Bootstrapped Gaussian Mixture Model-Based Data-Driven Forward Stochastic Reachability Analysis

Joonwon Choi, *Graduate Student Member, IEEE*, Hyunsang Park, *Graduate Student Member, IEEE*, and Inseok Hwang, *Member, IEEE*

***Abstract*—We propose a data-driven forward stochastic reachability analysis algorithm for a system with unknown dynamics. In this letter, we assume a limited number of trajectory data is available and one cannot obtain additional data from the target system. The proposed algorithm learns the evolution of the state probability density function (pdf) as a Gaussian mixture model (GMM) from the given trajectory data and computes the pdf of the future state at a desired future time instance. We leverage the bootstrapping algorithm to account for the parameter estimation error of the GMM by computing the confidence interval of the estimated parameters. Then, the bootstrapped GMM is synthesized by selecting the optimal parameters within the confidence interval that yields the most informative model, thereby providing more reliable prediction results. The proposed algorithm is demonstrated via both numerical simulations and human subject experiments.**

***Index Terms*—Gaussian mixture model, reachability analysis, data-driven modeling.**

I. Introduction

THE FORWARD reachable set is a collection of all the possible states that a target system can reach at a specific future time instance. Using the forward reachable set, one can verify the safety of the target system by thoroughly examining its future operation. Thanks to this benefit, the reachability analysis has been widely used for the safety verification of a system, particularly a safety-critical system such as a human-in-the-loop system [1]. Nevertheless, the conventional reachability analysis methods rely on the knowledge of the dynamics model and thus, it is challenging to apply them to systems with unknown (black-box) elements [2].

The data-driven reachability analysis, on the other hand, does not require specific knowledge about the target system to obtain a feasible result. The data-driven reachability analysis leverages the trajectory data generated from the target system to compute a reachable set, making it suitable for a system with black-box elements such as a system with human intervention [2]. However, many existing algorithms assume one can obtain arbitrarily many data points from the target system [3], which might not be guaranteed in practice. Especially when the data is difficult or expensive to obtain, such as human subject experiments that require specific types of equipment with strict variable and scenario setting [4], one needs to compute the reachable set only using a limited amount of trajectory data. Meanwhile, some existing data-driven reachability analysis algorithms might not be able to explicitly consider the uncertainty inherent in a target system. For instance, in [5], the Koopman operator-based method was proposed but it is vulnerable to uncertainty or noisy data. In [6], a matrix zonotope is computed to enclose the uncertainty, which might yield overly conservative results [2].

Motivated by the aforementioned issues, we propose a data-driven forward stochastic reachability analysis algorithm for a system whose dynamics is unknown and only a limited number of trajectory data is available. The objective of the proposed algorithm is to compute the forward stochastic reachable set which is inferred as a state probability density function (pdf) at the desired future time instance. We assume the given trajectory data is the only source of the information, i.e., there is no side information.

Unlike many existing data-driven reachability analysis algorithms that directly retrieve the dynamics, the proposed algorithm learns the evolution of the state pdf as a Gaussian mixture model (GMM) using the expectation maximization (EM) algorithm. Then, the reachable set is computed based on the trained GMM. The GMM has been widely used for various applications to infer the characteristics of a target system from its trajectories. For example, it has been applied to model the behavior of complex systems, such as a human driver [4], and to imitate and predict the behavior of a target system [7]. By using the trained GMM instead of the dynamics model, one can obtain a feasible prediction result while considering the effect of uncertainty by leveraging only a set of trajectories.

One major issue that should be addressed in our approach is the accuracy of the trained GMM, since the fidelity of the GMM directly impacts the accuracy of the resulting reachable set. Although the EM algorithm is shown to provide sufficient

Manuscript received 15 September 2023; revised 18 November 2023; accepted 10 December 2023. Date of publication 25 December 2023; date of current version 22 January 2024. This work was supported by NSF under Grant CNS-1836952. Recommended by Senior Editor T. Oomen. *(Corresponding author: Joonwon Choi.)*

The authors are with the School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN 47907 USA (e-mail: choi774@purdue.edu; park1375@purdue.edu; ihwang@purdue.edu).

Digital Object Identifier 10.1109/LCSYS.2023.3347188

performance to train a GMM in many applications, it does not provide information regarding the parameter (e.g., mean and covariance) estimation errors [8]. This leads to the need for additional methods to estimate and account for such errors. Furthermore, one needs to fully utilize the information inherent in the given trajectory data as there is no supplementary information to infer the characteristics of the target system. One straightforward approach is using the information matrix, but the information matrix-based approaches require a large amount of data for a GMM and the error tends to be underestimated, making it not suitable for investigating the parameter estimation error of a GMM [8].

To tackle this problem, we estimate the parameter estimation error of the GMM using the bootstrapping algorithm. The bootstrapping algorithm is a resampling method that can quantify the characteristics of the estimated parameter such as the standard error [9]. In this letter, we utilize the bootstrapping algorithm to specify the confidence interval of the estimated parameter. Then, the optimal value within the interval that yields the most informative result is selected and composes a new GMM which we called as *bootstrapped GMM*. The future state pdf is then computed by propagating the current state pdf based on the bootstrapped GMM, thereby generating a more informative reachable set compared to the original GMM.

Our contributions in this letter are as follows: 1) We propose a data-driven forward stochastic reachability analysis algorithm that learns the evolution of the state pdf as a GMM. The proposed algorithm can be applicable when only a limited number of trajectory data is available and there is no chance to obtain additional data; and 2) We synthesize the bootstrapped GMM by leveraging the bootstrapping algorithm to consider the parameter estimation error of the GMM. Although no theoretical analysis is provided in this letter, we show that the proposed algorithm can generate a more informative reachable set through numerical simulations and human subject experiments.

