Influence Modeling of Complex Stochastic Processes

by

Wen Dong

M.E., Computer Science (2004)
The University of San Francisco

Submitted to the Department of Media Arts and Sciences
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Media Arts and Sciences

at the

Massachusetts Institute of Technology

September 2006

Signature of Author.....
Department of Media Arts and Sciences
July 20, 2006

Certified by.....
Alex (Sandy) Pentland
Toshiba Professor of Media Arts and Sciences
Thesis Supervisor

Accepted by .....
Andrew B. Lippman
Chair, Departmental Committee on Graduate Students

Influence Modeling of Complex Stochastic Processes

by

Wen Dong

Submitted to the Department of Media Arts and Sciences
on July 20, 2006, in partial fulfillment of the
requirements for the degree of Master of Science in
Media Arts of Sciences

ABSTRACT

A complex stochastic process involving human behaviors or human group behaviors is computationally hard to model with a hidden Markov process. This is because the state space of such behaviors is often a Cartesian product of a large number of constituent probability spaces, and is exponentially large. A sample for those stochastic processes is normally composed of a large collection of heterogeneous constituent samples. How to combine those heterogeneous constituent samples in a consistent and stable way is another difficulty for the hidden Markov process modeling. A latent structure influence process models human behaviors and human group behaviors by emulating the work of a team of experts. In such a team, each expert concentrates on one constituent probability space, investigates one type of constituent samples, and/or employ one type of technique. An expert improves his work by considering the *results* from the other experts, instead of the r*aw data* for them. Compared with the hidden Markov process, the latent structure influence process is more expressive, more stable to outliers, and less likely to overfit. It can be used to study the interaction of over 100 persons and get good results.

This thesis is organized in the following way. Chapter 0 reviews the notation and the background concepts necessary to develop this thesis. Chapter 1 describes the intuition behind the latent structure influence process and the situations where it outperforms the other dynamic models. In Chapter 2, we give inference algorithms based on two different interpretations of the influence model. Chapter 3 applies the influence algorithms to various toy data sets and real-world data sets. We hope our demonstrations of the influence modeling could serve as templates for the readers to develop other applications. In Chapter 4, we conclude with the rationale and other considerations for influence modeling.

Thesis Supervisor: Alex (Sandy) Pentland
Title: Toshiba Professor of Media Arts and Sciences

Influence Modeling of Complex Stochastic Processes

by

Wen Dong

Submitted to the Department of Media Arts and Sciences
on July 20, 2006, in partial fulfillment of the
requirements for the degree of Master of Science in
Media Arts of Sciences

Alex (Sandy) Pentland
Toshiba Professor of Media Arts and Sciences
Director, Human Dynamics
Thesis Supervisor

Joe Paradiso
Sony Corporation Career Development Professor of Media Arts and Sciences
Co-Director, Things that Think
Thesis Reader

V. Michael Bove, Jr.
Principal Research Scientist
Director, Consumer Electronics Laboratory
Thesis Reader

Acknowledgements

My gratitude should first go to Professor Pentland. Two years ago, Professor Pentland and his postdoc Martin C. Martin acquainted me with the (fully observable) influence model, and I was happy to realize that the research on the influence model could be my first step towards my goal of understanding the social intelligence. Professor Pentland also provided me many chances to meet the leading figures, to understand the important intuitions, and to acquire interesting data sets, in the field of machine learning and wearable computing. It is especially supportive and comfortable working in his group.

My grateful thanks also go to Professor Paradiso and Professor Bove. I had pleasant and fruitful cooperation in the UbER-badge project with Professor Paradiso's students: Mat Laibowitz and Ryan Aylward. I am also inspired by the suggestions and expectations of Professor Paradiso himself. I took the course *Signals, Systems and Information for Media Technology (MAS510/511)* offered by Professor Bove. Professor Bove and his teaching assistant Quinn Smithwick provided me with great experiences in attending this course.

I would like to thank Professor Verghese for offering to discuss with me on the latent structure influence modeling. I would also like to thank my peers, who have been generously sharing their data sets with me, and asking for my opinions on the mathematical modeling of the data sets. Several of my peers are (arranged in the reverse chronological order of when I last cooperated with them): Daniel Olguin, Jonathan Gipps, Anmol Maddan, Ron Caneel, Nathan Eagle, and Mike Sung.

Linda Peterson and Pat Solakoff have provided countless help in the administrative issues, and in making these things simple and smooth. Without their kind help, many things would go much tougher.

Contents

List of Algorithms

List of Figures

Chapter 0

Notation and Preliminaries

In this chapter, we collect together the concepts involved in this thesis, as well as the notation. This chapter proceeds in the following order. We first define probability space, random variable, random vector, and stochastic process. Then we review Markov chain, hidden Markov process, and the inference algorithms of a hidden Markov process. This is followed by a discussion of the computational time complexity of the coupled hidden Markov process, and other complex latent variable dynamic processes. We will describe briefly how the latent structure influence model copes with this time complexity issue by mapping the original large number of "concepts" into only a few concepts while preserving most information.

A **probability space** (Ω, F, P) is a measure space with total measure one ($P(\Omega) = 1$). The first term of this 3-tuple, Ω, is a nonempty set, whose elements are called the **outcomes (or states) of nature**. The second term of this 3-tuple, F, is a subset of the power set of Ω ($F \subseteq 2^{\Omega}$) and forms a σ-algebra. The set F is called a set of **events** (σ-algebra over Ω). The 2-tuple (Ω, F) forms a measurable space. The third term of this 3-tuple, $P : F \to [0, 1]$, is a **probability measure** of the measurable space (Ω, F).

A **random variable** X is a measurable function from a probability space $(\Omega, \Pr)$ to some measurable space R, normally a subset of real numbers or integers with Borel σ-algebra ($X : \Omega \to R$). A **function of a random variable** X is a measurable function $f : X \to Y$. Since the composition of measurable functions is a measurable function, a function of a random variable is another random variable. A random variable can be characterized by its **cumulative distribution function** $F_X(x) = P(X \leq x)$, or its **probability density function** (or **pdf**) $p(x) = \frac{\partial}{\partial x} F_X(x)$. A discrete random variable can also be characterized by its **probability mass function** (or **pmf**) $f_X(x) = P(X = x)$. The cumulative distribution function, probability density function, and the probability mass function are induced from the probability measure $\Pr$ of the original probability space $(\Omega, \Pr)$.

A **random vector** (or a **multivariate random variable**) is a vector of scalar random variables $\vec{X} = (X_1, X_2, \cdots X_n)$.

