Chapter 2

Interval-Based Hybrid Dynamical System

In this chapter, we introduce an interval-based hybrid dynamical system (interval system). The system consists of a finite state automaton and a set of multiple linear dynamical systems as we described in the previous chapter. Each linear dynamical system represents a dynamic primitive that corresponds to a discrete state of the automaton; meanwhile the automaton controls the activation timing of the dynamical systems. Thus, the interval system can generate and analyze complex multivariate sequences that consist of temporal regimes of dynamic primitives (see Figure 1.9 for the example).

2.1 System Architecture

An interval system has a two-layer architecture (Figure 2.1). The first layer (the top dashed box in Figure 2.1) has a finite state automaton as a discrete-event system that models stochastic transitions between discrete events. The second layer (the second-top dashed box in Figure 2.1) consists of a set of linear dynamical systems $\mathcal{D} = \{D_1, ..., D_N\}$. To integrate these two layers, we introduce *intervals* (the middle of Figure 2.1); each interval is described by $< q_i, \tau >$, where q_i denotes a discrete state in the automaton and τ denotes the physical temporal duration length of the interval.

As we described in Subsection 1.4.3, the ending points of the intervals can be considered to be the discrete events that the automaton models. While the automaton models only the order of discrete events without physical-time metric,

the intervals provides physical-time grounding for the automaton due to the duration length τ.

We assume that a dynamical system D_i characterizes the type of dynamics in the interval $< q_i, \tau >$. Therefore, each state in the automaton corresponds to a unique linear dynamical system in the second layer; that is, q_i denotes the label of the corresponding linear dynamical system as well as a state in the automaton. Note that multiple different intervals can correspond to the same state in the automaton (i.e., their dynamics are described by the same linear dynamical system).

Signal Generation and Segmentation

An interval system is a stochastic generative model. Once the interval system has been constructed by learning as will be described in Chapter 3, it can generate a multivariate signal sequence by activating the automaton. The activated automaton first generates a sequence of intervals (the middle of Figure 2.1), each of which then generates a signal sequence based on its corresponding linear dynamical system (the second bottom of Figure 2.1). Note that the activation timing and period of the linear dynamical system are controlled by the duration length of the interval.

When a temporal sequence of observed signal data (multivariate sequence) is given, the system finds the activation timing and period of the linear dynamical systems based on the likelihood calculation. That is, the observed sequence is partitioned into a group of sub-sequences so that the dynamic signal variation in each sub-sequence can be described by a linear dynamical system, which is denoted by the discrete-state label of the interval covering that sub-sequence (see Section 2.4 for details). As a result, the observation sequence is transformed into a sequence of internal states that is partitioned by an interval sequence.

Notations

We define some terms and notations for later discussions. Firstly, we simply use the term "dynamical systems" to denote linear dynamical systems.

Internal state. All the constituent dynamical systems are assumed to share an n-dimensional internal state space. Each activated dynamical system can generate sequences of real valued internal state vector $x \in \mathbf{R}^n$, which can be

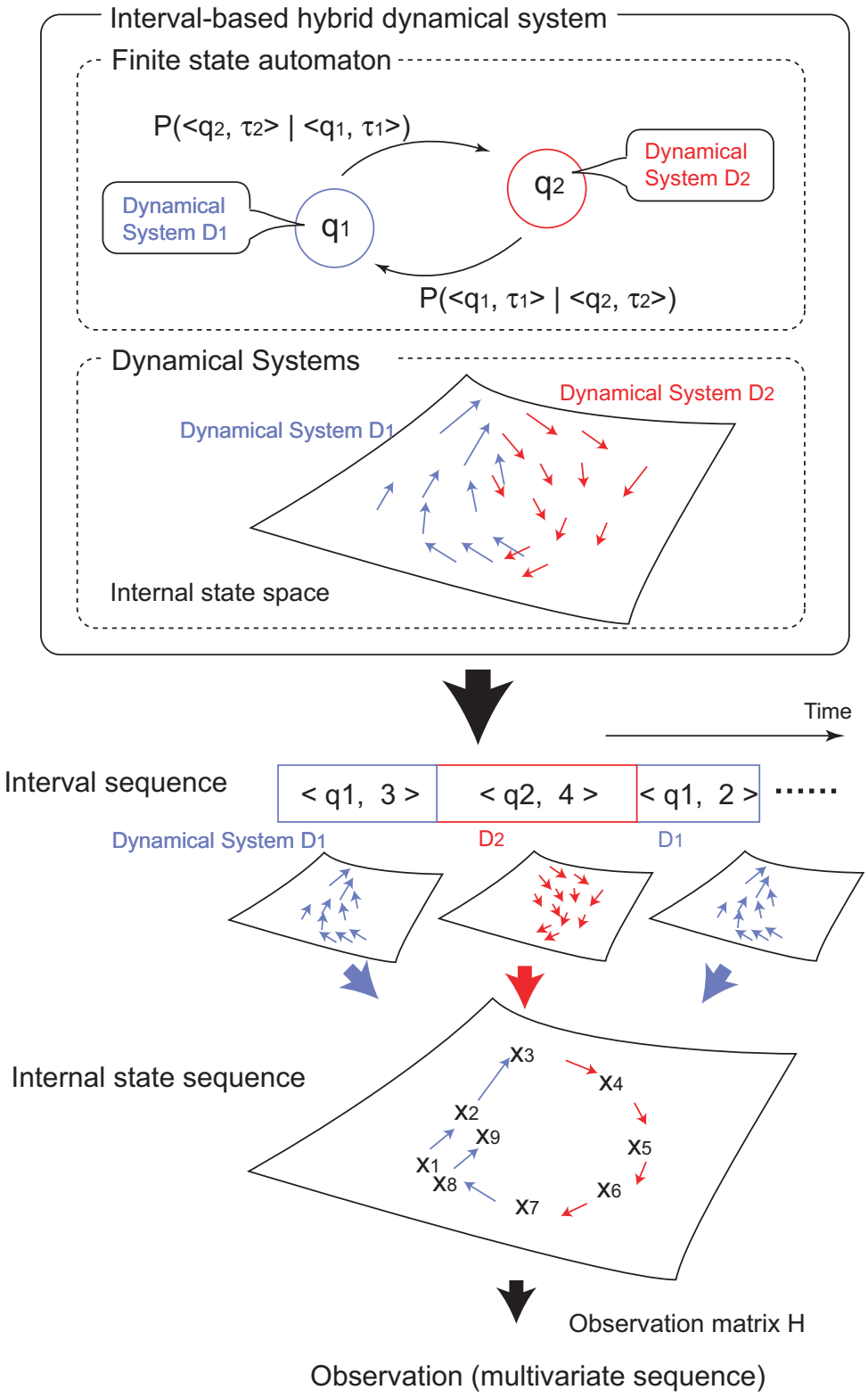


Figure 2.1: Interval-based hybrid dynamical system and the generation of a multivariate sequence.

mapped onto the observation space by a linear function. We assume such linear transformation function is also shared by all the dynamical systems.

Observation. An observation sequence is described by a multivariate vector $y \in \mathbf{R}^m$ sequence in a m-dimensional observation space.

Discrete state. The finite state automaton has a discrete state set $Q = \{q_1, ..., q_N\}$. Each state $q_i \in \mathcal{Q}$ corresponds to the dynamical system D_i, respectively.

Duration lengths of intervals. The duration length that an interval continues described by a positive integer because we assume the interval system as a discrete time model. To reduce parameter size, we set a minimum duration length $l_{\min}$ and a maximum duration length $l_{\max}$; we define a duration length as $\tau \in \mathcal{T} \triangleq \{l_{\min}, ..., l_{\max}\}$.

Interval. An interval generated by the automaton is defined as a combination of a discrete state and a duration length. We use notation $< q_i, \tau > \in \mathcal{Q} \times \mathcal{T}$ to represent the interval that has state q_i and duration τ.

2.2 Linear Dynamical Systems

2.2.1 Formulation

The state transition of dynamical system D_i in the internal state space, and the mapping from the internal state space to the observation space is modeled as the following linear equations:

$$\begin{aligned} x_t &= F^{(i)} x_{t-1} + g^{(i)} + \omega_t^{(i)} \\ y_t &= H x_t + v_t, \end{aligned} \tag{2.1}$$

where $F^{(i)}$ is a transition matrix and $g^{(i)}$ is a bias vector. H is an observation matrix that defines linear projection from the internal state space to the observation space. $\omega^{(i)}$ and v is the process noise and the observation noise. Note that each dynamical system has $F^{(i)}$, $g^{(i)}$, and $\omega_t^{(i)}$ individually. We assume each of noise term $\omega^{(i)}$ and v has Gaussian distribution $\mathcal{N}_{x_t}(0, Q^{(i)})$ and $\mathcal{N}_{y_t}(0, R)$, respectively. Here, we use the notation $\mathcal{N}_x(a, B)$ to denote a Gaussian distribution that has

average vector a and covariance matrix B in the space of variable x:

$$\mathcal{N}_x(a, B) = (2\pi)^{-n/2}|B|^{-1/2} \exp\left\{-\frac{1}{2}(x-a)^\top B^{-1}(x-a)\right\}, \tag{2.2}$$

where n is a dimension of vector x.

