Learning Hidden Markov Models for Linear Gaussian Systems with Applications to Event-based State Estimation

Kaikai Zheng, Dawei Shi, and Ling Shi

Abstract—This work attempts to approximate a linear Gaussian system with a finite-state hidden Markov model (HMM), which is found useful in solving sophisticated event-based state estimation problems. An indirect modeling approach is developed, wherein a state space model (SSM) is firstly identified for a Gaussian system and the SSM is then used as an emulator for learning an HMM. In the proposed method, the training data for the HMM are obtained from the data generated by the SSM through building a quantization mapping. Parameter learning algorithms are designed to learn the parameters of the HMM, through exploiting the periodical structural characteristics of the HMM. The convergence and asymptotic properties of the proposed algorithms are analyzed. The HMM learned using the proposed algorithms is applied to event-triggered state estimation, and numerical results on model learning and state estimation demonstrate the validity of the proposed algorithms.

Index Terms—Event-based state estimation, linear Gaussian system, hidden Markov models, parameter learning.

I. INTRODUCTION

THE widespread utilization of wireless communication in civil and industrial applications promotes the development of networked control systems [1]. The requirement on system performance under limited communication bandwidth and computation leads to the development of event-based control and estimation strategies [2], [3], with which network resources can be saved significantly while maintaining system performance.

An important aspect of event-based networked control system design is state estimation with event-triggered sensor measurements [4], [5], [6], [7], [8], [9]. A number of interesting attempts were made in the literature, mostly for linear Gaussian systems. For reliable channels, systematic approaches were proposed to schedule the networked sensors or to design the event-based state estimators. For instance, in [10], a minimum mean square error estimator was derived for an event-triggering scheme to quantify the magnitude of the innovation of the estimator, which was achieved based on Gaussian assumptions of the distributions of the states conditioned on the available hybrid measurement information.

More general event-triggering conditions and multiple sensor measurements were discussed in [11]. However, for unreliable channels, the authors in [12], [13] proved that one cannot find an event-based trigger such that under possible packet drops, the state variable could remain a Gaussian variable. Thus, the Gaussian filtering framework might not be appropriate for event-triggered state estimation problems with unreliable communication channels.

To deal with such problems, event-triggered state estimators based on hidden Markov model (HMM) received much attention during the past few years [14], [15], [16], [17]. For instance, a Markov chain approximation algorithm for event-based state estimation was presented in [14]. In [15], [16], the reference measure approach was shown effective in obtaining recursive event-based state estimates for HMMs with different lossy channels. However, modeling a dynamic system as an HMM is a necessary task to implement the estimators aforementioned, which motivates our research in this work.

HMM is a data-driven generative model whose mathematical foundations were set off in the 1960s [18] and it was widely applied in many fields (e.g., natural language processing [19], [20], image processing [21], [22], [23], pattern recognition [24], [25], [26] and so on). To model a system as an HMM, different approaches have been attempted. First, several methods were developed to choose states that described system dynamics. The authors in [27] designed a probabilistic classification method for HMM states which made the HMM more robust against the possible mismatch or variation between training and test data. Algorithms to quantize continuous vectors to finite discrete states were discussed in [28], [29]. Second, sampling strategies were also investigated. Monte Carlo Markov Chain is a typical sampling method including Gibbs sampling, Metropolis-Hastings sampling and so on [30].

A sampling algorithm that resampled the full state sequence by employing a truncated approximation of Dirichlet process was developed to improve the mixing rate [31]. Algorithms were proposed for the problem of generating a random sample in accordance with the steady-state probability law of Markov chains [32]. Finally, after choosing states and generating sample data, algorithms for estimating the parameters of an HMM were also developed. On one hand, algorithms based on expectation maximization (EM) were designed to calculate the maximum likelihood estimation of the parameters of an HMM on-line [33], [34] or off-line [35], [36]. On the other hand, some works [37], [38] developed alternative estimation procedures named corrective training and maximum mutual
information estimation, which were designed to minimize the recognition errors instead of minimizing the estimation errors of the parameters. Besides, realization problems were discussed in [39], [40] to establish an HMM given observation sequence only.

Compared with the existing methods aforementioned, modeling a dynamic system with continuous states as an HMM involves additional challenges. The choice of states will affect the complexity of the HMM. Although the algorithms for choosing the states of an HMM in [23] and [29] can be applied to linear Gaussian systems, these methods are complicated whose complexity increases in the order of the systems and the dimension of the states. In this work, a simpler mapping is constructed between quantized values of $n$-dimensional vectors and the states of an HMM, based on which we can describe higher-order dynamic systems with the HMM states. Although the HMM has been extensively applied to one-dimensional problems, the extension to higher dimensions is intractable in many cases because of the data size and computational complexity [41]. To overcome this difficulty, the structure and the pattern of transition matrices of linear Gaussian systems are investigated in this work, which are used to simplify our model training. Furthermore, in the aforementioned works [42], [47], [43], [44], [45], [46], an HMM was trained by data sampled from real systems directly but it is difficult to sample dynamic systems with the same methods. Based on this observation, we develop an indirect approach through building an emulator leveraging existing results in system identification. The main contributions of our work are summarized as follows:

1) An indirect approach of training an HMM is proposed, in which we model a dynamic system as a state space model (SSM) first and then learn an HMM by using data generated by the SSM. Specifically, a mapping is constructed between quantized $n$-dimensional state space of the SSM and the discrete states of an HMM. Based on the proposed mapping, the SSM is used as an emulator that generates the state and measurement information needed to learn the HMM.

2) Patterns and characteristics of an HMM of a Gaussian system are investigated and used to simplify our HMM parameter learning algorithms. The core idea comes from the similarity in the probability density functions of different blocks in the transition matrix of an HMM. A series of algorithms are designed to reduce the time complexity of learning an HMM. Upper bounds of parameter estimation errors of transition matrices are proposed and we show that the error bounds diminish to zero with the increase of the quantization precision.

3) The obtained HMM learning approach is applied to event-based state estimation with an unreliable communication channel. Through a numerical example with comparative results from a Kalman filter with intermittent observations [47], we show that the HMM-based estimator performs nearly as good as the Kalman filter when the communication rate is high, but performs better than the Kalman filter with the reduction of the communication rate, which demonstrates the validity and strength of the proposed learning approach.

The remainder of the paper is organized as follows. Section II presents the system description and problem formulation. An indirect method for modeling a Gaussian system as an HMM is provided in Section III. Implementation issues and numerical verification of the results are presented in Section IV, followed by some conclusions in Section V.

**Notation:** For a matrix $A = (a_{ij}) \in \mathbb{R}^{p \times q}$, let $|A| = [a_{11}, \ldots, a_{pq}]$ and $|A|^{j} = [a_{1j}, \ldots, a_{pj}]^{T}$. We use $x_{i}$ to denote the $i$th element of vector $x$; thus for $x \in \mathbb{R}^{n}$, we write $x = [x_{1}, x_{2}, \ldots, x_{n}]^{T}$. Define $\text{sum}(x) := \sum_{i=1}^{n} x_{i}$. Write the Euclidean norm of real vectors as $\| \cdot \|_{2}$ and the absolute value of a scalar as $| \cdot |$. The notation $\lfloor x \rfloor$ denotes the flooring operator for a scalar $x$. The notation $\delta_{N}$ is used to denote the $i$th column of $I_{N}$, where $I_{N} \in \mathbb{R}^{N \times N}$ is an identity matrix. A diagonal matrix with elements of the main diagonal equal to $\{q_{1}, \ldots, q_{n}\}$ is written as $Q = \text{diag}(q_{1}, \ldots, q_{n})$. Kronecker product and Khatri-Rao product are denoted by $\otimes$ and $\ast$, respectively. Let $A = (a_{ij}) \in \mathbb{R}^{n \times n}$, $B = (b_{ij}) \in \mathbb{R}^{p \times q}$ and $C = (c_{ij}) \in \mathbb{R}^{m \times n}$, the Kronecker product and the Khatri-Rao product are defined as

$$
A \otimes B := \begin{bmatrix}
    a_{11}B & a_{12}B & \cdots & a_{1n}B \\
    a_{21}B & a_{22}B & \cdots & a_{2n}B \\
    \vdots & \vdots & \ddots & \vdots \\
    a_{m1}B & a_{m2}B & \cdots & a_{mn}B
\end{bmatrix},
$$

$$
A \ast C := [(A)_{1} \otimes [C]_{1}, \ldots, (A)_{n} \otimes [C]_{n}]^{T}.
$$

In the derivations, the notions of ‘standard column’ and ‘target column’ are used. Variables for ‘standard column’ are marked by ‘$S$’ and variables for ‘target column’ are marked by ‘$T$’. For example, $\mu_{S}^{r}, x_{p}(k; S)$ are variables related to ‘standard column’, while $\mu_{T}^{r}, x_{p}(k; T)$ are variables related to ‘target column’.