The rest of this letter is organized as follows: In Section II, the bootstrapped GMM is presented. Section III provides the details on how the data-driven forward stochastic reachable set is computed. In Section IV, the numerical simulation results and human subject experiment results are presented and discussed. Lastly, the conclusion is given in Section V.

II. Bootstrapped Gaussian Mixture Model

A. Learning Transition Kernel as Gaussian Mixture Model

In this letter, we consider a general nonlinear discrete-time dynamical system driven by a process noise,

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{v}_k), \tag{1}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ is the state vector at time step k, $\boldsymbol{v}_k \in \mathbb{R}^l$ is the process noise, and f is the nonlinear dynamics of the system which is unknown. Note that (1) also can be seen as a closed-loop dynamics of a system controlled by a state-feedback control law. In (1), the evolution of the state pdf over time can be expressed using the Chapman-Kolmogorov equation [10],

$$P(\boldsymbol{x}_{k+1}) = \int P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)P(\boldsymbol{x}_k)d\boldsymbol{x}_k, \tag{2}$$

where $P(\boldsymbol{x}_k)$ is the state pdf at time step k and $P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)$ is called the *transition kernel* [11]. The objective of the proposed algorithm is to compute the future state pdf of an unknown system (1) at the desired future time step $T_d > 0$, $P(\boldsymbol{x}_{T_d})$, by solving (2). To this end, we first learn the evolution of the state pdf as a form of a GMM using the trajectory data of a target system.

The GMM is a convex combination of multiple Gaussian distributions. The trained GMM, $\mathcal{G}$, represents a joint pdf between $\boldsymbol{x}_f = [\boldsymbol{x}_1^T, \boldsymbol{x}_2^T, \ldots, \boldsymbol{x}_{N_D}^T]^T$ and $\boldsymbol{x}_p = [\boldsymbol{x}_0^T, \boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_{N_D-1}^T]^T$, $P(\boldsymbol{x}_f, \boldsymbol{x}_p)$, where N_D is the number of the data points in the given trajectory data. Let $\mathcal{G}$ be composed of M number of Gaussian components and its $i-$th component has mean $(\hat{\boldsymbol{\mu}}_i)$ and covariance $(\hat{\Sigma}_i)$ as follows [12]:

$$\hat{\boldsymbol{\mu}}_i = \begin{pmatrix} \hat{\boldsymbol{\mu}}_i^f \\ \hat{\boldsymbol{\mu}}_i^p \end{pmatrix}, \hat{\Sigma}_i = \begin{pmatrix} \hat{\Sigma}_i^f & \hat{\Sigma}_i^{f,p} \\ \hat{\Sigma}_i^{p,f} & \hat{\Sigma}_i^p \end{pmatrix}, \tag{3}$$

where $\hat{\boldsymbol{\mu}}_i^f \in \mathbb{R}^n$ is the mean fraction corresponding to $\boldsymbol{x}_f$, $\hat{\boldsymbol{\mu}}_i^p \in \mathbb{R}^n$ is that of $\boldsymbol{x}_p$, and $\hat{\Sigma}_i^f \in \mathbb{R}^{n\times n}$, $\hat{\Sigma}_i^p \in \mathbb{R}^{n\times n}$, $\hat{\Sigma}_i^{f,p} \in \mathbb{R}^{n\times n}$, and $\hat{\Sigma}_i^{p,f} \in \mathbb{R}^{n\times n}$ are the corresponding covariance fractions, respectively. Then, we can define $\mathcal{G}$ as

$$\mathcal{G} \coloneqq P(\boldsymbol{x}_f, \boldsymbol{x}_p) = \sum_i^M \hat{\pi}_i N\left(\hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i\right). \tag{4}$$

where $\hat{\pi}_i$ is the weight of each Gaussian component and $N(\hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i)$ is the Gaussian distribution with mean $\hat{\boldsymbol{\mu}}_i$ and covariance $\hat{\Sigma}_i$. The weight, mean, and covariance of $\mathcal{G}$ can be computed by using the EM algorithm while the number of the Gaussian components M can be chosen using a proper model selection criterion, such as the Bayesian information criterion (BIC) [12].

Once $\mathcal{G}$ is trained, one can compute the transition kernel $P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)$ using the Gaussian mixture regression (GMR) from $\mathcal{G}$, which is necessary to solve (2). Nevertheless, $\mathcal{G}$ trained using the conventional EM algorithm might include errors in its estimated parameters (mean and covariance) that need to be considered to obtain more informative prediction results.

B. Bootstrapping Model Parameters

In this subsection, we apply the bootstrapping algorithm to compute the errors on the estimated parameter of $\mathcal{G}$. Let $\hat{\theta}$ be the estimated parameter of $\mathcal{G}$ computed with the EM algorithm which can be defined as

$$\hat{\theta} = \left[\hat{\boldsymbol{\mu}}_1, \hat{\boldsymbol{\mu}}_2, \ldots, \hat{\boldsymbol{\mu}}_M, \hat{\Sigma}_1(1,1), \hat{\Sigma}_1(1,2), \ldots, \hat{\Sigma}_M(n,n)\right], \tag{5}$$

where $\hat{\Sigma}_i(j,k)$ is the (j,k) element of the covariance matrix of the $i-$th Gaussian component. We utilize the nonparametric bootstrapping algorithm to account for the parameter estimation errors of $\hat{\boldsymbol{\mu}}_i$ and $\hat{\Sigma}_i$. Let $\mathcal{T} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{N_D}\}$ be the given trajectory data where $\boldsymbol{y}_i = [\boldsymbol{x}_{f,i}^T, \boldsymbol{x}_{p,i}^T]^T$. $\boldsymbol{x}_{f,i}$ and $\boldsymbol{x}_{p,i}$ are the $i-$th element of $\boldsymbol{x}_f$ and $\boldsymbol{x}_p$, respectively. Then, the bootstrapping algorithm operates with the following steps [9]:

1) N_B bootstrap resamples $\{\mathcal{T}_{B,1}, \mathcal{T}_{B,2}, \ldots, \mathcal{T}_{B,N_B}\}$ are constructed by drawing N_D random observations with replacement from $\mathcal{T}$, i.e., compose $\mathcal{T}_{B,i} = \{\boldsymbol{y}_1^*, \boldsymbol{y}_2^*, \ldots, \boldsymbol{y}_{N_D}^*\}$ $\forall i = 1, 2, \ldots, N_B$ where $\boldsymbol{y}_{(\cdot)}^*$ is randomly selected from the element of $\mathcal{T}$.

2) For each $i = 1, 2, \ldots, N_B$, compute

$$\hat{\theta}_{B,i} = [\hat{\boldsymbol{\mu}}^i_{B,1}, \hat{\boldsymbol{\mu}}^i_{B,2}, \ldots, \hat{\boldsymbol{\mu}}^i_{B,M}, \hat{\Sigma}^i_{B,1}(1,1), \hat{\Sigma}^i_{B,1}(1,2), \ldots, \hat{\Sigma}^i_{B,M}(n,n)] \quad (6)$$

by feeding $\mathcal{T}_{B,i}$ to the EM algorithm.

3) Compute the sample mean and covariance of the parameters for each $i = 1, 2, \ldots, M$. For the mean of the $i-$th Gaussian component, we can compute

$$s_{B,i} = \frac{1}{N_B - 1} \sum_{l=1}^{N_B} \left(\hat{\boldsymbol{\mu}}^l_{B,i} - \bar{\boldsymbol{\mu}}_{B,i}\right)\left(\hat{\boldsymbol{\mu}}^l_{B,i} - \bar{\boldsymbol{\mu}}_{B,i}\right)^T, \quad (7)$$

where $\bar{\boldsymbol{\mu}}_{B,i}$ is defined as

$$\bar{\boldsymbol{\mu}}_{B,i} = \frac{1}{N_B} \sum_{k=1}^{N_B} \hat{\boldsymbol{\mu}}^l_{B,i}, \quad (8)$$

and the same for the covariance,

$$s'_{B,i}(j,k) = \frac{1}{N_B - 1} \sum_{l=1}^{N_B} \left(\hat{\Sigma}^l_{B,i}(j,k) - \bar{\Sigma}_{B,i}(j,k)\right)^2, \quad (9)$$

$$\bar{\Sigma}_{B,i}(j,k) = \frac{1}{N_B} \sum_{l=1}^{N_B} \hat{\Sigma}^l_{B,i}(j,k). \quad (10)$$

Using (7)-(10), one can analyze the desired properties, such as the bias or standard error, of the estimated parameter $\hat{\theta}$.

In the following section, we compute the confidence interval using the result of the bootstrapping algorithm to explicitly account for the parameter estimation error.

C. Computing Confidence Interval to Account for Parameter Estimation Error

In this letter, we consider the normal approximation, i.e., we assume the parameter estimation error is normally distributed and compute the normal confidence interval. Then, from (7)-(10), one can achieve the $1-\delta$ confidence interval of covariance elements $[\underline{\Sigma_{B,i}}(j,k), \overline{\Sigma_{B,i}}(j,k)]$ with [9]

$$\underline{\Sigma_{B,i}}(j,k) = \hat{\Sigma}_i(j,k) - \beta'_i(j,k) - Z_{\frac{\delta}{2}}\sqrt{s'_{B,i}(j,k)}, \quad (11a)$$

$$\overline{\Sigma_{B,i}}(j,k) = \hat{\Sigma}_i(j,k) - \beta'_i(j,k) + Z_{\frac{\delta}{2}}\sqrt{s'_{B,i}(j,k)}, \quad (11b)$$

for all $i = 1, 2, \ldots, M$ and $j, k = 1, 2, \ldots, n$, where $\delta > 0$ is a design variable, $Z_{(\cdot)}$ is the critical value of the standard normal distribution, and $\beta'_i(j,k) = \bar{\Sigma}_{B,i}(j,k) - \hat{\Sigma}_i(j,k)$ is the bias of (j,k) element of covariance.

A similar approach can be applied to the mean, but the parameter estimation error on the mean can be easily addressed by treating $\hat{\boldsymbol{\mu}}_i$ as a Gaussian random variable as opposed to computing its confidence interval. More specifically, we can assume that the estimated mean $\hat{\boldsymbol{\mu}}_i$ is distributed by $N(\boldsymbol{\mu}_i + \boldsymbol{\beta}_i, s_{B,i})$ from the normal approximation, where $\boldsymbol{\beta}_i = \bar{\boldsymbol{\mu}}_{B,i} - \hat{\boldsymbol{\mu}}_i$ is the bias of the mean and $\boldsymbol{\mu}_i$ is the true mean value [9]. Accordingly, the $i-$th Gaussian component of $\mathcal{G}$ can be seen as a Gaussian distribution with mean distributed by another Gaussian distribution, which is again a Gaussian distribution. As a result, we can define a new GMM that encompasses both mean and covariance parameter estimation errors as

$$\mathcal{G}_B = \sum_i^M \hat{\pi}_i N\left(\hat{\boldsymbol{\mu}}_{B,i}, \left[\underline{\Sigma_{B,i}}, \overline{\Sigma_{B,i}}\right] + s_{B,i}\right) \quad (12)$$

where $\hat{\boldsymbol{\mu}}_{B,i} = \hat{\boldsymbol{\mu}}_i - \boldsymbol{\beta}_i$. One can explicitly account for both mean and covariance parameter estimation error by using $\mathcal{G}_B$ instead of $\mathcal{G}$.