We assumed that all the dynamical systems share a single internal state space. The main reason is that we want to reduce parameters in the interval system; it is, however, possible to design the system with an individual internal state space for each dynamical system. In such cases, observation parameters $H^{(i)}$ and $R^{(i)}$ are required for each dynamical system. Although they provide more flexibility in models, a large parameter space causes problems such as over-fitting and high computational costs.

Probability Density Distributions

Using the formulation and notation mentioned above, we can consider probability density distribution as follows:

$$\begin{aligned} p(x_t|x_{t-1}, s_t = q_i) &= \mathcal{N}_{x_t}(F^{(i)}x_{t-1} + g^{(i)}, Q^{(i)}) \\ p(y_t|x_t, s_t = q_i) &= \mathcal{N}_{y_t}(Hx_t, R), \end{aligned} \tag{2.3}$$

where the probability variable s_t is an activated discrete state at time t (i.e., dynamical system D_i is activated). The second equation is independent of the probability variable s_t because of the assumption in the previous paragraph. In this thesis, we use p to denote probability density function and P for probability.

Since the state distribution is recursively calculated by the density distributions above, we define the initial state distribution as follows:

$$p(x_1|, s_1 = q_i) = \mathcal{N}_{x_1}(x^{(i)}_{\text{init}}, V^{(i)}_{\text{init}}). \tag{2.4}$$

2.2.2 Class of Linear Dynamical Systems

The class of linear dynamical systems can be categorized by the eigenvalues of the transition matrix, which determine the zero-input response of the system. In other word, these eigenvalues determine the behavior of generable time-varying patterns (trajectories) in the state space.

Without Bias Term

To concentrate on the temporal evolution of the state in the dynamical system, let us assume the bias and the process noise term is zero in Equation (2.1). Using the eigenvalue decomposition of the transition matrix:

$$F = E\Lambda E^{-1} = [e_1, ..., e_n]\text{diag}(\lambda_1, ..., \lambda_n)[e_1, ..., e_n]^{-1},$$

we can solve the state at time t with initial condition x_0:

$$x_t = F^t x_0 = (E\Lambda E^{-1})^t x_0 = E\Lambda^t E^{-1} x_0 = \sum_{p=1}^{n} \alpha_p e_p \lambda_p^t, \tag{2.5}$$

where e_p and λ_p is a corresponding eigenvalue and eigenvector pair. We omit the indices i for simplification. A weight value α_p is determined from the initial state x_0 by calculating $[\alpha_1, ..., \alpha_n]^\top = E^{-1}x_0$.

Hence, the generable patterns from the system can be categorized by the position of the eigenvalues (poles) $\lambda_1, ..., \lambda_n$ on the complex plane. Especially, the arguments (angle) of eigenvalues in a complex plain determine the state will oscillate or not:

- At least one negative or complex eigenvalue exists → oscillating.
- All the eigenvalues have real number → non-oscillating.

On the other hand, the absolute values of eigenvalues determine the state will converge or not:

- At least one absolute value of eigenvalue exceeds one → diverging.
- All the absolute values of eigenvalues are smaller than one → converging.

Figure 2.2 shows examples of state trajectories when the dimensionality of the state is two.

For instance, the system can generate time-varying patterns that converge to zero if and only if $|\lambda_p| < 1$ for all $1 \leq p \leq n$ (using the term in control theory, we can say that the system is stable); meanwhile, the system can generate non-monotonic or cyclic patterns if the imaginary parts of eigenvalues have nonzero values.

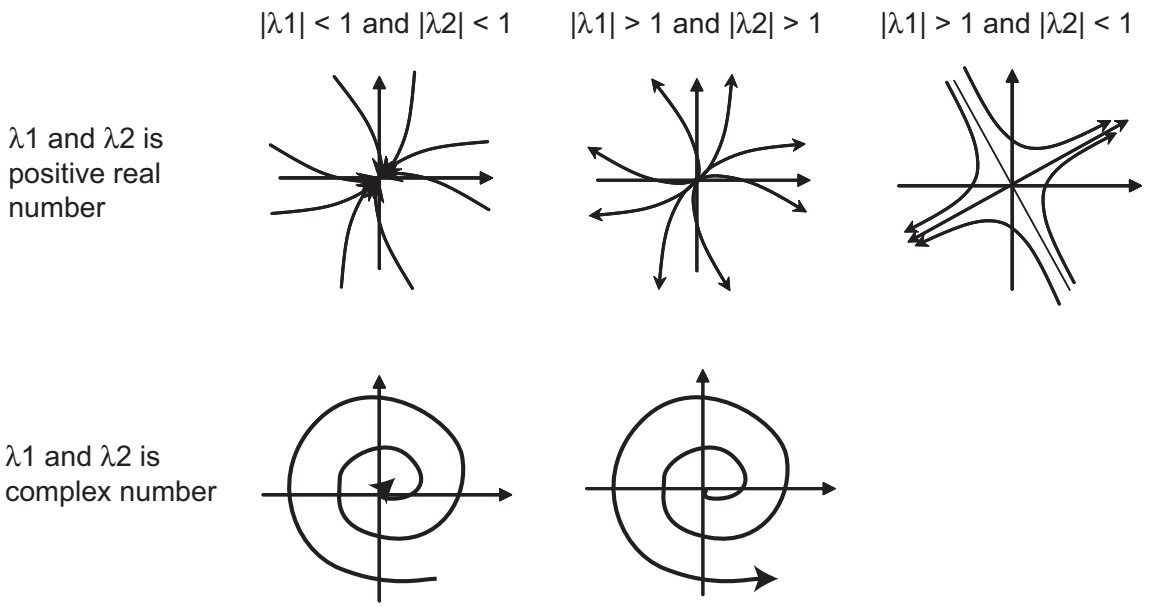


Figure 2.2: Examples of dynamics class when the dimensionality of the state is two.

With Bias Term

We first consider the dynamical system has converging behavior. In case that the system equation has bias vector g as shown in Equation (2.1), the state converges to a certain position x_{conv} in the state space. We can calculate state x_{conv} using a similar method to linear type recurrence equations.

Let us assume that the process is not stochastic but deterministic (i.e., noise term is zero) same as the previous paragraph. Substituting x_{conv} for x_t and x_{t-1} in Equation (2.1), we get the following equation:

$$x_{\text{conv}} = Fx_{\text{conv}} + g. \tag{2.6}$$

From the equation above, the convergence point becomes:

$$x_{\text{conv}} = (I - F)^{-1}g. \tag{2.7}$$

Calculating subtraction of each term between original Equation (2.1) and Equa-

tion (2.6), we get the following equation:

$$\begin{aligned} x_t - x_{\text{conv}} &= F(x_{t-1} - x_{\text{conv}}) \\ &= F^t(x_0 - x_{\text{conv}}) \end{aligned} \tag{2.8}$$

Here, the Equation (2.8) determine the temporal evolution of the state when the state converges to x_{conv}. From Equation (2.7) and Equation (2.8), we get

$$\begin{aligned} x_t &= F^t x_0 + (I - F^t)(I - F)^{-1} g \\ &= E\Lambda^t E^{-1} x_0 + E(I - \Lambda)^{-1}(I - \Lambda^t)E^{-1} g. \end{aligned} \tag{2.9}$$

In general case (i.e., the state might diverge), we can recursively apply the Equation (2.1) and get the following equation:

$$\begin{aligned} x_t &= Fx_{t-1} + g = F(Fx_{t-2} + g) + g \\ &= F^t x_0 + (\sum_{u=1}^{t-1} F^u) g \end{aligned} \tag{2.10}$$

Substituting $\sum_{u=1}^{t-1} F^u = (I - F^t)(I - F)^{-1}$ for the second term of the equation above, we get the same equation as Equation (2.9). Thus, the Equation (2.9) is a general (i.e., independent of eigenvalues) equation for the temporal evolution of the state. We can easily deduce Equation (2.7) from Equation (2.9) as a special case when $\lim_{t\to\infty} F^t = O$ (i.e., all the eigenvalues are smaller than one).

2.2.3 Probabilistic State Inference

In this section, we show the probabilistic inference of the internal state in linear dynamical systems. Let us assume that the internal state has a Gaussian distribution at each time points. Then, the transition of the internal state becomes a Gauss-Markov process, which is inferable in the same manner as Kalman filtering [AM79].

The inference consists of the following two steps:

1. Prediction step

2. Observation (Correction) step

In the next two paragraphs, we describe each of the steps.