## II. Problem Setup

Consider an $n$th-order linear Gaussian system with the following state space model

$$
\begin{align*}
x(k+1) &= Ax(k) + w(k), \\
y(k) &= Cx(k) + v(k),
\end{align*}
$$

where $x(k) = [x_{1}(k), \ldots, x_{n}(k)]^{T} \in \mathbb{R}^{n}$ is the state, $y(k) = [y_{1}(k), \ldots, y_{m}(k)]^{T} \in \mathbb{R}^{m}$ is the output, $w(k) \in \mathbb{R}^{n}$ and $v(k) \in \mathbb{R}^{m}$ are i.i.d. zero mean Gaussian noises with covariance $Q = \text{diag}(q_{1}, \ldots, q_{n})$ and $R = \text{diag}(r_{1}, \ldots, r_{n})$. For learning purpose, we assume the system is stable. Note that although we require the covariance matrices $Q$ and $R$ to be diagonal, the case of non-diagonal $Q$ and $R$ can be considered through performing linear transformations of the state and measurement equations in $\textbf{1}$. In this work, we attempt to build an HMM representation for the system in $\textbf{1}$. To aid our description, the notation related to an HMM is first introduced.
The state space $S_X$ and the measurement space $S_Y$ of an HMM are defined as finite sets

$$S_X := \{\delta_N^1, \ldots, \delta_N^N\}, \quad S_Y := \{\delta_M^1, \ldots, \delta_M^M\},$$

where the positive constants $N$ and $M$ denote the cardinalities of $S_X$ and $S_Y$, respectively. Let $X$ be a Markov chain with state $X(k) \in S_X$ and the Markov property implies that

$$P \left( X(k+1) = \delta_N^i | \{X(0), \ldots, X(k)\} \right)$$

$$= P \left( X(k+1) = \delta_N^i | X(k) \right).$$

Write

$$a_{i,j} := P \left( X(k+1) = \delta_N^i | X(k) = \delta_N^j \right),$$

$$A := (a_{i,j}) \in \mathbb{R}^{N \times N},$$

and we obtain

$$E(X(k+1)|X(k)) = AX(k). \quad (3)$$

The measurement process satisfies

$$P \left( Y(k) = \delta_M^j | \{X(0), \ldots, X(k)\} \right)$$

$$= P \left( Y(k) = \delta_M^j | X(k) \right).$$

Similarly, after defining

$$c_{i,j} := P \left( Y(k) = \delta_M^j | X(k) = \delta_N^i \right),$$

$$C := (c_{i,j}) \in \mathbb{R}^{M \times N},$$

we obtain

$$E(Y(k)|X(k)) = CX(k). \quad (4)$$

Note that in equation (1), the effect of deterministic input is omitted, which can be evaluated with the knowledge of the SSM. Specifically, write

$$x(k+1) = Ax(k) + Bu(k) + w(k),$$

$$y(k) = Cx(k) + Du(k) + v(k), \quad (5)$$

where $u(k)$ denotes the deterministic input signal. For systems (1) and (5), if the input signal $u(k)$ and the system parameters $A, B, C, D$ are known and the initial states of them are the same, the state and measurement $x(k), y(k)$ of system (1) can be calculated based on $x(k), y(k)$ according to

$$x(k) = x(k) - \Pi(k)U(k) \quad (6)$$

$$y(k) = y(k) - C\Pi(k)U(k) - Du(k), \quad (7)$$

where

$$\Pi(k) = [(A)^{k-1}B (A)^{k-2}B \ldots AB B]$$

$$U(k) = [u(0) u(1) \ldots u(k-1)]^T.$$
However, the state of an SSM is an \( n \)-dimensional vector in \( \mathbb{R}^n \) for an \( n \)th-order system while a state in an HMM is an \( N \)-dimensional unit vector in a finite set \( S_X \). Therefore an exact mapping is impossible. The idea is to build a quantization mapping from the state and output spaces \( \mathbb{R}^n \) and \( \mathbb{R}^m \) of the system in \([1]\) to the state spaces \( S_X \) and \( S_Y \) of the HMM in \([3]\) and \([4]\).

When \( A \) is stable, the probability distributions of \( x_p \) converges to Gaussian distributions at the steady state
\[
x_p \sim N(0, (\sigma_p^x)^2) \quad p \in \{1, \ldots, n\}.
\]
Now we introduce a naive quantization method to partition \( \mathbb{R} \) into \( N_p \) subintervals for \( x_p \). We observe that \( x_p \) takes values in a finite interval \([51]\)
\[
x_p \in [\underline{x}_p, \overline{x}_p] = [-\rho \sigma_p^x, +\rho \sigma_p^x],
\]
with a large probability depending on \( \rho \). This finite interval \([-\rho \sigma_p^x, +\rho \sigma_p^x]\) is then divided into \( N_p-2 \) equal sub-intervals\([4]\) and \( h_p^k \) is used to denote the length of sub-intervals. After adding \((-\infty, -\rho \sigma_p^x)\) and \([+\rho \sigma_p^x, +\infty)\) as the first and last sub-interval, respectively, we obtain \( N_p \) sub-intervals. For the event that \( x_p(k) \) belongs to the \( k \)th sub-interval, we define
\[
X_p(k) = \delta_{N_p}^x(k) \Leftrightarrow x_p(k) \in [\underline{x}_p(k), \overline{x}_p(k)],
\]
where \( X_p(k) \) denotes the quantized \( x_p(k) \), and \( \underline{x}_p(k), \overline{x}_p(k) \) represent the lower and upper bounds of the sub-interval, respectively.

Similarly, \( y_p \) converges to Gaussian distribution as
\[
y_p \sim N(0, (\sigma_p^y)^2) \quad p \in \{1, \ldots, m\}.
\]
Based on the observation that
\[
y_p \in [\underline{y}_p, \overline{y}_p] = [-\rho \sigma_p^y, +\rho \sigma_p^y],
\]
the quantization of \( y_p \) is defined as
\[
Y_p(k) = \delta_{M_p}^y(k) \Leftrightarrow y_p(k) \in [\underline{y}_p(k), \overline{y}_p(k)],
\]
where \( Y_p(k) \) denotes the quantized \( y_p(k) \), and \( \underline{y}_p(k), \overline{y}_p(k) \) represent the lower and upper bound of the sub-interval, respectively. The notation \( h_p^y \) is used to denote the length of sub-intervals.

Then, through combining \( \{X_1(k), \ldots, X_n(k)\} \), \( X(k) \) can be calculated with the Kronecker product of them, which is obtained by
\[
X(k) = X_1(k) \otimes X_2(k) \otimes \cdots \otimes X_n(k) = \delta_N^x(k).
\]
Similarly, we obtain
\[
Y(k) = Y_1(k) \otimes Y_2(k) \otimes \cdots \otimes Y_m(k) = \delta_M^y(k).
\]

The mapping between \( \mathbb{R}^n, \mathbb{R}^m \) and \( S_X, S_Y \) is proposed so far and Algorithm 2 shows the detailed steps.

In Algorithms 2, \( m_p^0, \ldots, m_p^r \) are calculated as boundaries of sub-intervals (Lines 3-9), based on which \( x_p(k) \) is quantized to \( X_p(k) \) (Lines 10-14). Then, repeated work is performed to quantify \( y_p(k) \) to \( Y_p(k) \) (Lines 16-29). Finally, \( X(k) \) and \( Y(k) \) are obtained using equation \([10]\) and \([11]\) (Lines 30-31). Note that the boundaries are constant for a given system, thus the boundaries can be stored to avoid repeated calculation for different data \( x(k), y(k) \).

