j)$
 8:     Merge found pair                 ▷ Equations (7) to (9)
 9: end while
10: Normalize weights of Gaussian Mixture($k$)
Output: Gaussian mixture with $k$ Gaussian components
```

of the Gaussian mixture corresponds to k. After completing the merging task, the weights w of the Gaussian components are re-normalized as is done with naive pruning. The Gaussian Mixture reduction by merging based on the dissimilarity measure offers a more accurate approximation of the original Gaussian mixture, because it depends on all moments of the Gaussian components and not just the weight, and minimizes the information loss by calculating the merging costs. In Function 1, the implemented merging GMR method for the inference algorithm on SLDSs modeled in hybrid factor graphs is presented as pseudo code.

Analogous to the naive pruning approach of Stender et al. (2021), the KL discrimination based GMR method is modeled in the REDUCE function and can be called by the agent at any time in the message passing. The decision on when to reduce the Gaussian mixtures during inference is an important one that should not be underestimated. Calling the REDUCE function too frequently may lead to a blurred state estimation especially on noisy data. A less frequent reduction results in low performance indicated by high computational effort. For increased estimation accuracy and useful evaluation with respect to the naive pruning approach, in our work the agent calls the REDUCE function at the same point in the inference algorithm as it is done in the approach by Stender et al. (2021) on the edge H_t'', X_t'' (Fig. 2).

Next, we compare the naive pruning method with the merging method regarding accuracy and computation time in a case study.

5 Empirical Evaluation

This evaluation focuses on the aspects of accuracy and computation time regarding the agent's performance using the merging method versus the naive pruning method in the REDUCE function.

As we are interested in accuracy, we need ground truth and an identical setup to compare the results of the two GMR methods against. Therefore, we use simulated data. Furthermore, the use of simulated data ensures that we cover borderline cases such as worst and best case scenarios, which may not be covered by real data sets.

To evaluate the GMR methods for a real-world system, which in many cases is a switching system, we need to simulate a combination of discrete and continuous variables. We use the inference algorithm on the hybrid factor graph for evaluating the state estimation on 250 data points over time implemented with empirical selected parameters. The measurement signal is created by forward simulation of the mathematical model of the SLDS, refer to Stender et al. (2021) for the model definition. The SLDS is simulated with a discrete latent state that consists of three categories. The switches between the categories are modeled by switching process noise mean ($\mu = -1, 0, 1$, with constant standard deviation $\sigma = 0.01$) in the modeled system. The time sequence of the switches is given by the preset time sequence of the discrete state. The continuous transition model in the transition node $p_{H_t,X_t|H_{t-1},X_{t-1}}$ for cluster messages is defined constantly as $A = 0.9$ for every discrete category, implying that in the current time step the inference algorithm takes into account a large amount of information of the prediction from the previous time step. The discrete transition model in the transition node $p_{H_t,X_t|H_{t-1},X_{t-1}}$ is given by the following matrix

$$T = \begin{bmatrix} 0.990 & 0.005 & 0.005 \\ 0.005 & 0.990 & 0.005 \\ 0.005 & 0.005 & 0.990 \end{bmatrix} \tag{10}$$

which represents the probability that the system switches between the discrete categories in the next time step t. We simulate a system that is more likely to remain in a discrete category than to switch between the three categories. The measurement model in node $p_{Y_t|H_t,X_t}$ is set as $C = 1$ for every discrete category, which implies high trust in the measurements. The measurement noise is modeled with a standard deviation of $\sigma = 0.8$ (mean $\mu = 0$).

In Figs. 3 and 4, exemplary state estimations on simulated data using the naive pruning method by Stender et al. (2021) and using the merging method presented in this paper is depicted, respectively. Real-world systems are subject to uncertainties that must be taken into account already in the evaluation based on a simulation. We add a signal-to-noise ratio (SNR) of $SNR = 15$ to the simulated signal for best visual representation of the distinction between the estimation results of the two GMR methods.

In both figures, the top plot represents the simulated signal, which is used as measurements in the node $\hat{Y}_t$ in the hybrid factor graph in Fig. 2. The middle plot shows the real (simulated) continuous state as solid black line and the result of the inference algorithm on the hybrid factor graph – the estimated continuous state – as dashed dark red line. The bottom plot represents the estimation of the discrete state in gray scales and the real (simulated) discrete state as solid blue line. The state estimation for the latent continuous and discrete states are calculated by the inference algorithm on the edge H_t'', X_t'' (Fig. 2). The plots in Fig. 3 show more frequent large artefacts in the estimates of the latent continuous and discrete state than the plots in Fig. 4, which is induced by the loss of information when using the naive pruning GMR.

For analyzing state estimation under uncertainties, we add different SNRs to the measurement signal and use precision and recall as metrics for evaluation. We consider the mean

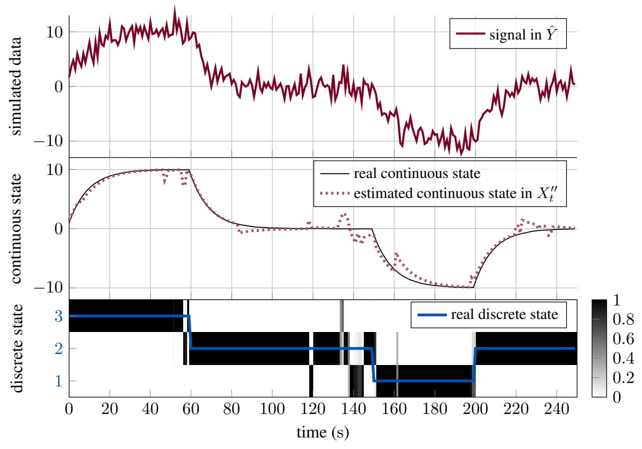

Figure 3: State estimation on simulated data using the naive pruning approach as the REDUCE function in the inference algorithm on the hybrid factor graph of Fig. 2.