Prediction Step

Because we assumed that the probability density of the internal state is a Gaussian distribution, the state distribution of time $t-1$ under the condition of observation from 1 to $t-1$ is represented by the following equation:

$$p(x_{t-1}|y_1^{t-1} = \hat{y}_1^{t-1}, s_{t-1} = q_i) = \mathcal{N}_{x_{t-1}}(x_{t-1|t-1}^{(i)}, V_{t-1|t-1}^{(i)}), \tag{2.11}$$

where $x_{t-1|t-1}^{(i)}$ is a mean vector and $V_{t-1|t-1}^{(i)}$ is a covariance matrix, and $\hat{y}_1^{t-1} = \hat{y}_1, ..., \hat{y}_{t-1}$ is an observation sequence from 1 to $t-1$.

Using Equation (2.3) and (2.11), we can calculate the predicted state distribution under the condition of observations from 1 to $t-1$ as follows:

$$\begin{aligned} p(x_t|y_1^{t-1} = \hat{y}_1^{t-1}, s_t = q_i) &= \int_{x_{t-1}} p(x_t|x_{t-1}, s_t = q_i) p(x_{t-1}|y_1^{t-1} = \hat{y}_1^{t-1}, s_{t-1} = q_i) \\ &= \int_{x_{t-1}} \mathcal{N}_{x_t}(F^{(i)}x_{t-1} + g^{(i)}, Q^{(i)}) \mathcal{N}_{x_{t-1}}(x_{t-1|t-1}^{(i)}, V_{t-1|t-1}^{(i)}) \\ &= \mathcal{N}_{x_t}(x_{t|t-1}^{(i)}, V_{t|t-1}^{(i)}), \end{aligned} \tag{2.12}$$

$$\text{where} \quad \begin{cases} x_{t|t-1}^{(i)} &= F^{(i)}x_{t-1|t-1}^{(i)} + g^{(i)} \\ V_{t|t-1}^{(i)} &= F^{(i)}V_{t-1|t-1}^{(i)}{F^{(i)}}^\top + Q^{(i)} \end{cases}$$

We can also calculate the predicted observation distribution using the predicted state distribution and Equation (2.3):

$$\begin{aligned} p(y_t|y_1^{t-1}, s_t = q_i) &= \int_{x_t} p(y_t|x_t, s_t = q_i) p(x_t|y_1^{t-1}, s_t = q_i) \\ &= \int_{x_t} \mathcal{N}_{y_t}(Hx_t, R) \mathcal{N}_{x_t}(x_{t|t-1}^{(i)}, V_{t|t-1}^{(i)}) \\ &= \mathcal{N}_{y_t}(y_{t|t-1}^{(i)}, M_{t|t-1}^{(i)}), \end{aligned} \tag{2.13}$$

$$\text{where} \quad \begin{cases} y_{t|t-1}^{(i)} = Hx_{t|t-1}^{(i)} \\ M_{t|t-1}^{(i)} = HV_{t|t-1}^{(i)}H^\top + R \end{cases}$$

Observation Step

After the prediction step in the previous paragraph, the state distribution at time t can be calculated once the observation data y_t becomes available. Using Bayesian rule with Equation (2.3), (2.12), and (2.13), we can update the state distribution at

time t under the condition of observations from time 1 to t as follows:

$$\begin{aligned} p(x_t|y_1^t = \hat{y}_1^t, s_t = q_i) &= \frac{p(y_t|x_t)p(x_t|y_1^{t-1} = \hat{y}_1^{t-1}, s_t = q_i)}{p(y_t|y_1^{t-1} = \hat{y}_1^{t-1}, s_t = q_i)} \\ &= \frac{\mathcal{N}_{y_t}(Hx_t, R)|_{y_t=\hat{y}_t}\mathcal{N}_x(x_{t|t-1}^{(i)}, V_{t|t-1}^{(i)})}{\mathcal{N}_{y_t}(y_{t|t-1}^{(i)}, M_{t|t-1}^{(i)})|_{y_t=\hat{y}_t}} \\ &= \mathcal{N}_{x_t}(x_{t|t}^{(i)}, V_{t|t}^{(i)}) \end{aligned} \tag{2.14}$$

$$\text{where} \quad \begin{cases} x_{t|t}^{(i)} &= x_{t|t-1}^{(i)} + K_t^{(i)}(\hat{y}_t - y_{t|t-1}^{(i)}) \\ V_{t|t}^{(i)} &= ({V_{t|t-1}^{(i)}}^{-1} + H^\top R^{-1} H)^{-1} \\ &= (I - K_t^{(i)} H) V_{t|t-1}^{(i)} \\ K_t^{(i)} &= V_{t|t}^{(i)} H^\top R^{-1} \end{cases}$$

Hence, the mean vectors $x_{t|t-1}^{(i)}, x_{t|t}^{(i)}, y$ and covariance matrices $V_{t|t-1}^{(i)}, V_{t|t}^{(i)}$ are updated every sampled time t using the prediction and observation steps by turns.

2.2.4 Likelihood Calculation of the Linear Dynamical System

Now, we show how to calculate the likelihood of a linear dynamical system with respect to the observation sequence in an interval.

Suppose that the dynamical system D_i represents an observation sequence $\hat{y}_{t-\tau+1}^t \triangleq \hat{y}_{t-\tau+1}, ..., \hat{y}_t$ in the interval $< q_i, \tau >$, which has a duration length τ. Then, the likelihood score of the system D_i with respect to the observation sequence $\hat{y}_{t-\tau+1}^t$ is calculated by the following equation:

$$\begin{aligned} d_{[t-\tau+1,t]}^{(i)} &\triangleq P(y_{t-\tau+1}^t = \hat{y}_{t-\tau+1}^t | < q_j, \tau >) \\ &= \prod_{t'=t-\tau+1}^{t} \gamma^m \; p(y_{t'} = \hat{y}^{t'} | y_{t-\tau+1}^{t'-1} = \hat{y}_{t-\tau+1}^{t'-1}, s_{t'} = q_j), \end{aligned} \tag{2.15}$$

where we assume Gaussian distribution $N(x_{\text{init}}^{(i)}, V_{\text{init}}^{(i)})$ for the initial state distribution in the interval as we described in Subsection 2.2.1 (see Equation (2.4)); that is, we substitute $p(y_{t'} = \hat{y}^{t'} \,|\, y_{t-\tau+1}^{t'-1} = \hat{y}_{t-\tau+1}^{t'-1}, s_{t'} = q_j)$ with $\mathcal{N}_{y_t'}(Hx_{\text{init}}^{(i)}, \quad HV_{\text{init}}^{(i)}H^\top + R)$ when $t' = t - \tau + 1$. On the other hand, γ^m is a volume size of observations in the observation space to convert probability density values to probabilities. m is the dimensionality of observation vectors. γ is

assumed to be provided manually based on the size of the observation space (the range of each element in observation vectors). This likelihood score is used in Section 2.4 to evaluate the fitting of linear dynamical system to the given multivariate sequences.

Substituting Equation (2.13) into Equation (2.15), we finally get the likelihood of linear dynamical system D_i under the assumption of the Gauss-Markov process:

$$
\begin{aligned}
& d^{(i)}_{[t-\tau+1,t]} = \prod_{t'=t-\tau+1}^{t} \gamma^m \mathcal{N}_{y_{t'}}(y^{(i)}_{t'|t'-1}, M^{(i)}_{t'|t'-1})|_{y_{t'}=\hat{y}_{t'}} \\
= \; & \frac{\gamma^m}{(2\pi)^{m/2}} \prod_{t'=t-\tau+1}^{t} |M^{(i)}_{t'|t'-1}|^{-1/2} \exp\left\{ -\frac{1}{2} (\hat{y}_{t'} - y^{(i)}_{t'|t'-1})^\top M^{(i)\,-1}_{t'|t'-1} (\hat{y}_{t'} - y^{(i)}_{t'|t'-1}) \right\} \\
= \; & \exp\left\{ \tau m \log \frac{\gamma}{\sqrt{2\pi}} - \frac{1}{2} \sum_{t'=t-\tau+1}^{t} \left(\log |M^{(i)}_{t'|t'-1}| + e_{t'}^\top M^{(i)\,-1}_{t'|t'-1} e_{t'} \right) \right\}, \\
& \text{where} \quad e_{t'} = \hat{y}_{t'} - y^{(i)}_{t'|t'-1}.
\end{aligned} \tag{2.16}
$$

Note that we use $y^{(i)}_{t'|t'-1} = H x^{(i)}_{\text{init}}$ and $M^{(i)}_{t'|t'-1} = H V^{(i)}_{\text{init}} H^\top + R$ for the initial distribution parameters at $t' = t - \tau + 1$.

2.3 Interval-Based State Transitions of the Automaton

2.3.1 Interval-Based State Transition

In this section, we define transition of discrete states in the automaton that generate interval sequences. Here, we assume the first-order Markov property for the generated intervals. The difference from conventional state transition models, such as hidden Markov models, is that the automaton models not only the transition of discrete states but also the correlation between the adjacent interval duration lengths.