This mapping approximately but uniquely associates a state of \( x(k), y(k) \) to a state \( X(k), Y(k) \), which is detailed in Algorithm 2. In this way, the training data \( \{Y(k), X(k)\} \) needed to learn \( A \) and \( C \) are obtained based on \( \{y(k), x(k)\} \).

### B. Training of an HMM: A Naive Method

Based on the law of large numbers, the average of the results obtained from a large number of trials should be close to the expected value \([51]\). Combined with a state sequence \( \{X(0), X(1), \ldots, X(S)\} \) and an observation sequence \( \{Y(0), Y(1), \ldots, Y(S)\} \), the parameters \( A \) and \( C \) of an HMM can be evaluated as
\[
a_{i,j}^+ = \frac{X_{i,j}}{N} \quad c_{i,j}^+ = \frac{Y_{i,j}}{M},
\]
where

#### Algorithm 2  Mapping algorithm of state translation from \( x_k, y_k \) to \( X_k, Y_k \)

1: Input \( x(k), y(k) \);
2: for \( p \in \{1, 2, \ldots, n\} \) do
3: \( m_p^0 = -\infty \);
4: \( m_p^r = \pi_p \);
5: \( m_p^{N_p} = \bar{y}_p \);
6: \( m_p^r = \infty \);
7: for \( j \in \{2, 3, \ldots, N_p - 2\} \) do
8: \( m_p^j = m_p^1 + (j - 1) \frac{\bar{y}_p - \underline{y}_p}{M_p - 2} \);
9: end for
10: for \( \pi \in \{1, 2, \ldots, N_p\} \) do
11: if \( m_p^{\pi - 1} \leq x_p(k) \leq m_p^\pi \) then
12: \( X_p(k) = \delta_N^x \);
13: end if
14: end for
15: end for
16: for \( p \in \{1, 2, \ldots, m\} \) do
17: \( m_p^0 = -\infty \);
18: \( m_p^r = \bar{y}_p \);
19: \( m_p^{M_p} = \infty \);
20: \( m_p^r = \infty \);
21: for \( j \in \{2, 3, \ldots, M_p - 2\} \) do
22: \( m_p^j = m_p^1 + (j - 1) \frac{\bar{y}_p - \underline{y}_p}{M_p - 2} \);
23: end for
24: for \( \pi \in \{1, 2, \ldots, M_p\} \) do
25: if \( m_p^{\pi - 1} \leq y_p(k) \leq m_p^\pi \) then
26: \( Y_p(k) = \delta_M^y \);
27: end if
28: end for
29: end for
30: \( X(k) = X_1(k) \otimes X_2(k) \otimes \cdots \otimes X_n(k) \);
31: \( Y(k) = Y_1(k) \otimes Y_2(k) \otimes \cdots \otimes Y_m(k) \);
32: Output \( X(k), Y(k) \).
where $X_{i,j}$ represents the count of the event that $X_k = \delta_i, Y_k = \delta_j$, $Y_{i,j}$ denotes the count of the event that $X_k = \delta_i, Y_k = \delta_j$. The notations $a_{i,j}$ and $c_{i,j}$ are used to denote the transition probability estimated according to the law of large numbers. Then we define

$$
A^+ := (\hat{a}_{i,j}) \in \mathbb{R}^{N \times N}, \quad C^+ := (\hat{c}_{i,j}) \in \mathbb{R}^{M \times N}.
$$  \hspace{1cm} (13)

The method of the evaluation is shown in Algorithm 3.

**Algorithm 3** Training of a transition matrix $A^+$ and a measurement matrix $C^+$

1. Initialize $A^+, C^+$ and $A^x, C^x$;
2. Initialize the number of loop $L$ and the data size $D_x$;
3. for $i = 1$ to $L$
4. \hspace{0.5cm} Initialize $A^x, C^x$;
5. \hspace{0.5cm} Simulate the system (1) to generate $D_x$ new data;
6. \hspace{1cm} (Generally, the data is different in each loop because of the disturbances.)
7. for $k = 1$ to $D_x$
8. \hspace{1cm} Translate $x(k), y(k)$ to $x(k), y(k)$ using Algorithm 1;
9. \hspace{1cm} Quantize $x(k), y(k)$ to calculate $\pi_x(k)$ and $\pi_y(k)$ according to Algorithm 2;
10. \hspace{1cm} $c_{\pi_x(k), \pi_y(k)} = c_{\pi_x(k), \pi_y(k)} + 1$;
11. for $k = 1$ to $D_x - 1$
12. \hspace{1cm} $a_{\pi_x(k+1), \pi_x(k)} = a_{\pi_x(k+1), \pi_x(k)} + 1$;
13. end for
14. $A^+ = A^+ + A^x$;
15. $C^+ = C^+ + C^x$;
16. end for
17. for $j = 1$ to $N$
18. \hspace{0.5cm} if sum($[A^+]^j$) > 0 then
19. \hspace{1cm} $[A^+]^j = [A^+]^j \cdot (\text{sum}([A^+]^j))^{-1}$;
20. \hspace{0.5cm} end if
21. \hspace{0.5cm} if sum($[C^+]^j$) > 0 then
22. \hspace{1cm} $[C^+]^j = [C^+]^j \cdot (\text{sum}([C^+]^j))^{-1}$;
23. \hspace{0.5cm} end if
24. end for

Specifically, Algorithm 2 is mainly composed of the following steps. Training parameters and transition matrices are initialized (Lines 1-2). In each loop, training data are generalized (Line 5) and processed (Lines 7-8). The quantization results are used to count state transition events and stored as $A^x, C^x$ as sparse matrices (Line 9 and Lines 11-13). The matrices $A^x, C^x$ in every loop are summed up as $A^+, C^+$, respectively (Lines 14-15). Normalization of transition matrices $A^+, C^+$ is performed at the end of Algorithm 3 (Lines 17-24). The number of the parameters learned through “exhaustive training” in Algorithm 3 equals to $\sum_{p=1}^{n} (N_p)^2 + \sum_{p=1}^{n} N_p \sum_{p=1}^{m} M_p$. By “exhaustive training”, we mean that the parameter is trained/learned based on exhaustive Monte Carlo simulation, the precision of which depends on the size of the training data, $L \times D_x$.

**Remark 1:** Several tricks are taken in Algorithm 3 to reduce the time cost and errors. In view of the time cost of retrieving dense matrices, a series of sparse matrices are trained and the weighted average of them can be estimations of $A$ and $C$. The parameters are normalized at the end of Algorithm 3 to reduce round-off errors.

### C. Training of an HMM: An Improved Method

Equation (12) and Algorithm 3 provide a method to evaluate the parameters of an HMM. The evaluation will become more accurate with the increase of the data size $L \times D_x$. Unfortunately, the time complexity of the algorithm will become unacceptable at the same time, especially when the order of the system in 5 is high or the cardinalities $N, M$ are large. This prompts the design of learning algorithm with reduced complexity, which is discussed in this subsection.

Fortunately, we find a pattern which can be used to solve the problem. A column of the transition matrix is shown in Fig. 2 and Fig. 3, where the orders of the systems considered are 2 and 3, respectively. From these two figures, we observe that a similar pattern is shared in the curves that correspond to different blocks of the matrices.

![Fig. 2: One column of $A^+$ for a 2nd-order system.](image1)

![Fig. 3: One column of $A^+$ for a 3rd-order system.](image2)

The plots of a column of $A^+$ for different systems in Fig. 2 and Fig. 3 show that a pattern featuring a unimodal curve appears repeatedly. The periodical structural pattern arises from the mapping we constructed in the last subsection. The results developed below will gradually reveal the cause of the pattern.
Theorem 1: For an nth-order Gaussian system (1) and an HMM determined as Algorithm 2, the transition matrix $A$ satisfies
\[ A = A^1 * A^2 * \cdots * A^n, \]
where
\[ a_{i,j}^p = P \left( X_p(k+1) = \delta_{N_p}^i | X(k) = \delta_{N}^j \right), \]
\[ A^p = (a_{i,j}^p) \in \mathbb{R}^{N_p \times N}. \]

Proof: Considering that the covariance matrix $Q$ is a diagonal matrix, the independence between different elements leads to
\[ P \left( X(k+1) = \delta_{N_p}^{\pi^p(k+1)} | X(k) = \delta_{N}^j \right) = \prod_{p=1}^n P \left( X_p(k+1) = \delta_{N_p}^{\pi_p^p(k+1)} | X(k) = \delta_{N}^j \right), \]
thus we obtain
\[ a_{\pi^p(k+1),j} = \prod_{p=1}^n a_{\pi_p^p(k+1),j}. \]

Noticing that
\[ X_p(k+1) = \delta_{N_p}^{\pi_p^p(k+1)}, \quad X(k) = \delta_{N}^j, \]
the combination method (10) leads to
\[ \delta_{N}^j = \delta_{N_1}^{\pi^p(k+1)} \otimes \cdots \otimes \delta_{N_n}^{\pi_n^p(k+1)}. \]
Thus we obtain
\[ [A]^j = [A^1]^j \otimes \cdots \otimes [A^n]^j. \]
Then we have
\[ A = A^1 * A^2 * \cdots * A^n \]
based on the definition of Khatri-Rao product, which completes the proof. 