Example 1. *Let us consider the outcome of one flip of a coin. The outcomes are either heads or tails ($\Omega = \{H, T\}$). The set of events are (1) neither heads nor tails, (2) heads, (3) tails, and (4) heads or tails ($F = \{\{\}, \{H\}, \{T\}, \{H, T\}\}$). The tuple $\{\Omega, F\}$ forms a measurable space, since (1) the empty set $\{\}$ is in F, (2) for any set E in F, its complement $F \backslash E$ is also in F, and (3) the union of countably many sets in F is also in F. We can define a probability measure P as the following, (1) the probability of getting neither heads or tails is zero ($P(\{\}) = 0$), (2) the probability of getting heads is 0.5 ($P(\{H\}) = 0.5$), (3) the probability of getting tails is 0.5 ($P(\{T\}) = 0.5$, and (4) the probability of getting heads or tails is 1 ($P(\{H, T\}) = 1$). The function P is a measure since (1) the empty set has measure zero ($P(\{\}) = 0$), and (2) the measure of the union of a countable number of pairwise disjoint events is the sum of the measure of the individual events ($P(\{\} \cup \{H\}) = P(\{\}) + P(\{H\})$,$P(\{\} \cup \{T\}) = P(\{\}) + P(\{T\})$,$P(\{\} \cup \{H, T\}) = P(\{\}) + P(\{H, T\})$,$P(\{H\} \cup \{T\}) = P(\{H\}) + P(\{T\})$, and $P(\{\} \cup \{H\} \cup \{T\}) = P(\{\}) + P(\{H\}) + P(\{T\}))$.*

The function P is a probability measure since $P(\Omega) = P(\{H, T\}) = 1$.

We define a random variable X over the probability space of the outcome of one flip of a coin as the following. The random variable X can be either zero or one. The σ-algebra defined over X is $\{\{\}, \{0\}, \{1\}, \{0, 1\}\}$. The tuple $\{\{0, 1\}, \{\{\}, \{0\}, \{1\}, \{0, 1\}\}$ forms a measurable set. The function from the probability space of one flip of a coin to the measurable space $\{\{0, 1\}, \{\{\}, \{0\}, \{1\}, \{0, 1\}\}$ is defined as the following: $X(H) = 0$, $X(T) = 1$, $X(\{\}) = \{\}$,$X(\{H\}) = \{0\}$,$X(\{T\}) = \{1\}$, and $X(\{H, T\}) = \{0, 1\}$. The function X is a measurable function since the pre-image of every event in the σ-algebra of $\{0, 1\}$ is a σ-algebra of $\{H, T\}$. The random variable X can be characterized by its cumulative probability function $F_X(x) = \begin{cases} 0 & x < 0 \\ .5 & 0 \leq x < 1 \\ 1 & 1 \leq x \end{cases}$, probability density function $p_X(x) = .5\delta(x - 0) + .5\delta(x - 1)$ (where δ is the Dirac delta function), or probability mass function $f_X(0) = \begin{cases} .5 & x = 0, 1 \\ 0 & \text{otherwise} \end{cases}$. We can also define a random vector $\vec{X} = (X_1, X_2, X_3)$ of the outcome of three flips of a coin.

A **stochastic process** (or a **random process**) $\{X_i\}$ can be regarded as an indexed collection of random variables. Formally speaking, a stochastic process is a random function $X : I \to D$, which maps an **index** $i \in I$ taken from the **index set** I, to a **random variable** $X_i \in D$ defined over a probability space (Ω, P). A stochastic process is **discrete** when its index set is discrete. A stochastic process is **continuous** when its index set is continuous.

In most common applications, the index set I is a time interval or a region of space, and the corresponding index is either a time or a location. When the index set I is a time interval, the stochastic process is called a **time series**. When the index set I is a region of space, the stochastic process is called a **random field**.

A particular stochastic process can be characterized by the joint distributions of its random variables $p_{X[i_1]X[i_2]\cdots X[i_N]}(X[i_1]X[i_2]\cdots X[i_N])$ corresponding to the indices $i_1, i_2, \cdots i_N \in I$ for all natural numbers N. A time series is **stationary** if its distribution does not change with time

$$p_{X[i_1]\cdots X[i_N]}(X[i_1]\cdots X[i_N]) \quad = \quad p_{X[i_1+\tau]\cdots X[i_N+\tau]}(X[i_1+\tau]\cdots X[i_N+\tau])$$

A **discrete-time Markov process** (or **Markov chain**) is a discrete stochastic process $\{X_n\}$ with the **Markov property**, i.e., the probability distribution of the future **state** X_{n+1} is independent of the past states $X_i, i < n$ given the present state X_n,

$$P(X_{n+1}|X_0 \cdots X_t) \quad = \quad P(X_{n+1}|X_n)$$

. A Markov chain can be characterized by its initial probability distribution $P(X_1)$ and its one-step **transition probability** $P(X_{n+1}|X_n)$. The k-step transition probability of a Markov chain can be computed as

$$P(X_{n+k}|X_n) \quad = \quad \int P(X_{n+k}|X_{n+k-1}) \cdots P(X_{n+1}|X_n) dX_{n+k-1} \cdots X_{n+1}$$

. The marginal distribution $P(X_n)$ of a Markov chain can be expressed as

$$P(X_n) \quad = \quad \int P(X_1)P(X_n|X_1)dX_1$$

. If there exists a probability distribution π such that $\pi = \int P(X_{n+1}|X_n)\pi dX_n$, then the distribution π is called a **stationary distribution** of the Markov chain $\{X_n\}$ and corresponds to eigenvalue 1. The one-step transition probability of a Markov chain with finite number of states can be expressed as a **transition**

matrix $(P_{i,j}) = (P(X_{n+1} = j | X_n = i))$. This matrix is a **Markov matrix** since its rows sum up to one, $\sum_j P_{i,j} = 1$.

A **hidden Markov process** $\{(X_n, Y_n)\}$ is composed of two stochastic processes: $\{X_n\}$, and $\{Y_n\}$. The **latent state** process $\{X_n\}$ is a Markov chain with finite number of states. It is parameterized by the **initial probability distribution**

$$\vec{\pi} \quad = \quad \begin{pmatrix} P(X_1 = 1) & \cdots & P(X_1 = M) \end{pmatrix} \tag{1}$$

, and the **state transition matrix**

$$A \quad = \quad \begin{pmatrix} P(X_{n+1} = 1 | X_n = 1) & \cdots & P(X_{n+1} = M | X_n = 1) \\ \vdots & \ddots & \vdots \\ P(X_{n+1} = 1 | X_n = M) & \cdots & P(X_{n+1} = M | X_n = M) \end{pmatrix} \tag{2}$$