In the following subsection, we construct the bootstrapped GMM $\mathcal{G}^*_B$ by optimizing its parameter $\hat{\theta}^*_B$ to induce the most informative result among all the possible combinations within the confidence interval (11a) and (11b).

D. Maximizing the Volume of Confidence Ellipsoid

The confidence ellipsoid ($\mathcal{E}$) of a random variable $\boldsymbol{x} \sim N(\boldsymbol{\mu}, \Sigma)$ is defined as [13]

$$\mathcal{E}(\boldsymbol{\mu}, \Sigma c) = \{\boldsymbol{x} \in \mathbb{R}^n | (\boldsymbol{x} - \boldsymbol{\mu})^T (\Sigma c)^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \leq 1\} \quad (13)$$

where $c \in \mathbb{R}$ is a constant. The center of $\mathcal{E}(\boldsymbol{\mu}, \Sigma c)$ is located at the mean of a Gaussian distribution and it has the shape matrix as the multiplication of the covariance matrix and c. If we choose c as

$$c = \left(F_{\chi^2}\right)^{-1}(1-\delta, n) \quad (14)$$

where $F_{\chi^2}(\cdot, n)$ is a cumulative distribution function (cdf) of χ^2 distribution with n degree of freedom, the probability that $\boldsymbol{x}^T\boldsymbol{x}$ to be included in $\mathcal{E}(\boldsymbol{\mu}, \Sigma c)$ becomes $1-\delta$ [13].

The idea of the proposed algorithm is to find the optimal value in $[\underline{\Sigma_{B,i}}(j,k), \overline{\Sigma_{B,i}}(j,k)]$ for all $j, k = 1, 2, \ldots, n$ that maximizes the volume of each confidence ellipsoid. Since all the values within the confidence interval are still valid candidates, it is desirable to select the most informative one. However, simply increasing the volume might not necessarily lead $\mathcal{G}^*_B$ to be more informative, but on the contrary, it could lead to the loss of information. For instance, if the confidence ellipsoid of $\mathcal{G}^*_B$ has a shorter length in a certain axis than $\mathcal{G}$, the information regarding this axis would be lost even though the volume is larger. To tackle this problem, we design a constrained optimization problem that maximizes the volume of the confidence ellipsoids while enforcing each confidence ellipsoid of $\mathcal{G}^*_B$ to include that of $\mathcal{G}$.

Let the confidence ellipsoid of the $i-$th Gaussian components in $\mathcal{G}$ as $\mathcal{E}_{\mathcal{G},i}(\hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i c)$ and that of $\mathcal{G}^*_B$ as $\mathcal{E}_{\mathcal{G}^*_B,i}(\hat{\boldsymbol{\mu}}_{B,i}, \hat{\Sigma}^*_{B,i}c + s_{B,i}c)$ where $\hat{\Sigma}^*_{B,i}(j,k) \in [\underline{\Sigma_{B,i}}(j,k), \overline{\Sigma_{B,i}}(j,k)]$ is the optimization variable. From [14], $\mathcal{E}_{\mathcal{G}^*_B,i}$ includes $\mathcal{E}_{\mathcal{G},i}$ if there exists $\lambda \geq 0$ that satisfies the following $\forall \boldsymbol{x} \in \mathcal{E}_{\mathcal{G},i}$:

$$\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_{B,i}\right)^T \left(\hat{\Sigma}^*_{B,i}c + s_{B,i}c\right)^{-1} \left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_{B,i}\right) - \lambda\left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i\right)^T \left(\hat{\Sigma}_i c\right)^{-1} \left(\boldsymbol{x} - \hat{\boldsymbol{\mu}}_i\right) \leq 1 - \lambda. \quad (15)$$

Equation (15) can be rewritten as a corresponding LMI condition. As a result, one can maximize the volume of the ellipsoid by finding the optimal $\hat{\Sigma}^*_{B,i}$ that maximizes the trace of $\hat{\Sigma}^*_{B,i}$ while satisfying the constraints [13]

$$\max_{\hat{\Sigma}^*_{B,i}, \lambda} \quad \mathrm{tr}\left(\hat{\Sigma}^*_{B,i}\right) \quad (16a)$$

subject to

$$\begin{bmatrix} -\left(\hat{\Sigma}^*_{B,i}c + s_{B,i}c\right) & \hat{\boldsymbol{\mu}}_{B,i} - \hat{\boldsymbol{\mu}}_i & \left(\hat{\Sigma}_i c\right)^{1/2} \\ \left(\hat{\boldsymbol{\mu}}_{B,i} - \hat{\boldsymbol{\mu}}_i\right)^T & \lambda - 1 & 0 \\ \left(\hat{\Sigma}_i c\right)^{1/2} & 0 & -\lambda I \end{bmatrix} \leq 0 \tag{16b}$$

$$\lambda \geq 0 \tag{16c}$$

$$\hat{\Sigma}^*_{B,i} > 0 \tag{16d}$$

$$\underline{\Sigma_{B,i}}(j,k) \leq \hat{\Sigma}^*_{B,i}(j,k) \leq \overline{\Sigma_{B,i}}(j,k) \;\forall\, j,k = 1,2,\ldots,n \tag{16e}$$

From (16a)-(16e), we define $\mathcal{G}^*_B$ as

$$\mathcal{G}^*_B = \sum_i^M \hat{\pi}_i N\left(\hat{\boldsymbol{\mu}}_{B,i}, \hat{\Sigma}^*_{B,i} + s_{B,i}\right). \tag{17}$$

Consequently, one can compute the reachable set while explicitly accounting for the parameter estimation error by incorporating (17).