Remark 2: Note that the transition matrix $A^p, p \in \{1, \ldots, n\}$ denotes the transition process from $X(k)$ to $X_p(k+1)$. The results of Theorem 1 reveal the relationship between $A^p, p \in \{1, \ldots, n\}$ and $A$, which indicates that the focus of parameter estimation can be turned to $A^p$ instead of $A$.

Similarly, Theorem 2 is proposed to explore the pattern for measurement matrix $C$, the proof of which is stated in Appendix A.

Theorem 2: For an nth-order Gaussian system (1) and an HMM determined as Algorithm 2, the measurement matrix $C$ satisfies
\[ C = C^1 * C^2 * \cdots * C^m, \]
where
\[ c_{i,j}^p = P \left( Y_p(k) = \delta_{M_p}^i | X(k) = \delta_{N}^j \right), \]
\[ C^p = (c_{i,j}^p) \in \mathbb{R}^{M_p \times N}. \]

Theorems 1 and 2 allows us to focus our analysis on $A^p, C^p$ rather than $A, C$. A simple example of two columns of $A^1$ is shown in Fig. 4, which shows an obvious similarity between different columns.

The similarity between different columns of $A^p$ drives us to explore the relationship between different columns of transition matrix. Thus the elements of $A^p$ and $C^p$ are investigated as follows.

For a stochastic variable that corresponds to the Gaussian distribution $x \sim N(\theta, \sigma^2)$, write the probability density function as
\[ f_G(x, \sigma) := \frac{1}{\sqrt{2\pi} \sigma} \exp \left( -\frac{x^2}{2\sigma^2} \right). \]
Since $x_p(k), p \in \{1, \ldots, n\}$ correspond to zero-mean Gaussian distributions, the linear combinations of them also correspond to zero-mean Gaussian distributions. After writing
\[ \alpha_p^x(k) = [A]_{p,x}(k), \]
the independence between $\alpha_p^x(k)$ and $w_p(k)$ leads to a two-dimensional Gaussian distribution
\[ f_p^x(\alpha_p^x(k), w_p(k)) := f_G(\alpha_p^x(k), \sigma_p^x) f_G(w_p(k), q_p), \]
where $\sigma_p^x$ is the variance of $\alpha_p^x(k)$. Considering that $w_p(k) = x_p(k+1) - \alpha_p^x(k)$, we obtain
\[ f_p^x(\alpha_p^x(k), w_p(k)) = f_G(\alpha_p^x(k), x_p(k+1) - \alpha_p^x(k)) = f_G(\alpha_p^x(k), \sigma_p^x) f_G(x_p(k+1) - \alpha_p^x(k), q_p), \]
which is a two-dimensional Gaussian distribution of $\alpha_p^x(k)$ and $x_p(k+1)$. This substitution step is necessary because the quantization method is designed based on $x(k), x(k+1)$ rather than $w(k)$.

For $X(k) = \delta_{N}^j$, and $X_p(k+1) = \delta_{N_p}^i$, we define
\[ X(k) = \delta_{N}^j \iff \alpha_p^x(k) \in \left[ \alpha_p^x(k), \pi_p^x(k) \right]; \]
\[ X_p(k+1) = \delta_{N_p}^i \iff x_p(k+1) \in \left[ x_p(k+1), \pi_p(k+1) \right], \]
where $x_p(k+1), \pi_p(k+1)$ and $\alpha_p^x(k), \pi_p^x(k)$ denote the boundaries of $x_p(k+1)$ and $\alpha_p^x(k)$, respectively.

Then, the transition probability $a_{i,j}^p$ can be calculated by a.
The integral of equation (20) as

\[ \Omega_p^x := \{ \alpha_p^x(k), x_p(k+1) \mid \alpha_p^x(k) \leq \alpha_p^x(k) \leq \alpha_p^x(k), \]

\[ x_p(k+1) \leq x_p(k+1) \leq \Omega_p^x(k+1) \}. \]  

(22)

The results above show that, the relationship between different elements is concealed by complicated integrals. In order to avoid those complex integrals, a method of numerical integration is adopted to analyze the transition probabilities.

Consider a continuous differentiable function \( f(x, y) \) on a rectangular region \( \Omega : \{ x \leq x \leq \pi, y \leq y \leq \pi \} \). In order to approximate those complicated integrals, a method of numerical integration is adopted to analyze the transition probabilities.

\[ \Phi(\cdot, \cdot, x, \pi, y, \pi) := (\pi - y)(\pi - x) f \left( \frac{x + \pi}{2}, \frac{y + \pi}{2} \right), \]  

and the approximation error is provided in Lemma 1.

**Lemma 1:** The double integral of a continuous differentiable function, \( \int f(x, y) dx dy \), can be approximated as equation (23) with the estimation error satisfying

\[ \left| \int f(x, y) dx dy - \Psi(\cdot, \cdot, x, \pi, y, \pi) \right| \leq \frac{(\pi - x)^3(\pi - y)}{24} \max_{\Omega} \frac{\partial^2 f(x, y)}{\partial x^2} \]

\[ + \frac{(\pi - x)(\pi - y)^3}{24} \max_{\Omega} \frac{\partial^2 f(x, y)}{\partial y^2}. \]  

(24)

**Proof:** The Taylor expansion of \( f(x, y) \) about \( x \) at the midpoint \( \left( \frac{\pi + x}{2}, y \right) \) with a Lagrange reminder term is

\[ f(x, y) = f \left( \frac{\pi + x}{2}, y \right) + \frac{\partial f}{\partial x} \left( \frac{\pi + x}{2}, y \right) \left( x - \frac{\pi + x}{2} \right) \]

\[ + \frac{1}{2} \frac{\partial^2 f}{\partial x^2} \left( \frac{\pi + x}{2}, y \right) \left( x - \frac{\pi + x}{2} \right)^2, \]

where \( \xi_x \in (x, \pi) \). Then, the integral of \( f(x, y) \) from \( x \) to \( \pi \) leads to

\[ \int f(x, y) dx \]

\[ = \int f \left( \frac{x + \pi}{2}, \frac{y + \pi}{2} \right) dx + \int \frac{\partial f}{\partial x} \left( \frac{\pi + x}{2}, \frac{y + \pi}{2} \right) \left( x - \frac{\pi + x}{2} \right) dx \]

\[ + \frac{1}{2} \frac{\partial^2 f}{\partial x^2} \left( \frac{\pi + x}{2}, \frac{y + \pi}{2} \right) \left( x - \frac{\pi + x}{2} \right)^2 \]

\[ = (\pi - x)^3 \frac{\partial^2 f}{\partial x^2} \xi_x^3 \frac{(\pi - x)^3}{24} \]  

(25)