. The **observation** process $\{Y_n\}$ is coupled with the latent state process through a memoryless channel. In other words, the probability distribution of Y_n is independent of $X_m, m \neq n$ given X_n, $P(Y_n | \{X_n\}) = P(Y_n | X_n)$. The types observation of our interest are finite alphabet, Gaussian, and mixture of Gaussians. When the observation is finite alphabet, it can be parameterized by an **observation matrix**

$$\theta \quad = \quad \begin{pmatrix} P(Y_n = 1 | X_n = 1) & \cdots & P(Y_n = N | X_n = 1) \\ \vdots & \ddots & \vdots \\ P(Y_n = 1 | X_n = M) & \cdots & P(Y_n = N | X_n = M) \end{pmatrix} \tag{3}$$

When the observation is Gaussian, it can be parameterized by the means and variances corresponding to individual latent states

$$\theta \quad = \quad \{P(Y_n | X_n = 1) = \mathcal{N}(Y_n; \mu_1, \sigma_1^2), \cdots P(Y_n | X_n = M) = \mathcal{N}(Y_n; \mu_M, \sigma_M^2)\} \tag{4}$$

When the observation is mixture of Gaussians, it can be parameterized by the means and variances corresponding to individual latent state – mixture identifier pairs.

$$\begin{aligned} \theta \quad &= \quad \left\{ \begin{matrix} P(Y_n | X_n = 1, \tau = 1) & \cdots & P(Y_n | X_n = 1, \tau = C) \\ \vdots & \ddots & \vdots \\ P(Y_n | X_n = M, \tau = 1) & \cdots & P(Y_n | X_n = M, \tau = C) \end{matrix} \right\} \\ &= \quad \left\{ \begin{matrix} \mathcal{N}(Y_n; \mu_{11}, \sigma_{11}^2) & \cdots & \mathcal{N}(Y_n; \mu_{1C}, \sigma_{1C}^2) \\ \vdots & \ddots & \vdots \\ \mathcal{N}(Y_n; \mu_{M1}, \sigma_{M1}^2) & \cdots & \mathcal{N}(Y_n; \mu_{MC}, \sigma_{MC}^2) \end{matrix} \right\} \end{aligned} \tag{5}$$

The usage of hidden Markov processes often involves the following problems. (1) Given an observation sequence of an HMM, as well as the corresponding latent state sequence, find out the characterization of this HMM. (2) Given the characterization of the latent state process of an HMM ($\{\pi, A\}$) and an observation sequence generated by this HMM ($\{Y_n\}$), infer the corresponding latent state distributions $P(X_n | \{Y_n\})$. (3) Given an observation sequence generated by an HMM, find the best parameters (in the maximum likelihood estimation sense) that accounts for the observations sequence, and the corresponding latent state distributions.

Given an HMM parameterized by (π, A, θ) and a realization of this HMM y^n, the maximum likelihood latent state distribution can be estimated using the **forward-backward algorithm** in terms of the **forward**

parameters $\alpha_t(s)$, and the **backward parameters** $\beta_t(s)$. The statistics involved are:

$$\begin{aligned}
\alpha_t(s_t) &= P(s_t, y_1 \cdots y_t) \\
\beta_t(s_t) &= P(y_{t+1} \cdots y_T | s_t) \\
\gamma_t(s_t) &= P(s_t, y_1 \cdots y_T) \\
\xi_{t \to t+1}(s_t, s_{t+1}) &= P(s_t s_{t+1}, y_1 \cdots y_T)
\end{aligned}$$

The iteration is given as the following:

$$\alpha_1(s_1) = \pi \cdot P(y_1|s_1) \tag{6}$$

$$\alpha_{t>1}(s_t) = \sum_{s_{t-1}=1}^{M} \alpha_{t-1}(s_{t-1}) \cdot a_{s_{t-1}s_t} \cdot P(y_t|s_t) \tag{7}$$

$$\beta_T(s_T) = 1 \tag{8}$$

$$\beta_{t<T}(s_t) = \sum_{s_{t+1}=1}^{M} a_{s_t s_{t+1}} \beta_{t+1}(s_{t+1}) \cdot P(y_{t+1}|s_{t+1}) \tag{9}$$

$$\gamma_t(s_t) = \alpha_t(s_t) \cdot \beta_t(s_t) \tag{10}$$

$$\xi_{t \to t+1}(s_t, s_{t+1}) = \sum \alpha_t(s_t) \cdot \beta_{t+1}(s_{t+1}) \cdot P(y_{t+1}|s_{t+1}) \tag{11}$$

The maximum latent state assignment can be computed using the **Viterbi algorithm**

$$\begin{aligned}
\delta_1(s_1) &= \pi_1 \cdot P(y_1|s_1) \\
\psi_1(s_1) &= s_1 \\
\delta_t(s_t) &= \max_{s_{t-1}} \delta_{t-1}(s_{t-1}) \cdot P(s_t|s_{t-1}) \cdot P(y_t|s_t) \\
\psi_t(s_t) &= \texttt{argmax}_{s_{t-1}} \delta_{t-1}(s_{t-1}) \cdot P(s_t|s_{t-1}) \cdot P(y_t|s_t) \\
\texttt{path}(T) &= \texttt{argmax}_{s_T} \delta_T(s_T) \\
\texttt{path}(t < T) &= \psi_{t+1}(\texttt{path}(t+1))
\end{aligned}$$

Given the observations $(y_t)_{1 \le t \le T}$, the maximum likelihood estimation of the parameters, as well as the corresponding latent states $(s_t)_{1 \le t \le T}$ can be computed via the **EM algorithm**. The EM algorithm works by alternating between two steps:

E-step inference of latent state distributions from the parameters and the observations. The statistics to be inferred are $\alpha_t\left(s_t; \pi, A, p(y_t|s_t)\right)$, $\beta_t\left(s_t; \pi, A, p(y_t|s_t)\right)$, $\gamma_t\left(s_t; \pi, A, p(y_t|s_t)\right)$, $\xi_{t \to t+1}$. We use Equations 6-11 to infer the latent state distributions.

M-step maximization of the **expected log likelihood** $\mathrm{E}_{(s_t)_{1 \le t \le T}}\left(\log\left((y_t)_{1 \le t \le T} \,\middle|\, (s_t)_{1 \le t \le T}\right)\right)$ with the latent states inferred in the previous E-step and the observations.

The parameters related to the latent states are maximized in the following way:

$$\begin{aligned}
\pi_s &= \gamma_1(s) \\
A_{ij} &= \frac{\sum_{t=1}^{T-1} \xi_{t \to t+1}(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}
\end{aligned}$$

The parameters related to finite observations are maximized in the following way:

$$P(y|s) = \frac{\sum_{t=1}^{T} \gamma_t(s) \cdot \delta(y_t, y)}{\sum_{t=1}^{T} \gamma_t(s)}$$

where

$$\delta(i,j) \quad = \quad \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

is the Kronecker delta function. The parameters related to Gaussian observations are maximized in the following way:

$$\begin{aligned} \vec{\mu}_s &= \frac{\sum_{t=1}^{T} \gamma_t(s) \cdot \vec{y}_t}{\sum_{t=1}^{T} \gamma_t(s)} \\ \Sigma_s &= \frac{\sum_{t=1}^{T} \gamma_t(s) \cdot \vec{y}_t \cdot \vec{y}_t^{\mathsf{T}}}{\sum_{t=1}^{T} \gamma_t(s)} - \vec{\mu}_s \cdot \vec{\mu}_s^{\mathsf{T}} \end{aligned}$$

where $\vec{\mu}_s$ and $\vec{y}_t$ are column vectors, A^{T} means the transpose of matrix$A$.