Remark 1: The feasibility of the normal approximation can be assessed by using the quantile-quantile plot [9]. If the normal approximation is not feasible, one can compute the percentile confidence interval instead [9]. Then, the parameter estimation error in the mean can be considered by generating a new confidence ellipsoid $\mathcal{E}'_i$ that includes $\mathcal{E}_{\mathcal{G},i}(\hat{\boldsymbol{\mu}}_i, \hat{\Sigma}_i c) \bigoplus B_{r,i}$ $\forall\, i = 1,2,\ldots,M$, where $\bigoplus$ is the Minkowski sum, $B_{r,i}$ is a ball whose radius $r = max(|\hat{\boldsymbol{\mu}}_i - \overline{\boldsymbol{\mu}_{B,i}}|, |\hat{\boldsymbol{\mu}}_i - \underline{\boldsymbol{\mu}_{B,i}}|)/2$, and $\overline{\boldsymbol{\mu}_{B,i}}$ and $\underline{\boldsymbol{\mu}_{B,i}}$ are the corresponding percentile confidence bounds. Such $\mathcal{E}'_i$ can be formulated as an optimization problem with an LMI constraint [15].

III. Forward Stochastic Reachability Analysis Using Uncertainty Propagation

Once $\mathcal{G}^*_B$ is computed, one can compute the one-step propagation transition kernel, $P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k)$, from $\mathcal{G}^*_B$ using the GMR. Let the mean and covariance of $\mathcal{G}^*_B$ be

$$\hat{\boldsymbol{\mu}}_{B,i} = \begin{pmatrix} \hat{\boldsymbol{\mu}}^f_{B,i} \\ \hat{\boldsymbol{\mu}}^p_{B,i} \end{pmatrix}, \tag{18}$$

$$\hat{\Sigma}^*_{B,i} + s_{B,i} = P_{B,i} = \begin{pmatrix} P^f_{B,i} & P^{f,p}_{B,i} \\ P^{p,f}_{B,i} & P^p_{B,i} \end{pmatrix}, \tag{19}$$

where $\hat{\boldsymbol{\mu}}^f_{B,i}, \hat{\boldsymbol{\mu}}^p_{B,i} \in \mathbb{R}^n$ and $P^f_{B,i}, P^{f,p}_{B,i}, P^{p,f}_{B,i}, P^p_{B,i} \in \mathbb{R}^{n\times n}$ are the corresponding mean and covariance fraction of the $i-$th component of $\mathcal{G}^*_B$.

Then, using the GMR, the conditional pdf on the propagated state can be computed as follows [12]:

$$P(\boldsymbol{x}_{k+1}|\boldsymbol{x}_k) = \sum_{i=1}^M \tilde{\pi}_i(\boldsymbol{x}_k) N\left(\tilde{\boldsymbol{\mu}}_i(\boldsymbol{x}_k), \tilde{\Sigma}_i\right), \tag{20}$$

where

$$\tilde{\boldsymbol{\mu}}_i(\boldsymbol{x}_k) = \hat{\boldsymbol{\mu}}^f_{B,i} + P^{f,p}_{B,i} P^{p^{-1}}_{B,i} \left(\boldsymbol{x}_k - \hat{\boldsymbol{\mu}}^p_{B,i}\right), \tag{21}$$

$$\tilde{\Sigma}_i = P^f_{B,i} - P^{f,p}_{B,i} P^{p^{-1}}_{B,i} P^{p,f}_{B,i}, \tag{22}$$

$$\tilde{\pi}_i(\boldsymbol{x}_k) = \frac{\hat{\pi}_i N\left(\boldsymbol{x}_k | \hat{\boldsymbol{\mu}}^p_{B,i}, P^p_{B,i}\right)}{\sum_{j=1}^M \hat{\pi}_j N\left(\boldsymbol{x}_k | \hat{\boldsymbol{\mu}}^p_{B,j}, P^p_{B,j}\right)}. \tag{23}$$

Algorithm 1 Bootstrapped Gaussian Mixture Model-Based Data-Driven Stochastic Reachability Analysis

Input: trajectory data $\mathcal{T}$, initial state pdf $P(\boldsymbol{x}_0)$, desired future time step $T_d > 0$, bootstrap sample number N_B, PGM filter sample number N_P, and δ **Output**: $P(\boldsymbol{x}_{T_d})$

EM algorithm

$M, \mathcal{G}, \hat{\theta} \leftarrow$ computed by the EM algorithm and $\mathcal{T}$. M can be chosen based on a proper criterion (e.g., BIC)

Bootstrapping

$\hat{\theta}_{B,i} \leftarrow$ bootstrapping sample (6) $\forall\, i = 1,2,\ldots,N_B$

$[\underline{\Sigma_{B,j}}, \overline{\Sigma_{B,j}}] \leftarrow$ from (7)-(10) $\forall\, j = 1,2,\ldots,M$