Let $\mathcal{I} = I_1, ..., I_K$ be an interval sequence generated by the automaton. To simply the model, we assume that the adjacent intervals have no temporal gaps or overlaps. Here, the interval I_k depends on only the previous interval I_{k-1} because of the Markov property assumption. Then, the Markov process of intervals can

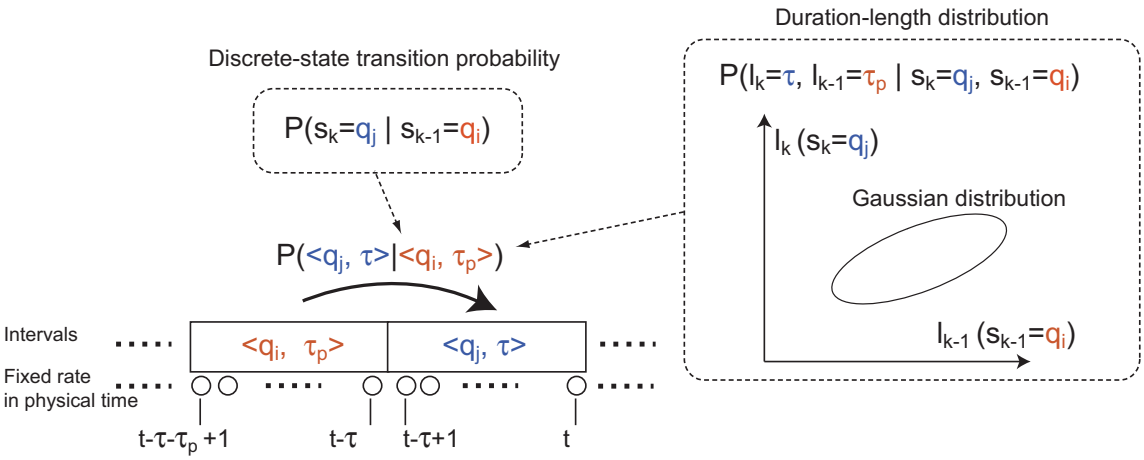


Figure 2.3: First-order Markov property is assumed for a sequence of intervals.

be modeled by the following conditional probability:

$$P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >),$$

where it denotes that the interval $< q_j, \tau >$ occurs after the interval $< q_i, \tau_p >$ (see Figure 2.3).

The probability $P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >)$ requires a large parameter set, which cause not only computational costs but also the problem of over-fitting during a training phase. We therefore use a parametric model for the *duration-length distribution*:

$$h^{(ij)}(l_k, l_{k-1}) \triangleq P(l_k, l_{k-1} | s_k = q_j, s_{k-1} = q_i), \tag{2.17}$$

where the two-dimensional distribution models a joint probability density function of duration lengths in adjacent interval pairs that has state q_i and q_j in this order. s_k and l_k is a probability variable of the discrete state and the duration length in the interval I_k, respectively.

We can assume an arbitrary density function as $h^{(ij)}(l_k, l_{k-1})$. For convenience, we use a two-dimensional Gaussian distribution normalized in the range of $[l_{\min}, l_{\max}]$, as shown in the top right in Figure 2.3; thus, the parameter set of the function $h^{(ij)}(l_k, l_{k-1})$ becomes $\{h_m^{(i)}, h_v^{(i)}, h_c^{(ij)}\}$, where $h_m^{(i)}$ and $h_v^{(i)}$ denotes mean and variance of duration lengths in discrete state q_i, and $h_c^{(ij)}$ denotes covariance between the adjacent duration lengths in discrete-state sequences $q_i q_j$ $(i \neq j)$.

Using the above notations and assuming that the current discrete state is independent on the duration of the previous interval, we can calculate the interval transition probability as follows:

$$\begin{aligned} P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >) &= P(l_k = \tau | s_k = q_j, s_{k-1} = q_i, l_{k-1} = \tau_p) \\ &\quad \times P(s_k = q_j | s_{k-1} = q_i) \\ &= \hat{h}^{(ij)}(\tau, \tau_p) A_{ij}, \end{aligned} \tag{2.18}$$

where $\hat{h}^{(ij)}(l_k, l_{k-1})$ is a one-dimensional Gaussian distribution:

$$\begin{aligned} \hat{h}^{(ij)}(l_k, l_{k-1}) &\triangleq P(l_k | s_k = q_j, s_{k-1} = q_i, l_{k-1}) \\ &= \frac{h^{(ij)}(l_k, l_{k-1})}{\sum_{l_{k-1}} h^{(ij)}(l_k, l_{k-1})}, \end{aligned}$$

and A_{ij} is a *discrete-state transition probability*:

$$A_{ij} \triangleq P(s_k = q_j | s_{k-1} = q_i), \tag{2.19}$$

where $i \neq j$.

Note that, in the conventional discrete state models such as HMMs and SLDSs, the diagonal elements of the matrix $[A_{ij}]$ define the probabilities of the self loops. In the interval system, on the other hand, the diagonal elements are separated from the matrix and defined as duration-length distributions. As a result, the balance between diagonal and non-diagonal elements varies due to the current state duration.

2.3.2 Probabilistic Inference of the Intervals

Now, we describe how to inference the intervals (i.e., states and duration lengths) based on the interval-based state transition. Unlike conventional (i.e., frame-wise) discrete state inference, we have to consider two different temporal representation: the order of intervals k and time t at the same time. As shown in the following paragraphs, the probabilistic inference becomes recursive calculation based on time t.

Let us consider how to calculate all the probabilities of every possible interval when the parameters of the automaton are given. Because we assumed the first-order Markov property, the probability of the interval $< q_j, \tau >$ can be cal-

culated using all the possible intervals that have occurred just before the interval $< q_j, \tau >$:

$$\begin{aligned} & P(I_k =< q_j, \tau >) \\ = & \sum_{<q_i,\tau_p> \in \mathcal{Q}\times\mathcal{T}} P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >) P(I_{k-1} =< q_i, \tau_p >), \end{aligned} \tag{2.20}$$

where the summation for $< q_i, \tau_p >$ does not include q_j (i.e., there are no self loops such as $q_j \rightarrow q_j$).

Although this equation gives us general idea of the inference algorithm, this recursion is based on the temporal order of intervals k, and we need to map all the intervals to the physical time line. Here, we introduce a variable f_t that takes one of the binary values {0, 1}. If $f_t = 1$, it denotes the interval "finishes" at time t, which follows Murphy's notation that is used in a research note about segment models [Mur02]. Using this notation, we can rewrite Equation (2.20) as the following time-based equation:

$$\begin{aligned} & P(s_t = q_j, l_t = \tau, f_t = 1) \\ = & \sum_{i(i\neq j)} \sum_{\tau_p} \big\{ P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \\ & \qquad \times P(s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \big\}, \end{aligned} \tag{2.21}$$

where

$$\begin{aligned} & P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \\ = & P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >). \end{aligned} \tag{2.22}$$

If we assume a finite length for the generated sequence, let the length be T, the probability $\sum_j \sum_\tau P(s_t = q_j, l_t = \tau, f_t = 1) = P(f_t = 1)$ becomes 1 at the final time point $t = T$.

Approximation (assuming the independence of the previous duration length)

If we can approximate that the interval probability is independent of the duration length of the previous interval, we can use the following equation as substitute

for Equation (2.22):

$$\begin{aligned} & P(s_t = q_j, f_t = 1) = \sum_{\tau} P(s_t = q_j, l_t = \tau, f_t = 1) \\ = & \sum_{\tau} \sum_{i(i \neq j)} P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, f_{t-\tau} = 1) P(s_{t-\tau} = q_i, f_{t-\tau} = 1). \end{aligned} \tag{2.23}$$

Example (Interval lattice)