We often formulate the EM algorithm in matrix form, since vectorized formulas are computationally more efficient in Matlab or S/S+. Sometimes vectorized formulas are easier to understand. When we formulate the statistics in matrix form,

$$\begin{aligned} \vec{\alpha}_t &= \begin{pmatrix} \alpha_t(s_t = 1) & \cdots & \alpha_t(s_t = M) \end{pmatrix} \\ \vec{\beta}_t &= \begin{pmatrix} \beta_t(s_t = 1) \\ \vdots \\ \beta_t(s_t = M) \end{pmatrix} \\ \vec{\gamma}_t &= \begin{pmatrix} \gamma_t(s_t = 1) & \cdots & \gamma_t(s_t = M) \end{pmatrix} \\ \theta_t &= \begin{pmatrix} P(y_t|s_t = 1) & & \\ & \ddots & \\ & & P(y_t|s_t = M) \end{pmatrix} \\ \texttt{diag}[\begin{pmatrix} x_1 & \cdots & x_M \end{pmatrix}] &\triangleq \begin{pmatrix} x_1 & & \\ & \ddots & \\ & & x_M \end{pmatrix} \end{aligned}$$

the forward-backward algorithm can be expressed as the following matrix form:

$$\begin{aligned} \vec{\alpha}_1 &= \vec{\pi} \cdot \theta_1 \\ \vec{\alpha}_{t>1} &= \vec{\alpha}_{t-1} \cdot A \cdot \theta_t \\ \vec{\beta}_T &= 1 \\ \vec{\beta}_{t<T} &= A \cdot \theta_t \cdot \vec{\beta}_{t+1} \\ \vec{\gamma}_t &= \vec{\alpha}_t \cdot \texttt{diag}[\vec{\beta}_t] \\ \xi_{t \to t+1} &= \texttt{diag}[\vec{\alpha}_t] \cdot A \cdot \texttt{diag}[\theta_{t+1} \cdot \vec{\beta}_{t+1}] \end{aligned}$$

The joint latent state inference and parameter estimation is normally computed by the **EM algorithm**. The parameter maximization step proceeds in the following way. The parameters involved with the latent states can be computed as the following

$$\begin{aligned} A &= \texttt{normalize}[\sum_{t=2}^{T} \xi_{t-1 \to t}] \\ \vec{\pi} &= \texttt{normalize}[\vec{\gamma}_1] \end{aligned}$$

The parameters related to finite observations can be re-estimated using the following formulas:

$$p(\vec{y}|\vec{s}) \quad = \quad \texttt{normalize}[\sum_t \vec{\gamma}_t^{\mathsf{T}} \cdot \vec{\delta}_{y_t,\vec{y}}]$$

The parameters related to Gaussian observations can be re-estimated using the following formulas:

$$\begin{aligned} \vec{\mu} &= \frac{\sum_t \vec{y}_t \cdot \vec{\gamma}_t^{\mathsf{T}}}{\sum_t \vec{1}_{m_c \times 1} \cdot \vec{\gamma}_t^{\mathsf{T}}} \\ \Sigma_i &= \frac{\sum_t \gamma_t(i) \cdot \vec{y}_t \cdot \vec{y}_t^{\mathsf{T}}}{\sum_t \vec{1}_{m_c \times 1} \cdot \vec{\gamma}_t^{\mathsf{T}}} - \vec{\mu}_i \cdot \vec{\mu}_i^{\mathsf{T}} \end{aligned}$$

Example 2. *The speaking/non-speaking status of a person can be modeled as a hidden Markov process. The latent states are whether this person is currently uttering a sentence. The observations are the indicators of vowel utterances.*

In the below, we train the HMM-based speaking/non-speaking classifier with the audio clip "well, happy birthday, Juan Carlos". Afterwards, we apply the trained model to the same audio clip, and find the latent state distributions conditioned on the observations, as well as the most likely latent state sequence (the Viterbi path).

The training clip looks like this:

```
[s,f,t]=spectrogram(aud,256,128,[],8000);
```

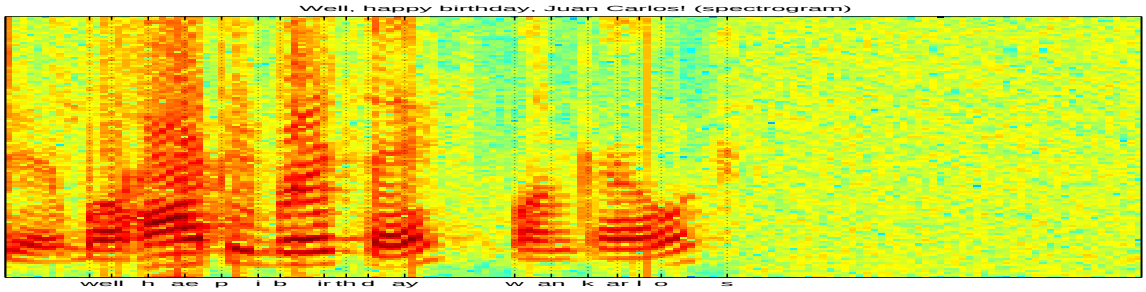

Figure 1: The spectrogram of a training clip

The latent states are hand labeled in the following way:

```
stairs(s);
```

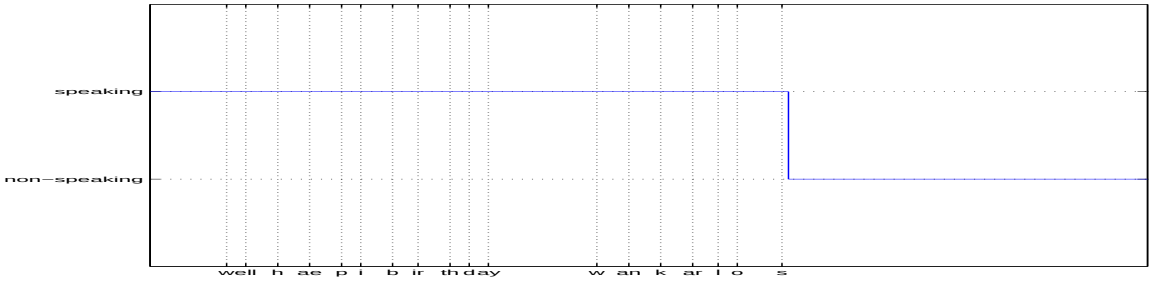

Figure 2: The features for the training clip

The features[1] *(i.e., the observations of the hidden Markov process) we use looks like this:*

```
feat = fast_voicing_features(aud,256,128,sum(aud.^2)/length(aud)/5);
```