Similarly, the Taylor expansion of \( (\pi - x) f \left( \frac{\pi + x}{2}, y \right) \) about \( y \) at the point \( \left( \frac{\pi + x}{2}, \frac{\pi + y}{2} \right) \) with a Lagrange reminder term is

\[ (\pi - x) f \left( \frac{\pi + x}{2}, \frac{\pi + y}{2} \right) \]

\[ + (\pi - x) \frac{\partial f}{\partial y} \left( \frac{\pi + x}{2}, \frac{\pi + y}{2} \right) \left( y - \frac{\pi + y}{2} \right) \]

\[ + \frac{(\pi - x)^2 \frac{\partial^2 f}{\partial y^2} \xi_y^2}{24} \left( y - \frac{\pi + y}{2} \right)^2, \]

where \( \xi_y \in \{ y, \pi \} \). Then, the integral of \( (\pi - x) f \left( \frac{\pi + x}{2}, y \right) \) from \( y \) to \( \pi \) leads to

\[ \int_y \left( \frac{\pi - x}{2}, \frac{\pi + y}{2} \right) dy - \Psi(\cdot, \cdot, x, \pi, y, \pi) \]

\[ = (\pi - x)^3 \frac{\partial^2 f}{\partial y^2} \xi_y^3 \frac{\pi - x}{24} \]

(26)

The error in the approximation is

\[ \left| \int f(x, y) dx \right| \leq \frac{1}{24} (\pi + x)^3 (\pi + y)^3 (\pi + x + y) \max_{\Omega} \frac{\partial^2 f}{\partial x^2} (\pi + x, y) \left( x - \frac{\pi + x}{2} \right) \]

\[ + \frac{1}{24} (\pi + x)^3 (\pi + y)^3 (\pi + x + y) \max_{\Omega} \frac{\partial^2 f}{\partial y^2} (\pi + x, y) \left( y - \frac{\pi + y}{2} \right) \]

(27)
The integral of equation (24) from \( y \) to \( \overline{y} \) can be written as
\[
\int_{y}^{\overline{y}} f(x,y) \, dx \, dy = \int_{(x-\frac{x}{2})}^{x} f(x, \frac{x}{2} + y) \, dy,
\] (26)

\[
\int_{y}^{\overline{y}} \frac{(x-x)^3}{24} \partial^2 f(\xi_1, y) \, dy = \frac{(\overline{y} - y)(x-x)^3}{24} \partial^2 f(\xi_1, \xi_2),
\]

where \( \xi_1 \in (y, \overline{y}) \). Finally, after writing \( \xi_2 = \frac{\overline{y} + y}{2} \) and adding equation (26) to equation (25), we have
\[
\int_{\Omega} f(x,y) \, dx \, dy - \Psi(f(x,y), x, x, y, \overline{y}) = \frac{(\overline{y} - y)(x-x)^3}{24} \partial^2 f(\xi_1, \xi_2). \]

Therefore, the claim holds. 

Based on Lemma 1, the transition probability \( a_{i,j}^p \) can be evaluated using numerical integral method in Lemma 2.

**Lemma 2:** The transition probability \( a_{i,j}^p \) in equation (21) can be approximated by \( a_{i,j}^p \) as
\[
a_{i,j}^p = \Psi(f_p(x,y), \alpha_p^p(k), \pi_p^p(k), x_p, y_p, \pi_p) \] (27)
for \( p \in \{1, \ldots, n\} \), where \( i = \pi_p, j = \pi_p \), and \( \pi_p \). The estimation error satisfies equation (28).

**Proof:** The claims follow easily from Lemma 1 by differentiation. 

**Remark 3:** Lemma 2 proposes a simple estimation of the transition probabilities in equation (21). Considering that a larger choice of \( \rho \) makes the boundary conditions smaller, the boundary cases \( x_p \in (-\infty, \pi_p), x_p \in [\pi_p, +\infty) \) are excluded in Lemma 2 since their influence on our estimation result can be controlled to be sufficiently small.

Similarly, an estimate \( \tilde{c}_{i,j}^p \) of transition probability \( c_{i,j}^p \) can be formulated with a similar method. Write \( \alpha_p^p(k) = |c_{i,j}^p(x, \pi_p) \), then the independence between \( \alpha_p^p(k) \) and \( v_p(k) \) leads to a two-dimensional Gaussian distribution
\[
f_p^p(\alpha_p^p(k), v_p(k)) = f_C(\alpha_p^p(k), \sigma_p^p) f_C(v_p(k), r_p), \quad \text{(29)}
\]
where \( \sigma_p^p \) is the variance of \( \alpha_p^p(k) \). Considering that \( v_p(k) = y_p(k) - \alpha_p^p(k) \), we obtain
\[
\begin{align*}
\frac{\partial^2 f_p^p(\alpha_p^p(k), y_p(k) - \alpha_p^p(k))}{\partial(\alpha_p^p(k))^2} & = f_p^p(\alpha_p^p(k), v_p(k)) \quad \text{(30)} \\
& = f_p^p(\alpha_p^p(k), y_p(k) - \alpha_p^p(k)) \\
& = f_p^p(\alpha_p^p(k), \sigma_p^p) f_p^p(y_p(k) - \alpha_p^p(k), r_p),
\end{align*}
\]

which is a two-dimensional Gaussian distribution of \( \alpha_p^p(k) \) and \( y_p(k) \).

For \( X(k) = \delta_{M_p}^k \) and \( Y_p(k) = \delta_{M_p}^k \), we define
\[
X(k) = \delta_{M_p}^k \Leftrightarrow \alpha_p^p(k) \in [\pi_p^p(k), \pi_p^p(k)]; \\
Y_p(k) = \delta_{M_p}^k \Leftrightarrow y_p(k) \in [\pi_p^p(k), \pi_p^p(k)],
\]

where \( y_p(k), \pi_p(k) \) and \( \alpha_p^p(k), \pi_p^p(k) \) denote the boundaries of \( y_p(k) \) and \( \alpha_p^p(k) \), respectively. Then, the transition probability \( c_{i,j}^p \) can be calculated by an integral of equation (31)
\[
c_{i,j}^p = P(Y_p(k) = \delta_{M_p}^k | X(k) = \delta_{N_p}^k) = \int_{\Omega_p^p} f_p^p(\alpha_p^p(k), v_p(k)) \, d\alpha_p^p(k) \, dy_p(k),
\]

where \( \Omega_p^p \) is a rectangular region as
\[
\Omega_p^p : \{\pi_p^p(k) \leq \alpha_p^p(k) \leq \pi_p^p(k), y_p(k) \leq y_p(k) \leq \pi_p^p(k)\}.
\]

Based on Lemma 1, \( c_{i,j}^p \) can be evaluated using numerical integral method as follows.

**Lemma 3:** The transition probability \( c_{i,j}^p \), shown in equation (31), can be approximated by \( \tilde{c}_{i,j}^p \) as
\[
\tilde{c}_{i,j}^p = \Psi(f_p(x,y), \alpha_p^p(k), \pi_p^p(k), y_p(k), \pi_p(k)) \] (32)
for \( p \in \{1, \ldots, m\} \), where \( i = \pi_p^p, j = \pi_p^p \), and \( \pi_p^p \). The estimation error satisfies equation (33).

**Proof:** The claims follow easily from Lemma 1 by differentiation. 

**Remark 4:** Lemma 3 proposes a simple estimation of the transition probabilities in equation (31). Similar to the analysis in Remark 3, the boundary cases \( x_p \in \pi_p(k), x_p \in [\pi_p, +\infty) \), \( y_p \in (-\infty, y_p) \), \( y_p \in [\pi_p, +\infty) \) are excluded as their influence on estimation can be sufficiently small.