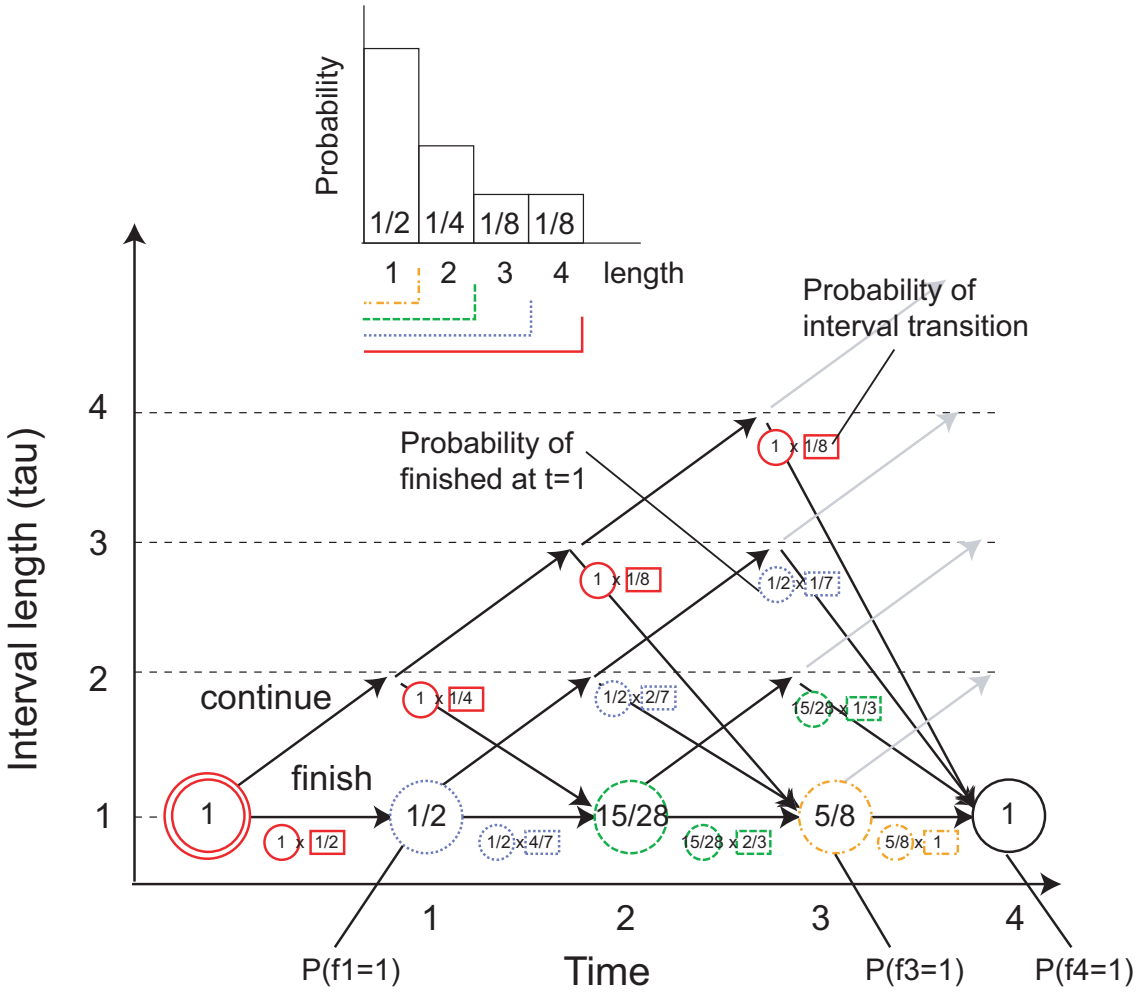
Figure 2.4 shows an intuitive example of this recursive calculation when the total sequence length is four. In this example, the number of states is two (e.g., q_1 and q_2), we can therefore omit state transition probability $A_{12} = A_{21} = 1$. The initial state is q_1 or q_2, and the succeeding intervals are labeled by these two states by turns. To simplify the example, we assume the following duration-length distribution (not a Gaussian):

$$\begin{aligned} \hat{h}^{(ij)}(l_k = 1, l_{k-1}) = 1/2, \quad \hat{h}^{(ij)}(l_k = 2, l_{k-1}) = 1/4, \\ \hat{h}^{(ij)}(l_k = 3, l_{k-1}) = 1/8, \quad \hat{h}^{(ij)}(l_k = 4, l_{k-1}) = 1/8, \end{aligned}$$

where the distribution is independent of l_{k-1}. Therefore, this is an example of Equation (2.23).

The arrow that has beginning point at (time t, length τ) represents the state q_1 (or q_2) continues or finishes at time t with duration length τ. Four circle nodes at the bottom (except the leftmost node) represent $P(f_t = 1) = \sum_{j=1}^{2} \sum_{\tau=1}^{4} P(s_t = q_j, l_t = \tau, f_t = 1)(t = 1, 2, 3, 4)$, and one of the path from the leftmost circle to the rightmost circle determines an interval sequence (partitioned sequence). For example, the bottom path represents that all the comprising intervals have length 1, and the indices of the intervals therefore correspond to time points (i.e., $k = t$).

Each node has multiple input from other nodes, which corresponds to the summation over duration length τ in Equation (2.23). We see that $P(f_4 = 1)$ takes 1 because all the possible interval sequences finish at the final time point $t = 4$.



Examples

Interval sequence that corresponds to the bottom path

Interval sequence that corresponds to the top path

Figure 2.4: Interval lattice and examples of the generated interval sequences

2.4 Inference of the Interval-Based Hybrid Dynamical System

This section describes a probabilistic inference method that searches the optimal interval sequence to represents an input multivariate sequence. The method assumes that the interval system have been trained beforehand.

As we will see in the following paragraphs, the inference method recursively finds the intervals that provide the highest likelihood score with respect to the input. This is done by generating all the possible intervals and by selecting the optimal interval sets at every time t based on a dynamic programming technique. As a result, the input sequence is partitioned and labeled by discrete states that determine the most likely dynamical system to represent a multivariate sequence in each interval. In other words, the inference is a model fitting process that fits intervals to the given multivariate sequences.

The likelihood of the trained model with respect to the input sequence is obtained simultaneously as the score of the fitting precision. This inference process is required in the EM algorithm of the interval system identification as we will see in Section 3.4.

2.4.1 Forward Algorithm

The most naive method for the interval-sequence search is that first calculates the likelihood scores of the model from all the possible interval sequences independently, and then finds the best interval sequence that provides the largest likelihood. However, the computational cost becomes order of $O(N^T)$ in this case. To avoid unnecessary calculation, we exploit a recursive calculation similar to HMMs.

Let us first consider the forward algorithm of the interval system. Suppose that input multivariate data y have been observed from time 1 to t, and the interval $I_k =< q_j, \tau >$ ends at time t. Considering all the possible intervals that have occurred just before the interval I_k, we can decompose the joint probability

$P(I_k =< q_j, \tau >, y_1^t)$ as the following recursive equation:

$$\begin{aligned}
&P(I_k =< q_j, \tau >, y_1^{e_k}) \\
&= P(y_{e_{k-1}+1}^{e_k} | I_k =< q_j, \tau >) \\
&\times \sum_{<q_i, \tau_p> \in \mathcal{Q} \times \mathcal{T}} \left\{ P(I_k =< q_j, \tau > | I_{k-1} =< q_i, \tau_p >) P(I_{k-1} =< q_i, \tau_p >, y_1^{e_{k-1}}) \right\}
\end{aligned}$$

where e_k and e_{k-1} are the ending points of interval I_k and I_{k-1}, respectively, and $e_k = t$.

As we described in Subsection 2.3.2, this recursion is based on the interval order, and is difficult to cope with observation data that comes every time point t. We therefore rewrite this equation as the following time-based recursion, which is similar to Equation (2.22) rewritten from Equation (2.22):

$$\begin{aligned}
& P(s_t = q_j, l_t = \tau, f_t = 1, y_1^t) \\
= \; & P(y_{t-\tau+1}^t | s_t = q_j, l_t = \tau, f_t = 1) \\
& \times \sum_{i(i \neq j)} \sum_{\tau_p} \big\{ P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1) \\
& \quad \times P(s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1, y_1^{t-\tau}) \big\}
\end{aligned} \tag{2.24}$$

To initialize the forward algorithm, we have to calculate the probability of the interval that appears at the first time in each interval sequence.

$$\begin{aligned}
& P(s_t = q_j, l_t = \tau, f_t = 1, f_1^{t-1} = 0, y_1^t) \\
= \; & P(y_1^t | s_t = q_j, l_t = t) P(s_t = q_j) P(l_t = t | s_t = q_j)
\end{aligned} \tag{2.25}$$

The duration length probability $P(l_t | s_t = q_j)$ can be calculated by the following equation:

$$\begin{aligned}
\hat{h}^{(j)}(l_t) \triangleq P(l_t = \tau | s_t = q_j) &= \sum_i \sum_{l_{k-1}} P(l_t, l_{k-1} | s_k = q_j, s_{k-1} = q_i) \\
&= \sum_i \sum_{l_{k-1}} h^{(ij)}(l_k, l_{k-1}).
\end{aligned} \tag{2.26}$$

Here, we show the overall forward algorithm in Algorithm 1, where we use the following notation:

$$\alpha_t(i, \tau) \triangleq P(s_t = q_i, l_t = \tau, f_t = 1, y_1^t).$$

We also use the notation $d^{(i)}_{[t-\tau+1,t]}$ for $P(y^t_{t-\tau+1}|s_t = q_i, l_t = \tau, f_t = 1)$, which we defined in Equation (2.15). As we have shown in Section 2.2, $d^{(i)}_{[t-\tau+1,t]}$ denotes the likelihood of the linear dynamical system D_i with respect to the observation from $t-\tau+1$ to t, and can be calculated by using parameters of dynamical system D_i. On the other hand, the interval transition probability $P(s_t = q_j, l_t = \tau, f_t = 1|s_{t-\tau} = q_i, l_{t-\tau} = \tau_p, f_{t-\tau} = 1)$ can be calculated by Equation (2.18) in Section 2.3.