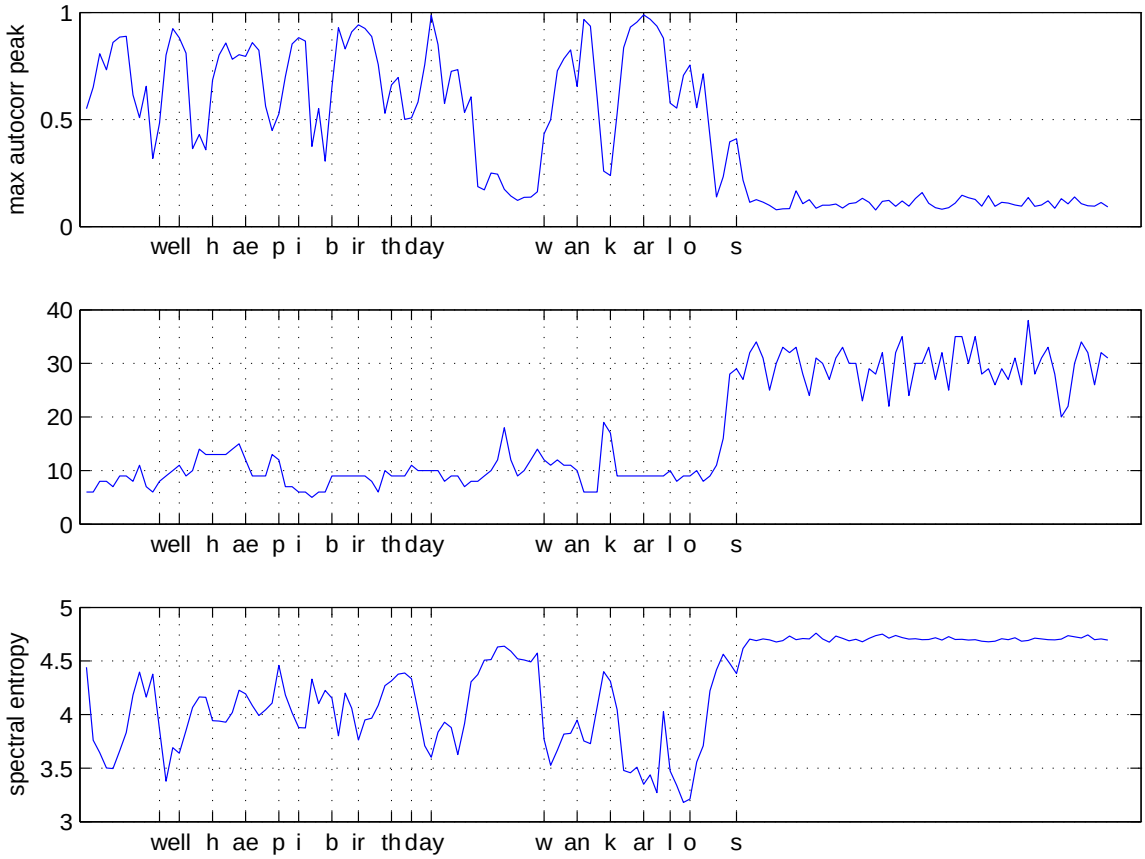

Figure 3: The features for the training clip

We proceed to train a hidden Markov model with mixture of Gaussians observations. The reason why we use a hidden Markov model is that there are two discrete states in the system: speaking, and non-speaking. Each of these two states moves to the next state with a relatively invariant transition probability. The reason why we use mixture of Gaussians model to fit the observations (i.e. the audio features) is that one Gaussian is not powerful enough to fit the features well.

The parameters corresponding to the latent states are the transition matrix A, and the initial sate distribution π. The initial state distribution π only affects the log likelihood slightly, and we normally use the eigenvector corresponding to eigenvalue 1 of the state transition matrix A. Since the latent states are already given as $S_t = s_t$, the one-slice parameters $\gamma_t(i)$ and the two-slice parameters $\xi_{t\to t+1}(i,j)$ can be solved directly from the latent state assignment:

$$\begin{aligned} \gamma_t(i) &= \delta(s_t, i) \\ \xi_{t\to t+1}(i,j) &= \delta(s_t, i) \cdot \delta(s_{t+1}, j) \end{aligned}$$

```
>> A = full(sparse(s(1:end-1),s(2:end),ones(length(s)-1,1)));
```

[1] A discussion of the three features can be found in Basu [1]

```
>> A(1,2) = 1; % hack! add a transition from non-speaking to speaking
>> A = diag(1./sum(A,2))*A;
>> disp(A)
    0.9999    0.0001
    0.0001    0.9999
```

The parameters related to the observations are the mixture prior, and the means and covariance matrices for the various Gaussian distributions. Suppose we use a mixture of M Gaussian distributions to capture the observations y corresponding to state s, and the means and covariance matrices are μ_i, and Σ_i for the i-th Gaussian distribution in the mixture. Then the probability density of an observation $Y_t = y_t$ conditioned on a latent state $S_t = s_t$ equals to the weighted sum of the probability densities for various Gaussian distribution components. The weights are the prior probabilities that the observation y_t is taken from the i-the Gaussian distribution:

$$P(Y_t = y_t|S_t = s_t) \quad = \quad \sum_i \mathcal{N}(y_t; \mu_i, \Sigma_i) \cdot P(M_t = i|S_t = s_t)$$

The mixture priors and the various Gaussian distribution parameters are computed using the EM algorithm[2]*. The reason why the likelihoods decrease is that two mixtures are more than what we need for this training clip. As a result, one component Gaussian distribution for each of the two latent states cannot capture enough observations.*

```
>> feat1 = feat(:,s(1:128:128*size(feat,2))==1);
>> [mu1,sigma1,prior1] = mixgauss_em(feat1,2,'cov_type','diag');
******likelihood decreased from 6.4552 to 3.8445!
>> feat2 = feat(:,s(1:128:128*size(feat,2))==2);
>> [mu2,sigma2,prior2] = mixgauss_em(feat2,2,'cov_type','diag');
******likelihood decreased from 2.2061 to 2.2042!
```