$\overline{\mathcal{G}_B} \leftarrow$ computed using (12)

$\mathcal{G}^*_B \leftarrow$ computed by solving (16a)-(16e)

Reachability analysis

while $k < T_d$ **do**

$S_l \leftarrow$ sample from $P(\boldsymbol{x}_k)$ $\forall l = 1,2,\ldots,N_P$

$P(\cdot|S_l) \leftarrow$ computed by $\mathcal{G}^*_B$ (17) and GMR (20) $\forall l = 1,2,\ldots,N_P$

$S'_l \leftarrow$ propagate according to $P(\cdot|S_l)$ $\forall l = 1,2,\ldots,N_P$

$P(\boldsymbol{x}_{k+1}) \leftarrow$ clustered as a GMM from S'_l

$k \leftarrow k+1$

end while

One can compute $P(\boldsymbol{x}_{k+1})$ by solving the Chapman-Kolmogorov equation using (20)-(23). However, analytically computing $P(\boldsymbol{x}_{k+1})$ is challenging due to the state dependency of (23). To tackle this problem, we apply the time update algorithm of the particle Gaussian mixture (PGM) filter. The time update algorithm of PGM filter starts by generating $N_P \in \mathbb{N}$ number of samples, S_i, $i = 1,2,\ldots,N_P$, from $P(\boldsymbol{x}_k)$. Then, the propagated samples S'_i are computed by propagating S_i through the transition kernel $P(\cdot|S_i)$. $P(\boldsymbol{x}_{k+1})$ is then approximated by clustering S'_i as a GMM.

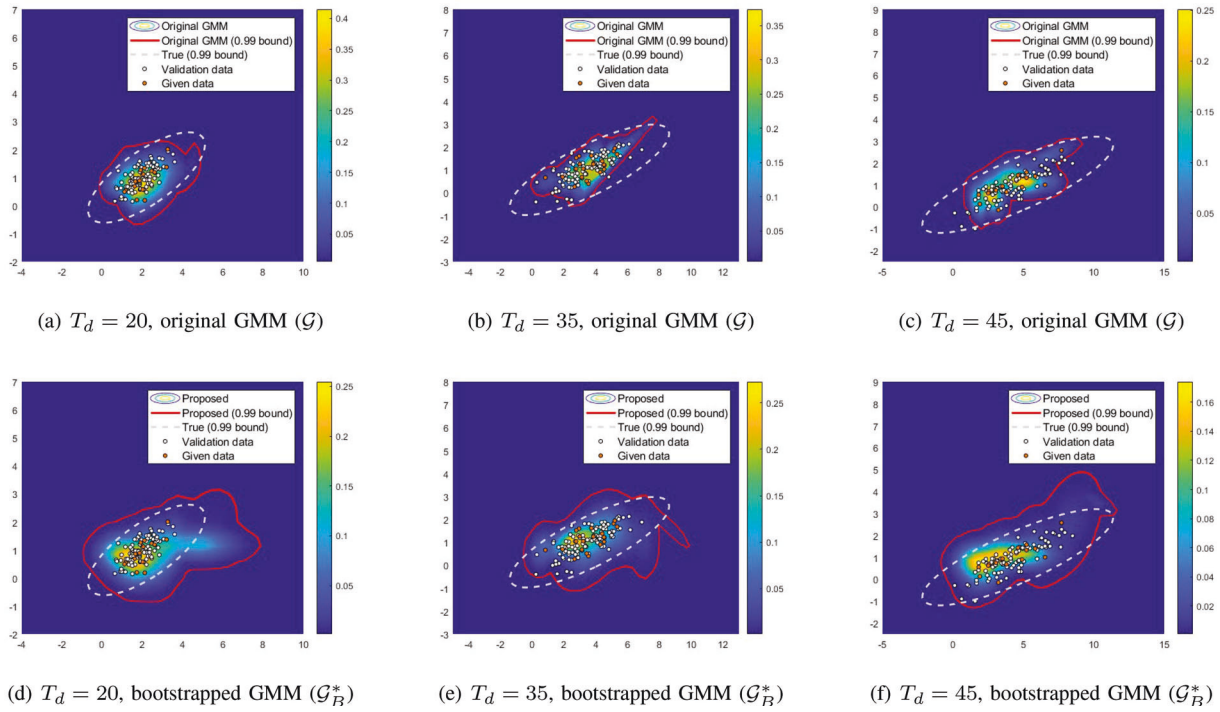
If $P(\boldsymbol{x}_{k+1})$ is distributed by a GMM with M number of Gaussian components (or approximated by the Gaussian sum approximation [16]), with the perfect clustering algorithm, there exists N_P that allows the time update algorithm of PGM filter to approximate the parameter of the $P(\boldsymbol{x}_{k+1})$ with desired confidence and accuracy [11]. Accordingly, one can predict how the system state pdf will evolve based on $\mathcal{G}^*_B$ with arbitrary confidence and accuracy on its parameters. Algorithm 1 shows the overall structure of the proposed algorithm.

IV. Numerical Simulations

A. Case Study 1: Noise-Driven Double Integrator

In this section, we provide the numerical simulation results of the proposed algorithm. We compare the results of Algorithm 1 based on $\mathcal{G}^*_B$ with that of $\mathcal{G}$ to show how the proposed method enlarges the volume of the reachable set. Moreover, we also compare the results with the true future state pdf at the desired future time step T to check the feasibility of the proposed method.