Algorithm 1 Forward Algorithm

for $t \leftarrow 1$ to $l_{\min} - 1$ **do**
  Fill $\alpha_t(i, \tau)$ by 0
end for
for $t \leftarrow l_{\min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    for $\tau \leftarrow l_{\min}$ to $\tau_{\max}$ **do**
      $\tau_{p\max} \leftarrow \min(l_{\max}, t - \tau)$
      $\alpha_t(j, \tau) \leftarrow d^{(j)}_{[t-\tau+1,t]} \sum_i \sum_{\tau_p=l_{\min}}^{\tau_{p\max}} A_{ij}\hat{h}^{(ij)}(\tau, \tau_p)\alpha_{t-\tau}(i, \tau_p)$ # Eq. (2.24)
    end for
    if $t \leq l_{\max}$ **then**
      $\alpha_t(j, t) \leftarrow d^{(j)}_{[1,t]}\pi(j)\hat{h}^{(j)}(t)$ # Initialization (Eq. (2.25))
    end if
  end for
end for

Approximated Forward Algorithm

Assuming that the duration-length distribution is independent of the duration length of the previous interval, we can use the following recursive equation, which is deduced from Equation (2.23), on instead of Equation (2.24):

$$\begin{aligned} & P(s_t = q_j, f_t = 1, y^t_1) \\ = & \sum_\tau P(y^t_{t-\tau+1}|s_t = q_j, l_t = \tau, f_t = 1) \\ \times & \sum_{i(i\neq j)} \{P(s_t = q_j, l_t = \tau, f_t = 1|s_{t-\tau} = q_i, f_{t-\tau} = 1, y^{t-\tau}_1) \\ & \qquad \times P(s_{t-\tau} = q_i, f_{t-\tau} = 1, y^{t-\tau}_1)\} \end{aligned} \tag{2.27}$$

In precise, the above equation calculates $P(s_t = q_j, f_t = 1, y^t_1)$, where one of f_1 to f_{t-1} is 1. Therefore, we also require the probability of the interval that appears

at the first time in each interval sequence: $P(s_t = q_j, f_t = 1, f_1^{t-1} = 0, y_1^t)$. Then, we have to add to this probability to Equation (2.27), when t is not grater than the maximum interval length $l_{\max}$.

Here, we can use the following relation:

$$P(s_t = q_j, f_t = 1, f_1^{t-1} = 0, y_1^t) = P(s_t = q_j, f_t = 1, l_t = t, y_1^t),$$

because $l_t = t$ implicitly denotes that the interval does not finish from time 1 to $t-1$. Therefore, the initialization becomes exactly the same equation as Equation (2.25).

The overall forward algorithm becomes Algorithm 2, where we use the following notation:

$$\alpha_t(i) \triangleq P(s_t = q_i, f_t = 1, y_1^t).$$

Algorithm 2 Forward Algorithm (assuming the independence of previous dur.)

for $t \leftarrow 1$ to $l_{\min} - 1$ **do**
  Fill $\alpha_t(i)$ by 0
end for
for $t \leftarrow l_{\min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    $\alpha_t(j) \leftarrow \sum_{\tau=l_{\min}}^{\tau_{\max}} d^{(j)}_{[t-\tau+1,t]} \sum_i A_{ij} \hat{h}^{(j)}(\tau) \alpha_{t-\tau}(i)$ # Equation (2.27)
    if $t \leq l_{\max}$ **then**
      $\alpha_t(j) \leftarrow \alpha_t(j) + d^{(j)}_{[1,t]} \pi(j) \hat{h}^{(j)}(t)$
    end if
  end for
end for

2.4.2 Viterbi Algorithm

The forward algorithm often causes numerical underflow when the length of the input becomes longer. That is, at each step of the recursion in Equation (2.24), the probability such as $\alpha_t(j, \tau)$ becomes smaller than the previous probability, and finally the probabilities get below the machine epsilon.

In the following paragraphs, we describe the Viterbi algorithm in which we can take logarithm of the formulation to avoid the numerical underflow problem. Although this algorithm returns only the most likely interval sequence, it

is enough information for some applications. For instance, we use this Viterbi algorithm to train the interval system, as we see in Chapter 3.

This algorithm is based on a dynamic programming method similar to the Viterbi algorithm of HMMs without that it requires the consideration of duration lengths. Suppose that the algorithm have found the optimum interval sequence from time 1 to $t-\tau-\tau_p$ that maximize probability $P(s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1, y_1^{t-\tau})$. Let use the following notation for this maximized probability:

$$\delta_{t-\tau}(i,\tau_p) \triangleq \max_{s_1^{t-\tau-\tau_p}} P(s_1^{t-\tau-\tau_p}, s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1, y_1^{t-\tau}). \tag{2.28}$$

Then, we can calculate probability $\delta_t(j,\tau)$ based on the following equation, which can be deduced from Equation (2.24).

$$\begin{aligned} & \delta_t(j,\tau) = \max_{s_1^{t-\tau}} P(s_1^{t-\tau}, s_t=q_j, l_t=\tau, f_t=1, y_1^t) \\ = \;& P(y_{t-\tau+1}^t | s_t=q_j, l_t=\tau, f_t=1) \\ & \times \max_{i(i\neq j),\tau_p} \{P(s_t=q_j, l_t=\tau, f_t=1 | s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1)\delta_{t-\tau}(i,\tau_p)\}. \end{aligned} \tag{2.29}$$

Since this recursive calculation gives us the maximum probabilities of all the time points with possible states and duration lengths, we can get the most likely interval sequence using traceback of arguments (i.e., states and duration lengths) that gives the maximized probabilities at each recursion step. We therefore need to record the following arguments together with $\delta_t(j,\tau)$ at the maximization of Equation (2.29):

$$\begin{aligned} & (s_t^*(j,\tau), l_t^*(j,\tau)) \\ = \;& \arg\max_{i(i\neq j),\tau_p} \{P(s_t=q_j, l_t=\tau, f_t=1 | s_{t-\tau}=q_i, l_{t-\tau}=\tau_p, f_{t-\tau}=1)\delta_{t-\tau}(i,\tau_p)\}. \end{aligned}$$

Traceback

Now, we describe a traceback algorithm for searching the most likely interval sequence. Using Equation (2.29) recursively, we get the maximized probability at final time point ($t=T$)

$$\max_{s_1^{T-\tau}} P(s_1^{T-\tau}, s_T=q_j, l_T=\tau, f_T=1, y_1^t) = \delta_T(j,\tau). \tag{2.30}$$

Here, we can find the state and duration length of the most likely interval sequence by finding the maximum probability of $\delta_T(j,\tau)$:

$$(cs_K, cl_K) = \arg\max_{j,\tau} \delta_T(j,\tau). \tag{2.31}$$

Then, we the most likely interval sequence by calculating the following recursion:

$$\begin{aligned} cs_{k-1} &= s^*_{ce_k}(cs_k, cl_k), \quad cl_{k-1} = l^*_{ce_k}(cs_k, cl_k) \\ ce_{k-1} &= ce_k - cl_k, \end{aligned}$$

where ce_k is the ending point of interval k, which is initialized by $ce_K = T$, cs_k and cl_k is the state and duration length of interval k. Finally, we get intervals $< cs_k, cl_k > (k \leq K)$ that comprise the most likelihood interval sequence. Note that the total number of intervals (K) is known. In the actual algorithm, we therefore use very large integer for K, or use increment of the indices on behalf decrementing index k.

Taking logarithm of all the equations, we get the overall algorithm shown in Algorithm 3.

Approximated Viterbi Algorithm

Assuming that the duration-length distribution is independent of the duration length of the previous interval, we can use the following recursive equation, which is deduced from Equation (2.27), instead of Equation (2.29):

$$\begin{aligned} \delta_t(j) &\triangleq \max_{s_1^{t-1}} P(s_1^{t-1}, s_t = q_j, f_t = 1, y_1^t) \\ &= \max_{\tau} [P(y_{t-\tau+1}^t | s_t = q_j, l_t = \tau, f_t = 1) \\ &\quad \times \max_{i(i \neq j)} \{P(s_t = q_j, l_t = \tau, f_t = 1 | s_{t-\tau} = q_i, f_{t-\tau} = 1) \max_{u_1^{t-\tau-1}} \delta_{t-\tau}(i)\}] \end{aligned}$$

The overall algorithm is shown in Algorithm 4. As we described in the previous paragraph, the most likely interval sequence is given by the final traceback step. The difference from Algorithm 3 is that this approximated model does not require to recording maximized probabilities for the previous duration length.