Thus, the trained parameters for the HMM-based speaking/non-speaking classifier are

$$\begin{aligned}
A &= \begin{pmatrix} .9999 & .0001 \\ .0001 & .9999 \end{pmatrix} \\
\pi &= \begin{pmatrix} .5 & .5 \end{pmatrix} \\
P(Y_t = y|S_t = 1) &= \mathcal{N}\left(y; \mu_1 = \begin{pmatrix} .1090 \\ 29.6727 \\ 4.7067 \end{pmatrix}, \Sigma_1 = \begin{pmatrix} .0105 & & \\ & 1.6374 & \\ & & .0104 \end{pmatrix}\right) \\
P(Y_t = y|S_t = 2) &= .0862 \cdot \mathcal{N}\left(y; \mu_1 = \begin{pmatrix} .2551 \\ 20.1436 \\ 4.4968 \end{pmatrix}, \Sigma_1 = \begin{pmatrix} .0105 & & \\ & 1.6374 & \\ & & .0104 \end{pmatrix}\right) + \\
&\quad .9138 \cdot \mathcal{N}\left(y; \mu_1 = \begin{pmatrix} .6590 \\ 9.2836 \\ 3.9553 \end{pmatrix}, \Sigma_1 = \begin{pmatrix} .0180 & & \\ & 38.0866 & \\ & & .0224 \end{pmatrix}\right)
\end{aligned}$$

Let us apply the trained HMM to the same features and find out the latent state probability distributions conditioned on the given observation sequence. We first need to find out the observation likelihoods $P(Y_t = y_t|S_t)$, i.e., the probabilities of the observations $Y_t = y_t$ conditioned on the latent states $S_1 = 1$ and $S_t = 2$.

[2]The Matlab functions mixguass_em.m, mixgauss_prob.m, fwdback.m, and viterbi_path.m was written by Murphy [2].

```
Bx = mixgauss_prob(feat,mu1,sigma1);
By = mixgauss_prob(feat,mu2,sigma2);
plot([prior1'*Bx;prior2'*By]')
```

The observation likelihoods are plotted as the following:

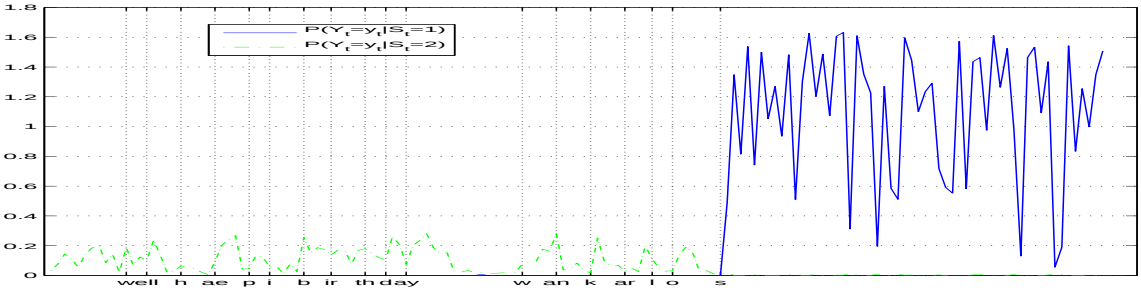

Figure 4: The features for the training clip

The forward parameters $\alpha_t(s_t)$, backward parameters $\beta_t(s_t)$, one-slice parameters $\gamma_t(s_t)$, and two-slice parameters $\xi_{t\rightarrow t+1}(s_t, s_{t+1})$ can be computed from π, A, $P(Y = y_t|S_t)$ in the following way:

```
[alpha,beta,gamma] = fwdback([.5 .5], [.9999 .0001;.0001 .9999],...
                             [prior1'*Bx;prior2'*By]);
obslik = [prior1'*Bx;prior2'*By];
for i=1:size(alpha,2)-1
  xi(:,:,i) = [.9999 .0001;.0001 .9999].*...
      (alpha(:,i)*(beta(:,i).*obslik(:,i))');
  xi(:,:,i) = xi(:,:,i)/sum(sum(xi(:,:,i)));
end
```

The one-slice parameters are plotted as the following

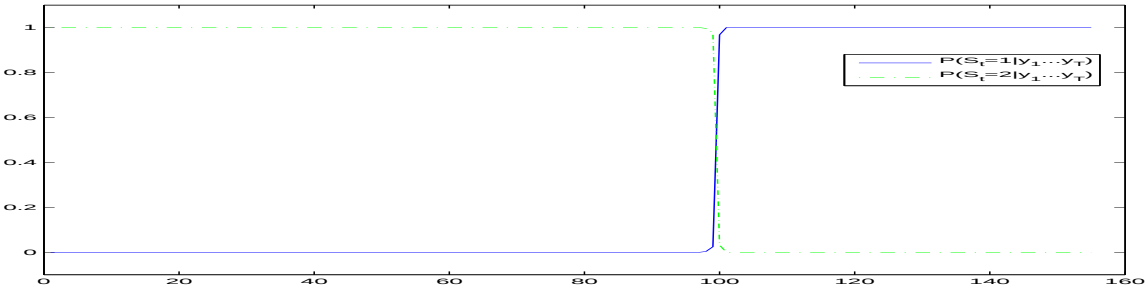

Figure 5: The features for the training clip

The Viterbi path can be computed in the following way:

```
obslik = [prior1'*Bx;prior2'*By];
path = viterbi_path([.5 .5],[.9999 .0001;.0001 .9999],obslik);
```

An interesting theoretical issue with the hidden Markov model is about its representability. In other words, what kinds of applications are suitable for hidden Markov modeling; What type of statistical characteristics is captured by a hidden Markov model; If an application is suitable for hidden Markov modeling,

how many latent states are needed (the appropriate order of the modeling hidden Markov process); If an application is suitable for hidden Markov modeling, and if we have the correct number of latent states, are we guaranteed to compute the correct model when the length of the sample sequence tends to infinity. According to the understanding of the author, although there are many theoretical answers to the above questions for Wiener filter / Kalman filters, only a few things are known for hidden Markov models.

Intuitively, the hidden Markov model is suitable for symbolically controlled processes. Examples of such processes are: computer programs, a worker following a list of instructions, a diary of a human's daily life, an utterance of a sentence. Although those processes can present themselves as real vector sequences $(X_t)_{1 \leq t \leq +\infty}$, where $X \in \mathcal{R}^n$, they are actually controlled by a discrete number of "states" and the procedure to follow from one state to another state. The states for the above example processes are: the variables in computer programs, the different situations on a work site, the schedule of a person, and the words in the language. The procedures to follow for the above example processes are: the control structures in computer programs, a description of what to do in different situations and what new situations to expect, the arrangement of the schedule, and the grammar of the language. Since many human activities and human society activities operate on the natural language, which is symbolic by nature, we would expect many applications of the hidden Markov process in those areas.

For a symbolically controlled process, it is a good practice to investigate the data, and compare the intuitively figured out control structure with the computed control structure based on hidden Markov modeling. For a simple symbolically controlled process, a human can normally understand the "control" of this process by observing it, via the Occam's razor principle (entities should not be multiplied beyond necessity). For a complex symbolically controlled process, a human can normally find out ways to approximately understand the control of this process by observing it. The computed control structures of those processes, based on the maximum likelihood principle, normally resemble the intuitively figured out control structures.