Lemmas 2 and 3 provide numerical integral representation of the transition probabilities, which can be used to investigate the relationship between different columns of \( A^p \) or \( C^p \). Note that variables for standard columns are marked by ‘S’ and variables for target columns are marked by ‘T’, where a
Theorem 3: For a standard column \([A^p]_x\) and a target column \([\hat{A}^p]_j\), the elements of the two columns satisfy
\[
\lim_{h_p \to 0} (\hat{a}_{i_s,j_s}^p - \kappa_x \hat{a}_{i_s,j_s}^p - g_p^x) = 0, \tag{34}
\]
where \(i_r = i_s\), and \(\kappa_x\) is a constant as
\[
\kappa_x = \frac{(\bar{a}_{i_r}^p(k; T) - a_{x+}^p(k; k; T)) (\sigma_0^p(k+1; T) - \bar{a}_{x-}^p(k+1; T)) \sigma_T^p}{(\sigma_0^p(k; k; S) - \bar{a}_{x+}^p(k; k; S)) (\sigma_0^p(k+1; S) - \bar{a}_{x-}^p(k+1; S)) \sigma_T^p}. \tag{35}
\]

The parameters \(\hat{a}_{i_r,j_s}^p\) and \(g_p^x\) in equation (34) satisfy
\[
\begin{align*}
\hat{a}_{i_r,j_s}^p &= \frac{\mu_T^x - \mu_S^x}{h_p^x}, \\
\mu_T^x &= \frac{1}{2} (\bar{a}_{x}^p(k; T) + \sigma_0^p(k; T)), \\
\mu_S^x &= \frac{1}{2} (\bar{a}_{x+}^p(k; S) + \sigma_0^p(k; S)), \\
g_p^x &= \frac{\mu_T^x - \mu_S^x}{h_p^x}.
\end{align*}
\]
with \(\bar{a}_{x}^p(k; T), \bar{a}_{x+}^p(k; S)\) and \(\bar{a}_{x-}^p(k; S)\) are integer multiples of \(h_p^x\). Therefore, \((\mu_T^x - \mu_S^x)\) is quantized to an integer multiples of \(h_p^x\). Thus, we can obtain
\[
\begin{align*}
\lim_{h_p^x \to 0} (|\mu_T^x - \mu_S^x| - \frac{|\mu_T^x - \mu_S^x|}{h_p^x}) h_p^x &= 0, \\
\lim_{h_p^x \to 0} \left| \frac{|\mu_T^x - \mu_S^x|}{h_p^x} \right| &= 0.
\end{align*}
\]

Then the uniform continuity of the Gaussian distribution function leads to
\[
\begin{align*}
\lim_{h_p^x \to 0} \left| f_G \left( \frac{\bar{a}_{x}^p(k+1; S) + \bar{a}_{x+}^p(k+1; S)}{2} - \mu_T^x, q_p \right) - \mu_S^x - \frac{|\mu_T^x - \mu_S^x|}{h_p^x} \right| &= 0.
\end{align*}
\]

Note that
\[
\begin{align*}
\hat{a}_{i_r,j_s}^p &= \frac{\bar{a}_{x}^p(k; T) + \bar{a}_{x+}^p(k; T)}{2} - \mu_T^x, \\
\hat{a}_{i_r,j_s}^p &= \frac{\bar{a}_{x}^p(k; S) + \bar{a}_{x+}^p(k; S)}{2} - \mu_S^x.
\end{align*}
\]

Equations (40) and (41) show that, a transition probability of the target column \(\hat{a}_{i_r,j_s}^p\) can be seen as a product of a constant \(\tau_p^x(k; T)\) (constant part) and a Gaussian distribution function with variance \(g_p^{x^2}\) (called Gaussian part), while a transition probability of the standard column \(\hat{a}_{i_r,j_s}^p\) can be seen as a product of a constant \(\tau_p^x(k; S)\) (constant part) and a Gaussian distribution function with variance \(g_p^{x^2}\) (Gaussian part). Thus a constant \(\kappa_x\) is formed as \(\kappa_x = \frac{\tau_p^x(k; T)}{\tau_p^x(k; S)}\) to close the gap between the constant parts of different columns.

Then, for Gaussian parts of \(\hat{a}_{i_r,j_s}^p\) and \(\hat{a}_{i_r,j_s}^p\), consider the elements of the standard column and the target column in a same row, where \(i_r = i_s\), \(\bar{a}_{x}^p(k+1; T) = \bar{a}_{x}^p(k+1; S), \bar{a}_{x+}^p(k+1; T) = \bar{a}_{x+}^p(k+1; S)\). It is obvious that
\[
\begin{align*}
f_G \left( \frac{\bar{a}_{x}^p(k+1; T) + \sigma_0^p(k+1; T)}{2} - \mu_T^x, q_p \right) &= f_G \left( \frac{\bar{a}_{x}^p(k+1; S) + \sigma_0^p(k+1; S)}{2} - \mu_T^x, q_p \right),
\end{align*}
\]
For quantized transition probabilities, \((\mu_T^x - \mu_S^x)\) needs to be quantized to an integer multiples of \(h_p^x\), where \(h_p^x = \bar{a}_{x}^p(k) - \bar{a}_{x+}^p(k)\) is the length of quantization sub-intervals in \(\{\hat{a}_{x}^p(k; T), \bar{a}_{x}^p(k; S)\}\). Therefore, \((\mu_T^x - \mu_S^x)\) is quantized as \(g_p^x h_p^x\), where
\[
\begin{align*}
\hat{a}_{i_r,j_s}^p &= \frac{\bar{a}_{x}^p(k) + \bar{a}_{x+}^p(k)}{2} - \mu_T^x, \\
\hat{a}_{i_r,j_s}^p &= \frac{\bar{a}_{x}^p(k) + \bar{a}_{x+}^p(k)}{2} - \mu_S^x.
\end{align*}
\]

Similar to Theorem 3, Theorem 4 is developed for \(\hat{a}_{i_r,j_s}^p\), which is proved in Appendix B.

Theorem 4: For a standard column \([C^p]_x\) and a target
column \([CP]^{j_1}\), the elements of two columns satisfy
\[
\lim_{h_p \to 0} (\gamma_{k,j_1} - \kappa_y \gamma_{k_y,j_1}) = 0,
\]
where \(k_y = k_s\), and \(\kappa_y\) is a constant as
\[
\kappa_y = \frac{(\sigma_p^y(k:T) - \alpha_p^y(k:T))(\gamma_{p_y}(k;T) - y(k;T))f_G(\mu_p^y, \sigma_p^y)}{(\alpha_p^y(k;S) - \alpha_p^y(k;S))(\gamma_{p_y}(k;S) - y(k;S))f_G(\mu_p^y, \sigma_p^y)}.
\] (47)

The parameters \(j_2, j_3\) and \(g_p^y\) in (46) satisfy
\[
\delta_{N_1}^{j_2} = \delta_{N_1}^{j_2}(k;T) \otimes \cdots \otimes \delta_{N_1}^{j_2}(k:T),
\]
\[
\delta_{N_2}^{j_3} = \delta_{N_2}^{j_3}(k;S) \otimes \cdots \otimes \delta_{N_2}^{j_3}(k;S),
\]
\[
g_p^{j_1} = \left[ \frac{\mu_p - \mu_{y_p}}{h_p^y} \right],
\]
\[
\mu_p = \frac{1}{2} \left[ \pi_p^y(k;T) + \alpha_p^y(k;T) \right],
\]
\[
\mu_y = \frac{1}{2} \left[ \pi_p^y(k;S) + \alpha_p^y(k;S) \right],
\]
with \(\pi_p^y(k;T), \pi_p^y(k;S), \pi_p^y(k;T), \pi_p^y(k;S) \notin \{-\infty, -\infty\}\), and \(\pi_p^y(k;S), \pi_p^y(k;T), \pi_p^y(k;S) \notin \{-\infty, -\infty\}\).

Based on the theorems and lemmas above, an improved algorithm is designed as Algorithm 4 which provides us an improved method that utilizes the periodic structural characteristics adequately. Since only one column of each small matrices \(A^1, \ldots, A^n, C^1, \ldots, C^m\) mentioned in Theorem 1 needs to be evaluated in this case, the number of parameters need to be evaluated is significantly reduced for Algorithm 4.

Specifically, Algorithm 4 is mainly composed of the following steps. First, data are generalized and processed based on Algorithms 1 and 2 (Lines 4-11). Second, the state transition process is counted and recorded in the \(\pi^x(k)\) column of matrices \(A^p, p \in \{1, \ldots, n\}\) and \(C^p, p \in \{1, \ldots, m\}\) as standard columns (Lines 12-18). Third, other columns of transition matrices are calculated based on the standard columns using Theorems 3 and 4 (Lines 20-35). Then, the transition matrix \(A^\pi\) and measurement matrix \(C^\pi\) are obtained by the Khatri-Rao product of \(A^p, p \in \{1, \ldots, n\}\) and \(C^p, p \in \{1, \ldots, m\}\), respectively (Lines 36-37). The theoretical basis is provided by Theorems 1 and 2. Finally, the transition matrices \(A^\pi\) and \(C^\pi\) are normalized (Line 38).