Algorithm 3 Viterbi Algorithm

for $t \leftarrow 1$ to $l_{\min} - 1$ **do**
  Fill $\log \delta_t(i, \tau)$ by $-\infty$
end for
for $t \leftarrow l_{\min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    for $\tau \leftarrow l_{\min}$ to $\tau_{\max}$ **do**
      $\tau_{p\,\max} \leftarrow \min(l_{\max}, t - \tau)$
      $\log \delta_t(j, \tau) \leftarrow \log d^{(j)}_{[t-\tau+1,t]} + \max_{i,\tau_p} \left[\log A_{ij} + \log \hat{h}^{(ij)}(\tau, \tau_p) + \log \delta_{t-\tau}(i, \tau_p)\right]$
      $(s^*_t(j, \tau), l^*_t(j, \tau)) \leftarrow \arg\max_{i,\tau_p} \left[\log A_{ij} + \log \hat{h}^{(ij)}(\tau, \tau_p) + \log \delta_{t-\tau}(i, \tau_p)\right]$
      # $i \in \{1, ..., N\}, \tau_p \in \{l_{\min}, ..., \tau_{p\,\max}\}$
    end for
    if $t \leq l_{\max}$ **then**
      $\log \delta_t(j, t) \leftarrow \log d^{(j)}_{[1,t]} + \log \pi(j) + \log \hat{h}^{(j)}(t)$
      $(s^*_t(j, t), l^*_t(j, t)) \leftarrow (0, 0)$
    end if
  end for
end for
# Traceback
$ce_K \leftarrow T$
$(cs_K, cl_K) \leftarrow \arg\max_{j,\tau} \log \delta_T(j, \tau)$
while $cs_k > 0$ **do**
  $cs_{k-1} \leftarrow s^*_{ce_k}(cs_k, cl_k), \quad cl_{k-1} \leftarrow l^*_{ce_k}(cs_k, cl_k)$
  $ce_{k-1} \leftarrow ce_k - cl_k$
  $k \leftarrow k - 1$
end while

Algorithm 4 Viterbi Algorithm (assuming the independence of previous dur.)

for $t \leftarrow 1$ to $l_{\min} - 1$ **do**
  Fill $\log \delta_t(i)$ by $-\infty$
end for
for $t \leftarrow l_{\min}$ to T **do**
  $\tau_{\max} \leftarrow \min(l_{\max}, t - l_{\min})$
  for $j \leftarrow 1$ to N **do**
    $\log \delta_t(j) \leftarrow \max_{i,\tau} \left[\log d^{(j)}_{[t-\tau+1,t]} + \log A_{ij} + \log \hat{h}^{(j)}(\tau) + \log \delta_{t-\tau}(i)\right]$
    $(s^*_t(j), l^*_t(j)) \leftarrow \arg\max_{i,\tau} \left[\log d^{(j)}_{[t-\tau+1,t]} + \log A_{ij} + \log \hat{h}^{(j)}(\tau) + \log \delta_{t-\tau}(i)\right]$
    $\# \, i = (1, ..., N), \tau = (l_{\min}, ..., \tau_{\max})$
    if $t \leq l_{\max}$ **then**
      $\log \delta'_t(j) \leftarrow \log d^{(j)}_{[1,t]} + \log \pi(j) + \log \hat{h}^{(j)}(t)$
      if $\log \delta_t(j) < \log \delta'_t(j)$ **then**
        $\log \delta_t(j) \leftarrow \log \delta'_t(j), \quad s^{(*)}_t(j) \leftarrow 0, \quad l^{(*)}_t(j) \leftarrow t$
      end if
    end if
  end for
end for
# Traceback
$ce_K \leftarrow T$
$cs_K \leftarrow \arg\max_j \log \delta_T(j)$
$k \leftarrow K$
while $cs_k > 0$ **do**
  $cl_k \leftarrow l^*_{ce_k}(cs_k), \quad cs_{k-1} \leftarrow s^*_{ce_k}(cs_k)$
  $ce_{k-1} \leftarrow ce_k - cl_k$
  $k \leftarrow k - 1$
end while

2.4.3 Calculation Cost

The forward and Viterbi algorithms require searching all the possible intervals for all the current intervals at every sampled time t. Therefore, the computational cost becomes $O((NL)^2T)$ $(L = l_{\max} - l_{\min} + 1)$, which requires a greater cost compared to HMMs, which requires only $O(N^2T)$. However, the cost can be reduced drastically in the case that the range L is small.

Approximated forward and Viterbi algorithms, which assume the independence of the previous interval duration length, requires computational cost $O(N^2LT)$.

In addition, if we calculate likelihood of dynamical systems (i.e., Equation (2.15)) before the recursive calculation, the actual cost can be also reduced.

2.5 Verification of the Inference Algorithms

In this section, we verify the capability of the interval system and the validity of the proposed inference algorithms (i.e., the forward and Viterbi algorithms).

First, the parameters of an interval system were given manually, and interval sequences were generated by the system for test data. Then, the data was used as input of the forward and Viterbi algorithms.

To concentrate on the interval-based state transition in the interval system, we set the observation matrix $H = I$ (unit matrix) and the observation noise covariance $R = O$ (zero matrix). The number of discrete states was $N = 3$ and the dimensionality of the internal state space was $n = 3$. The range of the interval duration was $[l_{\min} = 1, l_{\max} = 30]$. We set the probability matrix of the discrete-state transition as $A_{12} = A_{23} = A_{31} = 1$ and 0 for all the other elements to generate loops such as $q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_1$. The initial distribution of the discrete state was 1 for q_1 and 0 for the remaining states q_2 and q_3.

The two dimensional distribution of the duration length $h^{(ij)}(l_k, l_{k-1})$ had {mean $(h_m^{(i)})$, variance $(h_v^{(i)})$} of {6, 5}, {12, 30} and {16, 50} for q_1, q_2, and q_3, respectively. The covariance $(h_c^{(ij)})$ between the pairs of $\{q_1, q_2\}, \{q_2, q_3\}$ and $\{q_3, q_1\}$ was $12, 35$ and 15, respectively. These covariances were designed to generate interval sequences that the duration lengths of the intervals were monotonically increased in the sequence of q_1, q_2, and q_3.

The transition matrices, bias vectors, and initial mean vectors of the dynamical

systems were as follows:

$$F^{(1)} = \begin{bmatrix} 0.6 & -0.1 \\ -0.1 & 0.2 \end{bmatrix}, g^{(1)} = \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix}, x_{\text{init}}^{(1)} = \begin{bmatrix} 1.0 \\ -1.0 \end{bmatrix}$$

$$F^{(2)} = \begin{bmatrix} 0.3 & 0.0 \\ 0.0 & 0.6 \end{bmatrix}, g^{(2)} = \begin{bmatrix} -0.7 \\ 0.0 \end{bmatrix}, x_{\text{init}}^{(2)} = \begin{bmatrix} 0.3 \\ 1.0 \end{bmatrix}$$

$$F^{(3)} = \begin{bmatrix} 0.5 & 0.1 \\ -0.1 & 0.3 \end{bmatrix}, g^{(3)} = \begin{bmatrix} 0.6 \\ -0.6 \end{bmatrix}, x_{\text{init}}^{(3)} = \begin{bmatrix} -1.0 \\ 0.0 \end{bmatrix}$$

The process noise covariance matrices $Q^{(i)}$ $(i = 1, 2, 3)$ and the covariance matrices $V_{\text{init}}^{(i)}$ $(i = 1, 2, 3)$ of the initial state distribution were set to zero in the generation step, and was set to $0.001I$ (where I is a unit matrix) in the fitting step of intervals.

Figure 2.5(a) shows a generated interval sequence from the finite state automaton. The length of the sequence was $T = 100$. We see that the duration of the intervals increases monotonically in the sequence of q_1, q_2, and q_3 because we set positive correlation between the adjacent intervals (i.e., q_1 to q_2 and q_2 to q_3).

In parallel of this discrete-state transition, each dynamical system was activated by the discrete state, and generates a sequence of signal. Figure 2.5(b) shows a generated observation sequence. In this experiment, this sequence corresponds to the generated internal state sequence as a result of observation parameters $H = I$ and $R = O$. We see that the time-varying pattern of the observation changes based on the transition of the intervals.

To verify the algorithms of forward and Viterbi inference, we input a generated sequence shown in Figure 2.5(b) to the original interval system (i.e., the system that generated the input sequence). Figure 2.5 (c) shows the result of the forward inference. Each line denotes the following probabilities of the discrete states under the condition of observations from time 1 to t:

$$P(s_t = q_j | y_1^t) = \frac{\sum_\tau P(I_t =< q_j, \tau >, y_1^t)}{P(y_1^t)}$$

Figure 2.5 (d) shows the result of backtracked intervals after the Viterbi algorithm. We see that both of the forward and Viterbi inferences have found optimal interval sequences that are consistent with the original interval sequence in Figure 2.5 (a).