The target system is a noise-driven double integrator whose true dynamics is given as



(a) $T_d = 20$, original GMM ($\mathcal{G}$) (b) $T_d = 35$, original GMM ($\mathcal{G}$) (c) $T_d = 45$, original GMM ($\mathcal{G}$)

(d) $T_d = 20$, bootstrapped GMM ($\mathcal{G}_B^*$) (e) $T_d = 35$, bootstrapped GMM ($\mathcal{G}_B^*$) (f) $T_d = 45$, bootstrapped GMM ($\mathcal{G}_B^*$)

Fig. 1. Comparison between the results based on original GMM ($\mathcal{G}$) and bootstrapped GMM ($\mathcal{G}_B^*$). The informative reachable sets from the bootstrapped GMM include all the validation data whereas the original GMM fails.

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} N(0,1) \\ N(0,1) \end{bmatrix} \Delta t \tag{24}$$

where $\Delta t = 0.1s$ is the discretization time interval. The dynamics of the system is assumed to be unknown and the reachable set should be computed only using given 20 trajectories randomly generate from (24). We set the number of bootstrap samples (N_B) as 100, the number of Gaussian components (M) as 6, the number of samples for the PGM filter (N_P) as 500, and δ as 0.01. Also, we assume the initial state is distributed by $N(\boldsymbol{\mu}_0, \Sigma_0)$ where $\boldsymbol{\mu}_0 = [0, 1]^T$ and $\Sigma_0 = 0.0625 I_2$, where I_2 is the 2×2 identity matrix.

Fig. 1 shows the result of the simulations. The upper figures are the result using $\mathcal{G}$ and the lower figures are from $\mathcal{G}_B^*$. In the figures, the colored contour is the predicted state pdf at the desired future time step T_d, $P(\boldsymbol{x}_{T_d})$, the orange dots are the given trajectory data, and the white dots are the validation data obtained from (24) but never used to train the GMMs. The white dashed line represents the $1-\delta$ confidence bound of the true state pdf while the red line represents the $1-\delta$ confidence bound of the predicted state pdf. The bounds can be interpreted as a $\delta-$accurate reachable set, where the probability of the state to be within the bound is $1-\delta$ $(= 0.99)$ [3].

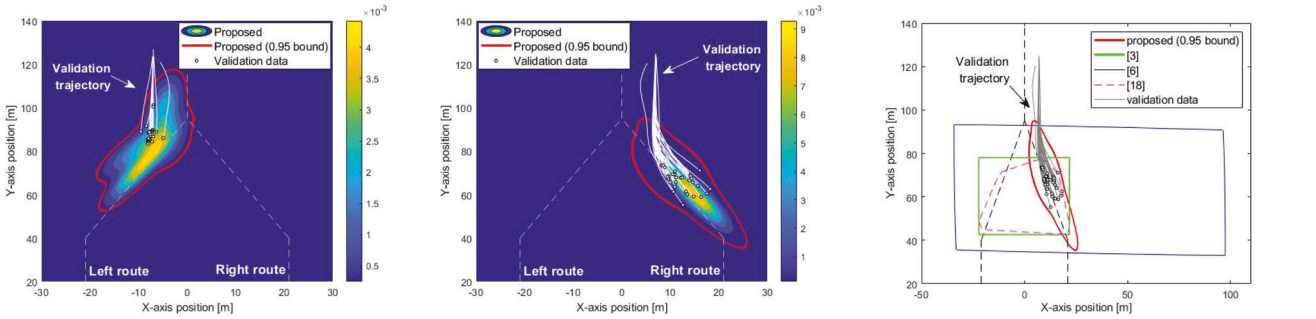
Throughout the simulation, one can easily notice that the proposed algorithm generates more informative results in comparison with the original GMM. This shows that the proposed scheme can increase the volume of each confidence ellipsoid and thus, yield a more informative model. As a result, the proposed algorithm succeeds in including the validation data whereas the $\mathcal{G}$ occasionally fails to enclose them. Compared to the true confidence area, the original GMM tends to underestimate the reachable set due to the small number of given trajectories. On the other hand, the proposed algorithm compensates for such a gap by explicitly accounting for the parameter estimation error, thereby generating a feasibly expanded reachable set.

B. Case Study 2: Human Subject Experiment

In this subsection, we demonstrate the proposed algorithm using the data obtained from human subject experiments. During the experiment, the participants were asked to safely land a multi-rotor in a virtual 2-D multi-rotor flight simulator. There are two predefined routes (left and right) and the participants were instructed to follow the routes assigned at the beginning of each simulation. We collected a total of 240 trajectories from 3 participants[1] and used 100 trajectories (50 for each route) to train $\mathcal{G}_B^*$. We also selected 60 trajectories (30 for each route) as the validation data. We set the number of bootstrap samples (N_B) as 50, the number of Gaussian components (M) as 6, the number of samples for the PGM filter (N_P) as 1000, and δ as 0.05. The nonlinear dynamics of the multi-rotor used in the flight simulator can be found in [17].

Fig. 2(a) and (b) illustrate the result of the proposed algorithm computed only using the initial state pdf of the validation data. In the figures, the contour is the predicted state pdf at the specific future time instance, the red line is the $1-\delta$ bound, the dashed gray line is the route that the participants should follow, and the white dots and lines are the true position and trajectory of the multi-rotor, respectively. As shown in the figures, the proposed algorithm successfully computes feasible reachable sets without knowing prior information about the dynamics. The computed reachable set includes all the validation trajectories showing that the proposed algorithm

[1]The Institutional Review Board (IRB) at Purdue University approved the study. IRB protocol number: IRB-2020-755.