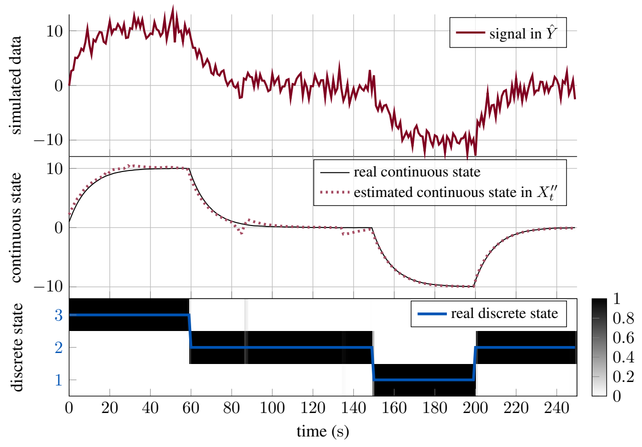

Figure 4: State estimation on simulated data using the merging method as the REDUCE function in the inference algorithm on the hybrid factor graph of Fig. 2.

deviation of the difference of the state estimation with the naive pruning and the merging method in percentage points depending on the number (#) of preserved Gaussian components and SNR as shown in Table 1. It turns out that preserving only two Gaussian components in the Gaussian mixture, the merging GMR enables up to 2.47 percentage points more precise state estimation whereas the accuracy for four or eight preserved Gaussian components is similar with the naive pruning. The reason for the more precise state estimation in the setting with two preserved Gaussian components is that the fewer Gaussian components are retained, the less information is retained. Reducing the Gaussian mixture to two Gaussian components by discarding Gaussian components leads to an increased loss of information. In the case of preserving four or eight Gaussian components in the Gaussian mixture, both methods lead to qualitatively suitable results. Since preserving a high number of Gaussian components results in higher computational costs, in practice, the

SNR: \ #:	2	4	8
15	2.47	0.47	0.22
20	1.98	0.15	0.19
25	2.12	0.14	0.11
30	1.75	0.10	0.09
35	1.51	0.18	0.18
40	1.99	0.18	0.08
45	1.55	0.15	0.10
50	1.76	0.18	0.07
55	2.29	0.17	0.16
60	2.05	0.23	0.15

Table 1: Evaluation of state estimation accuracy for different SNR settings and numbers of preserved Gaussian components.

GMR: \ #:	2	4	8
Pruning	12.04s	44.28s	389.51s
Merging	13.20s	49.95s	442.47s
Ratio	1.096	1.128	1.136

Table 2: Evaluation of computational time of both GMR methods for different numbers of preserved Gaussian components.

goal is to preserve as few Gaussian components as possible.

Table 2 depicts the computational times of the GMR methods naive pruning and KL discrimination based merging depending on the number of preserved Gaussian components, which shows that the merging method takes more time but only at a factor of around 1.1.

In summary, there is a trade-off between computational costs and accuracy but the more accurate state estimation costs only a factor of 1.1 of computing time, which is a reasonable trade-off. Thus, we succeed in avoiding information loss without bringing the agent to the limits of its computation capabilities. Additionally, the new merging GMR provides more accurate state estimation results independent of the SNR.

6 Conclusion

This paper presents an improved Gaussian mixture reduction modeled as a REDUCE function in the inference algorithm used on a hybrid factor graph modeling a switching linear dynamical system. The improved reduction method is based on KL discrimination and achieves better accuracy by considering all moments of Gaussian components. Due to the modularity of factor graphs, the merging GMR method can be used in any kind of hybrid factor graph without adjustment. The evaluation based on a simulated data set equipped with various SNRs shows a higher state estimation accuracy using the KL discrimination based merging GMR, especially while reducing the Gaussian mixture to two Gaussian components at the expense of a increased computational time. Thus, we have taken another step towards developing intelligent systems that let us model switching systems and now perform state estimation with higher accuracy.

For future work, we are interested in dealing with applying the findings of this work to real-world data and advancing the inference algorithm by implementing automated parameter learning in hybrid factor graphs.

References

Assa, A., and Plataniotis, K. N. 2018. Wasserstein-distance-based gaussian mixture reduction. *IEEE Signal Processing Letters* 25(10):1465–1469.

Kschischang, F. R.; Frey, B. J.; and Loeliger, H.-A. 2001. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory* 47(2):498–519.

Loeliger, H.-A.; Dauwels, J.; Hu, J.; Korl, S.; Ping, L.; and Kschischang, F. R. 2007. The factor graph approach to model-based signal processing. *Proceedings of the IEEE* 95(6):1295–1322.

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

Runnalls, A. R. 2007. Kullback-leibler approach to gaussian mixture reduction. *IEEE Transactions on Aerospace and Electronic Systems* 43(3):989–999.

Stender, M.; Graßhoff, J.; Braun, T.; Möller, R.; and Rostalski, P. 2021. A hybrid factor graph model for biomedical activity detection. In *2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI)*, 1–4.

Valverde, G.; Tortós, J. Q.; and Terzija, V. 2012. Comparison of gaussian mixture reductions for probabilistic studies in power systems. In *2012 IEEE Power and Energy Society General Meeting*, 1–7.

Williams, J., and Maybeck, P. 2003. Cost-function-based gaussian mixture reduction for target tracking. In *Sixth International Conference of Information Fusion, 2003. Proceedings of the*, volume 2, 1047–1054.