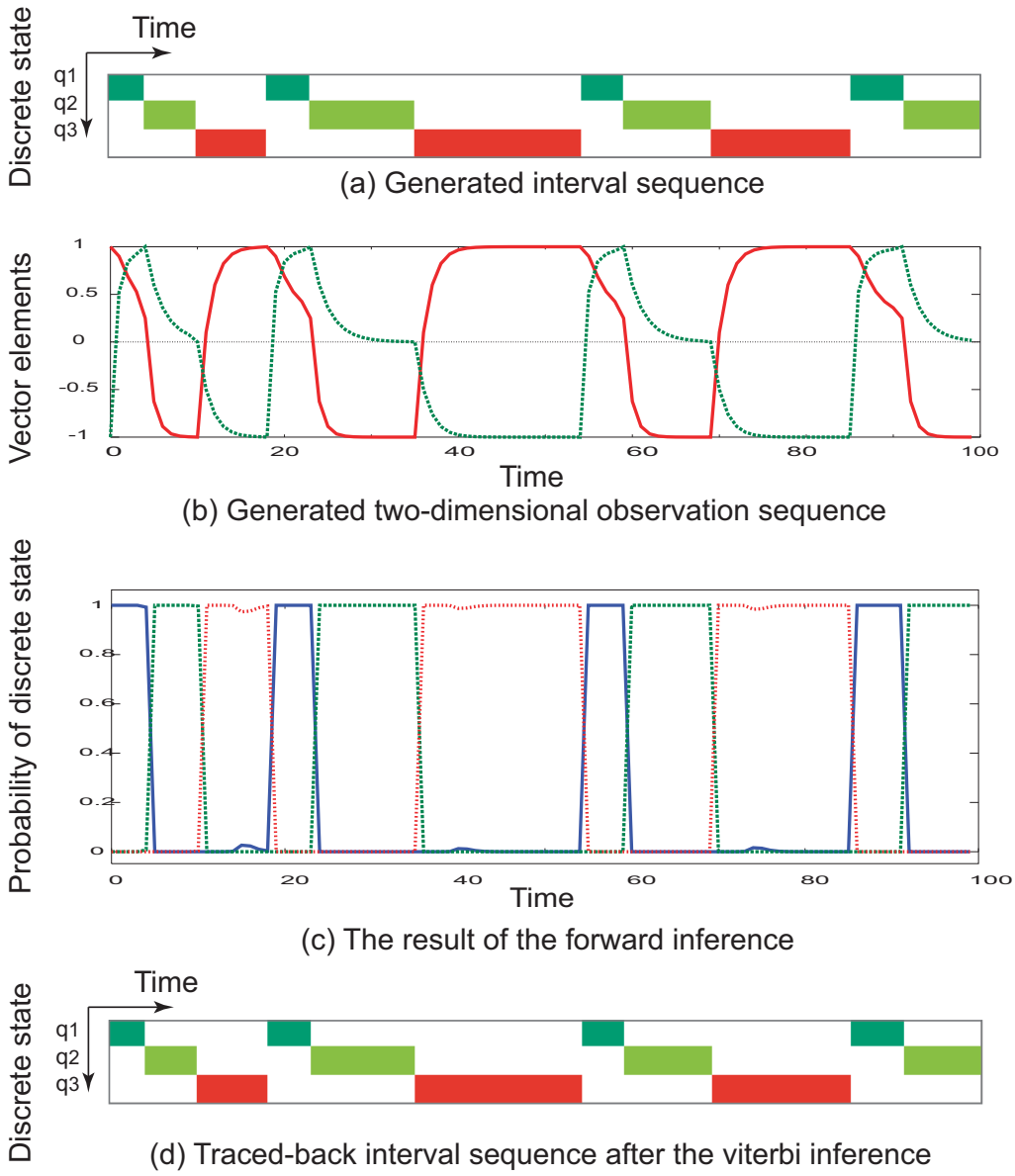


Figure 2.5: Verification of the forward and Viterbi algorithms. (a) A generated interval sequence by a finite state automaton. (b) A generated observation sequence by three dynamical systems using interval sequence of (a). The solid line denotes the first element; the dashed line denotes the second element. (c) The result of the forward inference using (b) as input. Each line represents probability $P(s_t = q_j | y_1^t)$ $(j = 1, 2, 3)$. (d) The result of the backtracked interval sequence after the Viterbi inference using (b) as input. We can see that the original interval sequence is obtained correctly.

To see the influence of parameters in the duration-length distributions, we generated interval sequences with zero covariance between adjacent intervals. Figure 2.6 shows an example of the generated sequence. We see that the first state sequence of q_1, q_2, and q_3 has non-monotonic changes of duration (i.e., q_3 is shorter than q_2, while q_2 is longer than q_1), which implies that the each discrete state duration is decided independently. Consequently, the correlation modeling with covariances between the adjacent intervals is necessary to represent rhythms and tempo as patterns of duration lengths.

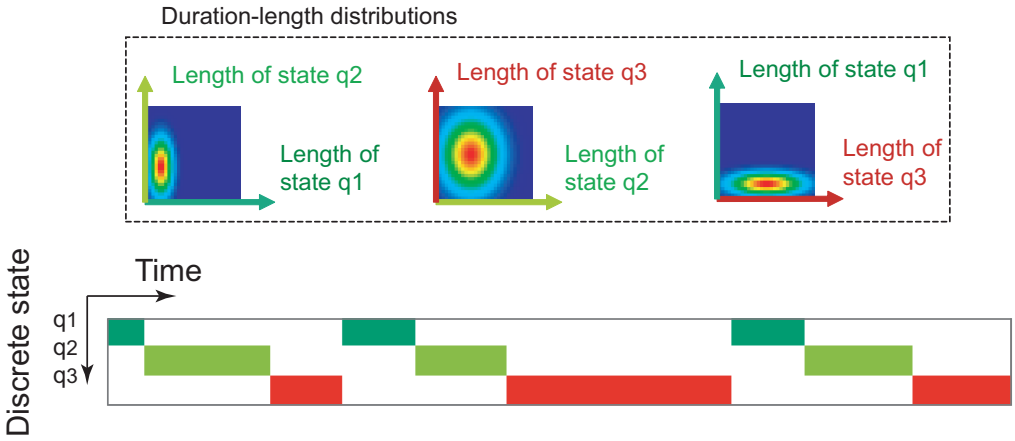
2.6 Discussion

In this chapter, we proposed a computational model, which we refer to as the interval-based hybrid dynamical system, that comprises a finite state automaton (discrete-event system) and multiple linear dynamical systems. Each linear dynamical system represents a dynamic primitive that corresponds to a discrete state of the automaton.

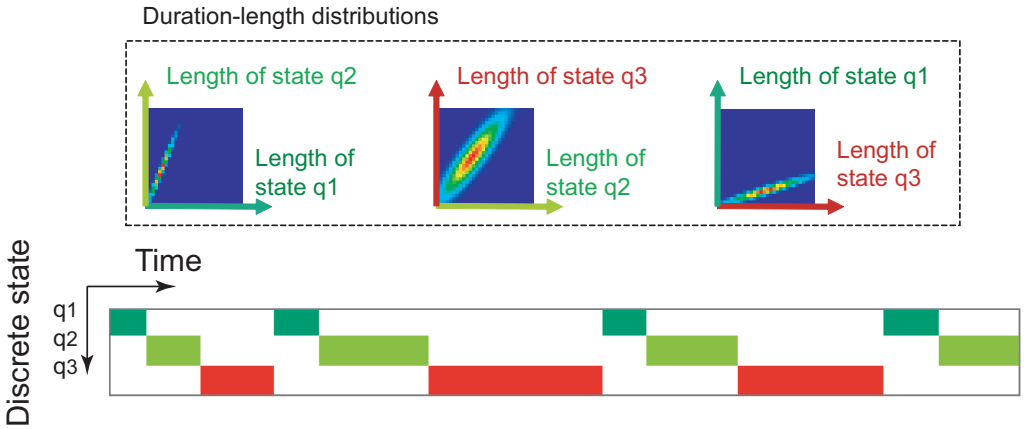
The key idea of integrating two systems is the use of temporal intervals in which each constituting linear dynamical system is activated. As a result, the temporal order of discrete state is mapped to the physical-time domain based on the duration lengths of intervals. Due to the duration-length modeling of discrete states, we successfully represent temporal patterns of discrete event such as rhythms based on the correlation of adjacent interval lengths.

In this chapter, we assumed that all the parameters of the interval system are given. We, however, require all the parameters to be estimated from training data in most of real problems. In Chapter 3, we introduce a learning method for the interval system.

Modeling relations among concurrent multiple streams (e.g., temporal relations of motions among facial parts) and relations among multimodal data (e.g., lip motion and speech data) is one of the important objectives for the interval-based representation. We try to analyze temporal structures among motion of facial parts in Chapter 4. A general modeling method of such a temporal structures, which we refer to as "timing structures", will be required to represent concurrent events. We will introduce a timing structure model in Chapter 5 using relation between intervals.



(a) Generated interval sequence using zero covariance distributions



(b) Generated interval sequence using non-zero covariance distributions

Figure 2.6: A generated interval sequence using zero covariance duration-length distributions (shown in (a)) for comparison to the interval sequence generated by using non-zero covariance distributions (shown in (b), which corresponds to Figure 2.5 (a)).