Ephraim [3] reviewed the history, theory, algorithms of the hidden Markov process, as well as the extensions of the original finite-state finite-alphabet hidden Markov process. In this review, the extended models were applied to speech recognition, communication theory, signal processing in audio, biomedical, and image, fault detection, economics, and meteorology. Cappe [4] reviewed the applications of the hidden Markov process from year 1989 to year 2000 in the following fields, with about 360 references: acoustics, biosciences, climatology, control, communications, econometrics, handwriting and text recognition, image processing and computer vision, signal processing, speech processing, and misc applications. The two reviews give a comprehensive idea of what a hidden Markov model is, and what a hidden Markov model is for.

Baum [5] proved the convergence of the maximum likelihood (ML) parameter estimation from an observation sequence of a hidden Markov process to the true parameters of the hidden Markov process, when the length of the observation sequence tends to infinity. Baum [6] also gave the (iterative) expectation maximization (EM) algorithm for estimating the parameters of a hidden Markov process from an observation sequence. The EM algorithm will attain a local maximum when the number of iterations tends to infinity. However, it should be noted that the convergence of the EM algorithm to the true parameters of the hidden Markov model is not guaranteed, even when the length of the observation sequence tends to infinity.

Anderson [7] gave a set of sufficient conditions for a finite-state finite-alphabet hidden Markov process to be realizable, and derived a constructive solution to the realization problem under those conditions. The problem under the discussion of [7] was: for a unknown hidden Markov process with finite number of states and finite number of observation symbols, under what conditions can we reconstruct this hidden Markov model from the probabilities of all finite-length output strings generated by it. The set of sufficient conditions given in [7] are: 1) The unknown hidden Markov process is *time-invariant*, i.e., its parameters does not change with time. 2) The state transition matrix of this unknown hidden Markov model is *irreducible*, i.e., all states of this unknown hidden Markov matrix can still be visited with a probability greater than some positive constant after an infinite time. 3) The observation sequences of the unknown hidden Markov process

are *long term independent*, i.e., $\lim_{|w|\to+\infty}\sum_w p(uwv) = p(u)p(v)$. 4) The observation sequences tend to be independent exponentially fast with their distance (*exponential forgetting*), i.e., $\frac{p(u|vw)}{p(u|v)} \sim 1 - O(e^{-|v|})$. In 3) and 4), u, v, w are consecutive observation sequences, and u occurs earlier.

Chapter 1

Introduction

The dynamics of a human related behavior normally involves a large probability space with sparsely distributed states. The latent structure influence process effectively compresses the state spaces of such behaviors and finds out the clusters by emulating the interaction of a group of humans/agents. When the latent structure influence process is applied to non-human-related stochastic processes as a team-of-experts model, there are pros and cons on whether a team of experts out-performs a single expert. In this chapter, we describe several data sets of either human behaviors or human group behaviors, and illustrate their large and highly clustered state spaces. We also discuss the pros and cons of the *team of experts* approach to general stochastic processes. We conclude this Chapter with a formal definition of the latent structure influence process.

1.1 The Stochastic Processes of Human Behaviors

The stochastic processes under discussion in this thesis are related to human behaviors involving multiple categories of concepts, or human group behaviors. Those behaviors have the following two properties. First, they normally involve large probability spaces, with sparsely distributed states. Second, the behaviors do not change rapidly with time.

For the first property of human-related behaviors, a behavior is often encoded by an n-tuple. Each position of this n-tuple takes one of a finite number of states, and represents a sub-behavior. As a result of the encoding, the size of a behavior space is the multiplication of the sizes of its sub-behavior spaces. The sub-behaviors are highly clustered, since they combine with each other in a very limited number of ways. As a result, the behaviors can actually take a very limited number of states. For example, a human group behavior can be encoded as: *A is in his office while B is in his apartment*. The behaviors of the individuals (i.e., the sub-behaviors of a group behavior) can in principle combine freely to form group behaviors. However, a human group normally shows some structure, and some group behaviors appear very rarely. In this example, it is very likely that most people are in their offices during the work hours, and in their apartments in the midnights. For another example, a human behavior can be encoded as: *I am sitting in the restaurant, eating, and it is noisy around me*. The locations, actions, and speaking/non-speaking status of a person can in principle combine freely for form human behaviors. But some combinations appear much more frequently than other combinations.

For the second property, the human-related behaviors are also smooth/continuous in the sense that the knowledge of the behavior of a stochastic process at time t provides information in predicting the behavior of the stochastic process at time $t + \triangle t$ for a small $\triangle t$. In addition, the smaller the $\triangle t$ is, the more information we have for predicting the behavior at $t + \triangle t$. For example, *given that A is in his office, A is very likely to remain in his office one minute later, and A is even more likely to be in his office one second later*.

Of the two properties of human-related behaviors, the smoothness property can be exploited to denoise the observation sequences and to gain a better understanding of the behavior sequences we are modeling. The large probabilistic state space for describing the behaviors is unfavorable, since modeling a large number of states often requires a large number of parameters, and results in overfitting. We can effectively compress the large probabilistic state space by exploiting the fact that only a few states appear often. The latent structure influence process explores the smoothness property. At the same time, it overcomes the difficulty of overfitting by studying how the probability distributions of constituent behaviors (e.g. *A is in his office, B is in his office, I am in the restaurant, I am sitting*) at a sample time t linearly combine to predict the probability distributions of them at sample time $t + 1$, and how to update the probability distributions of the constituent behaviors individually according to their respective observations. When the probability distributions of the constituent behaviors are not linearly related, we adopt the feature trick to map the old collection of constituent behaviors to a new collection of constituent behaviors with linearly related probability distributions, and work on the new collection of constituent behaviors.

One such stochastic process is the cellphone usage data collected in the Reality Mining project. In this project, Eagle and Pentland [8] recorded the cellphone usages of 81 participants from the MIT community for over nine months. The recording includes the participants' cellphone communications, their proximity information, their locations, and their activities. These types of information are indicated by the participants' voice/SMS conversations, cellphone Bluetooth scanning, cell tower usages, and cellphone on/off status. This data set provides the ground truth to reconstruct the participants' activities and communications, as well as to answer questions such as, what are the participants' relations, how their relations change over time, and how the participants' relations and their individual behaviors influence each other.

A dynamic modeling of such data sets answering the above questions involves a complex behavior space in nature: Suppose our interest is in the interaction dynamics of the participants' schedules, and we assign only two latent states (out-of-office/in-office) to each participant in our analysis. Since the participants' states can combine freely with each other, we will end up with 2^{81} number of states for the whole system, and this number of states is intractable. A dynamic model can cope with this large probability space by automatically factoring the 81 participants into several clusters, and placing the persons with similar schedules into the same cluster. As a result, any two different clusters are almost unrelated, and they can be studies independently. Each cluster requires only two latent states, and it can be studied easily.