Noticeably, from Lines 12-18 of Algorithm 4 only the \(\pi^x(k)\) column of matrices \(A^1, \ldots, A^n, C^1, \ldots, C^m\) are learned through exhaustive training, based on which the entire transition matrices \(A, C\) are further calculated. The number of the parameters learned through exhaustive training in Algorithm 4 equals to \(\sum_{p=1}^n N_p + \sum_{p=1}^m M_p\), which is much smaller than Algorithm 3. As a consequence, the size of training data \(L \times D_x\) needed in Algorithm 4 is also much smaller than that of Algorithm 3.

**Remark 5:** Since the normalization of the matrices is performed at the end of Algorithm 4, multiplying constants for every element of a column before normalization would not change the result. Therefore, \(\kappa_x = \kappa_y = 1\).

**Algorithm 4 Reduced-complexity training of a transition matrix \(A^\pi\) and a measurement matrix \(C^\pi\)**

1. Initialize \(A^\pi, C^\pi\);
2. Initialize the number of loop \(L\) and fix the data size \(D_x = 2\);
3. for \(i = 1\) to \(L\) do
4. Simulate the system (1) to generate \(D_x\) new data with initial state equals to zero vector; (Generally, the data is different in each loop because of the disturbances.)
5. for \(k = 1\) to \(D_x\) do
6. Translate \(x(k), y(k)\) to \(x(k), y(k)\) using Algorithm 1
7. for \(p = 1\) to \(n\), \(q = 1\) to \(m\) do
8. Quantize \(x_p(k), y_q(k)\) to calculate \(\pi^x_p(k)\) and \(\pi^y_p(k)\) according to Algorithm 2
9. end for
10. Calculate \(\pi^x(k)\) according to Theorem 1
11. end for
12. \(k = 1\);
13. for \(p = 1\) to \(m\) do
14. \(e_p^{\pi^x_p(k), \pi^y_p(k)} = e_p^{\pi^x_p(k), \pi^y_p(k)} + 1\);
15. end for
16. for \(p = 1\) to \(n\) do
17. \(a_p^{\pi^x_p(k+1), \pi^y_p(k)} = a_p^{\pi^x_p(k+1), \pi^y_p(k)} + 1\);
18. end for
19. end for
20. for \(j \in \{1, \ldots, \pi^x(k) - 1, \pi^x(k) + 1\}\) do
21. for \(p = 1\) to \(n\) do
22. for \(i = 1\) to \(N_p\) do
23. if \(1 \leq i - g_p^y \leq N_p\) then
24. \(a_{i,j} = a_{i-1,j}^{g_p^y, \pi^y_p(k)}\);
25. end if
26. end for
27. end for
28. for \(p = 1\) to \(m\) do
29. for \(i = 1\) to \(M_p\) do
30. if \(1 \leq i - g_p^y \leq M_p\) then
31. \(a_{i,j} = a_{i-1,j}^{g_p^y, \pi^y_p(k)}\);
32. end if
33. end for
34. end for
35. end for
36. \(A^\pi = A^1 \cdots A^n\); \(C^\pi = C^1 \cdots C^m\);
37. Normalize \(A^\pi\) and \(C^\pi\) as lines 7 to 24 in Algorithm 3.
38. **IV. Numerical Analysis**

In this section, the HMM learning approach proposed in this work is applied to event-based state estimation with an unreliable communication channel through numerical analysis. First, the performance of the low-complexity learning algorithm (Algorithm 4) is compared with that of Algorithm 3 to verify the validity of parameter estimation methods developed. Then, the HMMs learned using the proposed algorithms are applied to event-triggered state estimation and the estimation
results are compared with existing results in the literature. For comparison purpose, we ignore the numerical errors caused by system identification and assume the model obtained in [1] is accurate.

A. Comparison between Algorithm [3] and Algorithm [4]

Suppose the parameters in system (1) are chosen as
\[
A = \begin{bmatrix} 0.8 & 0.2 \\ 0.5 & 0.3 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 1 \end{bmatrix} \tag{48}
\]
and the variances of disturbances are selected as \( R = 0.01 \) and \( Q = \text{diag}(0.1, 0.1) \), respectively. In this subsection, let \( N_1 = N_2 = 64 \), \( M = 1024 \).

The HMM parameters are learned using the naive method in Algorithm [3] and the improved method in Algorithm [4]. The training results of Algorithm [3] are noted as \( a_{i,j}^+, c_{i,j}^+ \) (or \( A^+, C^+ \)), while the parameters learnt by Algorithm [4] are noted as \( a_{i,j}^−, c_{i,j}^− \) (or \( A^−, C^− \)). A comparison of the training results is shown in Fig. 5. The small differences between \( a_{i,j}^+, c_{i,j}^+ \) and \( a_{i,j}^−, c_{i,j}^− \) illustrate the validity of Algorithm [4] since the convergence of Algorithm [3] is guaranteed by the law of large numbers.

Fig. 5: Comparison of a column of the HMM obtained using Algorithm [3] and Algorithm [4]

B. Applications to Event-triggered Estimation

The task of event-triggered state estimation is to estimate the value of state \( x(k) \) based on event-triggered measurement information. In this work, the measurement is sent to the estimator according to a pre-specified send-on-delta schedule:
\[
\xi(k) = \begin{cases} 
0, & ||y(k) - y(\tau)||_2 < \delta, \\
1, & \text{otherwise}, 
\end{cases} \tag{49}
\]
where \( y(\tau) \) is the previous measurement received by the estimator. Since packet dropouts are normally inevitable for a practical communication channel, the remote estimator may not receive the measurements sent out by the sensor (i.e., when \( \xi(k) = 1 \)). In this work, the packet dropout effect is modeled by an independent and identically distributed Bernoulli process which is independent of \( X(k), Y(k) \) and \( \xi(k) \) satisfying
\[
\zeta(k) = \begin{cases} 
0, & \text{there exists a packet loss at time } k, \\
1, & \text{there exists no packet loss at time } k, 
\end{cases} \tag{50}
\]
\[
P(\zeta(k) = 1) = \lambda, \quad P(\zeta(k) = 0) = 1 - \lambda. \tag{51}
\]

To represent the recipient of the measurement information, we write
\[
\gamma(k) = \xi(k) \cdot \zeta(k) = \begin{cases} 
1, & \text{estimator receives } y(k), \\
0, & \text{otherwise}. 
\end{cases} \tag{52}
\]

The average communication rate \( \eta \) is defined as
\[
\eta := \frac{1}{N_d} \sum_{k=1}^{N_d} \gamma(k), \tag{53}
\]
where \( N_d \) is the length of the data set.

In this work, three state estimators are compared. The first estimator is Kalman filter with intermittent observations, which is the MMSE state estimator that ignores the unreceived measurement information (see equations (2)-(6) and (13)-(14) in [17]). The second and third estimators are HMM-based state estimators, which are designed using the HMMs obtained by using Algorithm [3] and Algorithm [4] respectively (see equations (24) and (54) in [13]).

To describe the estimation performance of state estimators, we define the average estimation error \( E_K, E_{H+} \) and \( E_{H−} \) as
\[
E_K = \frac{1}{N_d} \sum_{k=1}^{N_d} \|\hat{x}(k) - x^*(k)\|_2, \\
E_{H+} = \frac{1}{N_d} \sum_{k=1}^{N_d} \|\hat{x}^+(k) - x^+(k)\|_2, \tag{54}
\]
\[
E_{H−} = \frac{1}{N_d} \sum_{k=1}^{N_d} \|\hat{x}^−(k) - x^−(k)\|_2,
\]
where \( E_{H+} \) is the error of HMM estimator based on \( A^+ \) and \( C^+ \), \( E_{H−} \) is the error of HMM estimator based on \( A^− \) and \( C^− \), and \( E_K \) is the error of the Kalman filter with intermittent observations, and \( x^+, \hat{x}, \hat{x}^+ \) and \( \hat{x}^− \) denote the real value, the estimate of the Kalman filter, the estimate using \( A^+, C^+ \) and the estimate using \( A^−, C^− \), respectively.

Fig. 6: State estimation comparing HMM estimator and Kalman filter, \( \eta = 1 \).
First, we consider a limiting case that $\delta$ and $\lambda$ are set to 0 such that the limiting scenario that the measurements are available for all time instants, to compare the performance of the HMM-based estimators with the standard Kalman filter. The results are shown in Fig. 6. In this case, $E_{H+} = 0.261$, $E_{H-} = 0.264$, and $E_{K} = 0.254$, which indicates that HMM-estimators offer good performance for state estimation, nearly as good as the Kalman filter. The consistence observed between $\hat{x}^+$ and $\hat{x}^-$ illustrates the validity of the proposed algorithm (Algorithm 4).