(a) 4 seconds future prediction, left route (b) 6 seconds future prediction, right route (c) 6 seconds future prediction, comparative results

Fig. 2. Rechable sets and comparative simulation results using human subject experiment data.

can generate an informative reachable set based on $\mathcal{G}_B^*$ without additional data acquisition.

Meanwhile, the comparative simulation result between the proposed and existing algorithms is provided in Fig. 2(c), where each colored line describes the result of existing algorithms [3], [6], [18]. Note that [3] requires a certain amount of data for the probabilistic guarantee of the reachable set while [18] relies on asymptotic properties and thus, the given 100 trajectories might not be sufficient to fully leverage the benefit of the existing algorithms. One can easily observe that the existing algorithms show overly conservative results, including both of the routes. On the other hand, the proposed algorithm can reduce such conservativeness by specifying the vehicle's direction even with the scarce amount of data. This shows that the proposed method can perform well for an unknown (nonlinear) dynamical system controlled by a human operator, thereby demonstrating its capability to analyze a complex closed-loop (e.g., human-in-the-loop) dynamics in real-world applications. This can be beneficial in computing a reachable set for an unknown system, particularly if the additional trajectory data is too expensive to obtain.

V. Conclusion

In this letter, we presented a data-driven forward stochastic reachability analysis algorithm that can explicitly account for the parameter estimation error using the nonparametric bootstrapping technique. The proposed algorithm learns the evolution of the state probability density function (pdf) as a Gaussian mixture model (GMM) and compensates for the parameter estimation errors by using the confidence intervals computed from the bootstrapping. The stochastic reachable set is then computed by propagating the state pdf using the trained GMM and Gaussian mixture regression (GMR). The results of numerical simulations and human subject experiments show that the proposed algorithm can achieve informative prediction results even if a limited number of trajectory data is provided. In future work, we will extend our algorithm to the case where the data can be incrementally increased or regularly updated.

References

[1] A. Bajcsy, S. Bansal, E. Ratner, C. J. Tomlin, and A. D. Dragan, "A robust control framework for human motion prediction," *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 24–31, Jan. 2021.

[2] A. J. Thorpe and M. M. Oishi, "Model-free stochastic reachability using kernel distribution embeddings," *IEEE Control Syst. Lett.*, vol. 4, no. 2, pp. 512–517, Apr. 2020.

[3] A. Devonport and M. Arcak, "Data-driven reachable set computation using adaptive Gaussian process classification and Monte Carlo methods," in *Proc. Am. Control Conf. (ACC)*, 2020, pp. 2629–2634.

[4] P. Angkititrakul, R. Terashima, and T. Wakita, "On the use of stochastic driver behavior model in lane departure warning," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 1, pp. 174–183, Mar. 2011.

[5] O. Thapliyal and I. Hwang, "Approximate reachability for Koopman systems using mixed monotonicity," *IEEE Access*, vol. 10, pp. 84754–84760, 2022.

[6] A. Alanwar, A. Koch, F. Allgöwer, and K. H. Johansson, "Data-driven reachability analysis from noisy data," *IEEE Trans. Autom. Control*, vol. 68, no. 5, pp. 3054–3069, May 2023.

[7] T. Osa et al., "An algorithmic perspective on imitation learning," *Found. Trends Robot.*, vol. 7, nos. 1-2, pp. 1–179, 2018.

[8] A. O'Hagan, T. B. Murphy, L. Scrucca, and I. C. Gormley, "Investigation of parameter uncertainty in clustering using a Gaussian mixture model via jackknife, bootstrap and weighted likelihood bootstrap," *Comput. Statist.*, vol. 34, pp. 1779–1813, May 2019.

[9] A. C. Davison and D. V. Hinkley, *Bootstrap Methods and Their Application*. Cambridge, U.K.: Cambridge univ., 1997.

[10] G. Terejanu, P. Singla, T. Singh, and P. D. Scott, "Uncertainty propagation for nonlinear dynamic systems using Gaussian mixture models," *J. Guid. Control Dyn.*, vol. 31, no. 6, pp. 1623–1633, 2008.

[11] D. Raihan and S. Chakravorty, "Particle Gaussian mixture filters-i," *Automatica*, vol. 98, pp. 331–340, Dec. 2018.

[12] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," *Intell. Service Robot.*, vol. 9, no. 1, pp. 1–29, 2016.

[13] L. Asselborn and O. Stursberg, "Probabilistic control of uncertain linear systems using stochastic reachability," *IFAC-PapersOnLine*, vol. 48, no. 14, pp. 167–173, 2015.

[14] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. Philadelphia, PA, USA: Soc. Ind. Appl. Math., 1994.

[15] A. Halder, "On the parameterized computation of minimum volume outer ellipsoid of Minkowski sum of ellipsoids," in *Proc. IEEE Conf. Decision Control (CDC)*, 2018, pp. 4040–4045.

[16] H. W. Sorenson and D. L. Alspach, "Recursive Bayesian estimation using Gaussian sums," *Automatica*, vol. 7, no. 4, pp. 465–479, 1971.

[17] S. Byeon, W. Jin, D. Sun, and I. Hwang, "Human-automation interaction for assisting novices to emulate experts by inferring task objective functions," in *Proc. IEEE/AIAA Digital Avionics Syst. Conf. (DASC)*, 2021, pp. 1–6.

[18] T. Lew and M. Pavone, "Sampling-based reachability analysis: A random set theory approach with adversarial sampling," in *Proc. Conf. Robot Learn.*, 2021, pp. 2055–2070.