Second, state estimators are applied to event-triggered state estimation through an unreliable channel with $\delta = 0.4081$ and $\lambda = 0.95$, and the results are shown in Fig. 7 with the average communication rate being 38.5%. In Fig. 7 the estimation error $E_{H+} = 0.291$, $E_{H-} = 0.296$, and $E_{K} = 0.302$. This implies that the HMM-based estimators perform better than the Kalman filter, because of the exploration of the implicit information contained in the event-triggering schedule.

The results are shown in Fig. 6. In this case, $E_{H+} = 0.261$, $E_{H-} = 0.264$, and $E_{K} = 0.254$, which indicates that HMM-estimators offer good performance for state estimation, nearly as good as the Kalman filter. The consistence observed between $\hat{x}^+$ and $\hat{x}^-$ illustrates the validity of the proposed algorithm (Algorithm 4).

Second, state estimators are applied to event-triggered state estimation through an unreliable channel with $\delta = 0.4081$ and $\lambda = 0.95$, and the results are shown in Fig. 7 with the average communication rate being 38.5%. In Fig. 7 the estimation error $E_{H+} = 0.291$, $E_{H-} = 0.296$, and $E_{K} = 0.302$. This implies that the HMM-based estimators perform better than the Kalman filter, because of the exploration of the implicit information contained in the event-triggering schedule.

![Fig. 7: State estimation comparing HMM estimator and Kalman filter, $\delta = 0.4081$, $\eta = 0.385$.](image)

To further compare the performance of HMM-based estimator using Algorithm 4 and the Kalman filter with intermittent observations, we define

$$E_c = (E_K - E_{H^-})/E_{K^*},$$

where $E_{K^*}$ is the estimation error of Kalman filter when every measurement is received. To evaluate the performance of the event-based estimators, we range the communication rate $\eta$ from 0 to 1 by varying the parameter $\delta$ in the event-triggering condition and therefore a sequence of estimation errors for each schedule can be obtained, based on which the tradeoff curves between estimation performance and communication rate are obtained. The results are presented in Fig. 8 which shows that the HMM estimator achieves improved performance in terms of average estimation error compared with Kalman filter with intermittent observations, especially when the communication rate is less than 0.5. The index $E_c$ is less than zero when the communication rate $\eta$ belongs to $(0.6, 0.95)$, which could be caused by the hybrid effect of the numerical calculation error and the modeling error.

![Fig. 8: Tradeoff between estimation performance and trigger rate $\eta$ when $\lambda = 0.95$ and $\varphi = 5$.](image)

V. CONCLUSION

In this work, an indirect approach is designed to estimate the parameters of an HMM for a linear Gaussian system. A low-complexity parameter learning algorithm is proposed based on the periodical structural characteristics of the HMM. The convergence of our algorithms is proved using a numerical integration method. Numerical results on model learning and event-triggered state estimation illustrate the motivation and validity of the proposed results. In our next step, the problem of simultaneous HMM modeling and state estimation for a linear Gaussian system will be investigated.

APPENDIX A

PROOF OF THEOREM 4

Proof: Considering that the covariance matrix $R$ is a diagonal matrix, the independence between different elements leads to

$$P\left( Y(k) = \delta_N^y(k) | X(k) = \delta_N^y \right)$$

$$= \prod_{p=1}^{m} P\left( Y_p(k) = \delta_M^y_p | X(k) = \delta_N^y \right),$$

thus we obtain

$$c_{y_p(k),j} = \prod_{p=1}^{m} c_{y_p(k),j}.$$

Noting that

$$Y_p(k) = \delta_M^y_p,$$

the combination method (11) leads to

$$\delta_M^y(k) = \delta_M^y_1 \otimes \cdots \otimes \delta_M^y_m.$$

Thus we obtain

$$[C]^j = [C^1]^j \otimes \cdots \otimes [C^m]^j.$$

Then we have

$$C = C^1 \ast C^2 \ast \cdots \ast C^m$$
based on the definition of Khatri-Rao product, which completes the proof.

\[ \begin{align*}
\text{APPENDIX B} \\
\text{PROOF OF THEOREM} \\
\end{align*} \]

**Proof:** Based on Lemma 3, we obtain
\[ \begin{align*}
\tau^p_{i,s,j}\ell_t &= \left( \frac{\mu^p_s(T;k) - \alpha^p_s(k;\ell_t) \mu^p_s(k;T)}{2} \right) \left( \frac{\mu^p_s(T;k) - \mu^p_s(k;\ell_t)}{2} \right) \\
&= f_G \left( \frac{\mu^p_s(k;T) + y^p_{p}(k;T)}{2} - \mu^p_s, r_p \right) \\
&= f_G \left( \frac{\mu^p_s(k;T) + y^p_{p}(k;T)}{2} - \mu^p_s, r_p \right).
\end{align*} \] (64)

For quantized transition probabilities, \((\mu^p_s - \mu^p_s)\) needs to be quantized to an integer multiple of \(h^p_p\), where \(h^p_p = \overline{\mu^p_p}(k) - \underline{\mu^p_p}(k)\) is the length of quantization sub-intervals in (9). Therefore, \((\mu^p_s - \mu^p_s)\) is quantized as \(g^p_p h^p_p\), where \(g^p_p = \left\lfloor \frac{\mu^p_s - \mu^p_s}{h^p_p} \right\rfloor \). Since
\[ \left| (\mu^p_s - \mu^p_s) - g^p_p h^p_p \right| = \left| (\mu^p_s - \mu^p_s) - \frac{\mu^p_s - \mu^p_s}{h^p_p} \right| h^p_p \leq h^p_p, \]
we obtain
\[ \begin{align*}
\lim_{h^p_p \to 0} \left| (\mu^p_s - \mu^p_s) - g^p_p h^p_p \right| &= 0, \\
\lim_{h^p_p \to 0} \left[ \frac{\overline{\mu^p_p}(k;S) + y^p_{p}(k;S)}{2} - \frac{\mu^p_s - (\mu^p_s - \mu^p_s)}{2} - g^p_p h^p_p, r_p \right] &= 0.
\end{align*} \] (65)

Then the uniform continuity of the Gaussian distribution function leads to
\[ \begin{align*}
\lim_{h^p_p \to 0} \left( \frac{\overline{\mu^p_p}(k;S) + y^p_{p}(k;S)}{2} - \mu^p_s - g^p_p h^p_p, r_p \right) &= 0. \] (66)

Note that
\[ \begin{align*}
\tau^p_{i,s,j} &= \overline{\mu^p_p}(k;T) f_G \left( \frac{\overline{\mu^p_p}(k;T) + y^p_{p}(k;T)}{2} - \mu^p_s, r_p \right) \\
\tau^p_{i,s,j} &= \overline{\mu^p_p}(k;S) f_G \left( \frac{\overline{\mu^p_p}(k;S) + y^p_{p}(k;S)}{2} - \mu^p_s, r_p \right),
\end{align*} \] (67)

Equation (66) can be obtained by combining equations (64)-(67). Thus the claim holds.

**REFERENCES**

[1] J. P. Respanha, P. Naghshtabrizi, and Y. Xu, “A survey of recent results in networked control systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–162, 2007.

[2] K. J. Åström and B. Bernhardsson, “Comparison of Riemann and Lebesgue sampling for first order stochastic systems,” in *Proceedings of the 41st IEEE Conference on Decision and Control*, vol. 2, pp. 2011–2016.

[3] W. Heemels, K. Johansson, and P. Tabuada, “An introduction to event-triggered and self-triggered control,” in *51st IEEE Conference on Decision and Control (CDC)*, 2012, pp. 3270–3285.

[4] O. Imer and T. Başar, “Optimal estimation with limited measurements,” in *Proceedings of the 44th IEEE Conference on Decision and Control*.

[5] M. Rabi, G. Moustakides, and J. Baras, “Adaptive sampling for linear state estimation,” *SIAM Journal on Control and Optimization*, vol. 50, no. 2, pp. 672–702, 2012.