The data set is also noisy, and the noises can be filtered out by exploiting the smoothness property of human-related behaviors. Examples of such noises are: a user might run out of his flash memory/battery and cause data loss, he might leave his cellphone in his office/home thus the cellphone recording no longer reflects his behaviors, the Bluetooth device might fail to record proximity information, or record the proximity of people at the other side of certain types of walls. Thus a dynamic algorithm considering the participants' past behaviors and the participants' relations with each other will definitely remove certain types of noises and result in better performance.

The data sets collected by the Life-Wear system [9] provide another example of the complex stochastic process involved with human behaviors. In a Life-Wear system, data is collected in real-time from several accelerometers, microphones, cameras, and a GPS, all attached to different parts of a soldiers' clothing. Inference of soldier state is made in real-time, and data automatically shared among different soldiers wearing the Life-Wear systems based on the pattern of activity shown among the group of soldiers. In an early Life-Wear system [10], we were required to infer 8 locations (office, home, outdoors, indoors, restaurant, car, street, and shop) , 6 speaking/non-speaking status (no speech, I-speaking, other-speaker, distant voices, loud crowd, and laughter), 7 postures (unknown, lie, sit, stand, walk, run, and bike), and 8 events (no-event, eating, typing, shaking-hands, clapping-hands, driving, brushing teeth, and doing the dishes). There were $8 \times 6 \times 7 \times 8 = 2688$ number of different combinatorial states.

Similar to the Reality Mining project, we have a large behavior space involving 2688 number of combined behaviors. This means that a carelessly constructed model need to characterize all 2688 behaviors,

and the probabilities that the user of a Life-Wear system move from one of the 2688 number of behaviors to another. This is not necessary, since a concept from one category does not change the characteristics of another concept in a different category, although the former will bias the latter. As a result, we can study different categories of concepts independently, and bias the probability distributions among the concepts in a category with the probability distributions among the concepts in another category. For example, we sit in the same posture no matter whether we are indoors or outdoors. However, we are more likely to be sitting when we are indoors than when we are outdoors.

The data sets are noisy, and sensor failures are unavoidable due to insufficient power supply, sensor faults, connection errors, or other unpredictable causes. This means that an inference algorithm for soldier state must be robust against sensor noises and sensor failures.

The latent structure influence model copes with this problem by simultaneous learning the structure of multi-agent interaction and applying the learned structural information in combining past evidence. We believe that our latent structure influence model is an efficient, robust method for modeling the dynamics of interacting processes. It is in the tradition of N-heads dynamic programming on coupled hidden Markov models [11], the observable structure influence model [12], and the partially observable influence model [13], but extends these previous models by providing greater generality, accuracy, and efficiency.

1.2 Dynamic Team-of-experts Modeling

A dynamic team-of-experts model emulates the way a group of persons monitor a complex stochastic process and influence each other. This model is an abstraction of the human-related behaviors and should fit them well. There are pros and cons on whether it should be applied to a non-human-related stochastic process. In the same fashion, a group of persons are not guaranteed to out-perform an individual in general.

The dynamic team-of-experts approach can have the following benefits. (1) When the evidence is composed of several heterogeneous types of features, it is generally not a good idea to assume that the feature vector consisting of different types of features obeys a Gaussian distribution, or a mixture of Gaussian distributions. A better idea is to assign different types of features or different combinations of them to different experts, and let the experts adjust their models according to the *conclusions* of each other. The final conclusion comes by combining the *conclusions* of different experts. For example, when we classify the accelerometer recording from a person into different postures, the features we consider might include: the short window spectrograms, the number of peaks and the maximum value of the peak amplitudes in a short window, the mean and variance. It is normally better to inspect those different features individually, and combine the computation results from those features. (2) When a classification problem or a data mining problem is too complex to be solved by any single method/expert, the performance of a combination of different methods is at least as good as any single method, if we know the performances of different experts in different situations. (3) The experts can compare their results and adjust their models, so that their results with polarize towards / away from the results of each other. (4) The result of a team of experts is generally less sensitive to errors of a single expert, since the errors of any expert is restricted by their *influence.*

However, the dynamic team-of-experts approach can have the following drawbacks. (1) When we assign heterogeneous types of features or combinations of them to different experts, we generally do not know a priori what combinations of different features is enough for an expert to come to a conclusion. As a result, some experts may end up inconclusive due to the lack of information. (2) Sometimes the conclusions of the experts are very different, and the performances of them may be hard to evaluate or quantify. In this situation, combining the conclusions of the experts is not much easier than working directly on the feature vectors.

A striking characteristic of group learning is the group polarization phenomenon. This phenomenon was first presented by Stoner [14], who observed that after a group of persons discussed their individual

decisions on a imaginary life-or-death issue, and put their arguments together, the group decision as a whole tend to be more risky than the average of the individual decisions. In our latent structure influence process modeling, the group polarization phenomenon results in the following feedback process and reduces the overall entropy of the team-of-experts. When the experts compare their conclusions with each other, they tend to favor those experts (including themselves) who agree with them, will adjust their models to coincide with those of the favorable experts, and will be willing to take move influence from them. The adjusted models and influence coefficients will in turn make the favorable experts even more favorable. As a result, group polarization occurs.

1.3 The Latent Structure Influence Process

The latent structure influence process models the dynamics of human-related behaviors by simulating the interaction of a group of humans/agents. It compresses the large probability space of the sample sequences by clustering/polarizing different perspectives of the human-related behaviors. In the below, we use the Life-Wear example to motivate the latent structure influence process approach, compares and contrasts the influence approach with other state-space approaches, and give a formal definition of the latent structure influence process.

The four state-space models under our comparison are: the latent structure influence process, the coupled hidden Markov process, the hidden Markov process, and the dynamic Bayesian network. The Life-Wear system is an early prototype of the more complex DARPA ASSIST system [9]. The Life-Wear system samples its wearer's behavior with an accelerometer at his right hip, another accelerometer at his left wrist, a microphone at his chest, and a camera at his chest. Based on the sampled data sequences, the Life-Wear system understands the behavior of its user by inferring the probability distributions among eight locations, six speaking / non-speaking status, seven postures, and eight events at each sample time (i.e., where the user is, whether he is speaking, what he is doing, and whether something interesting is taking place). The inference algorithms for the four state-space models have a similar form: they all work by alternating between the time update step, and the measure update step. In the time update step, the Life-Wear system computes the probability distributions at the next sample time from the probability distributions at the current sample time. In the measure update step, the Life-Wear system adjusts the probability distributions by the new evidence collected with the accelerometers, the microphone, and the camera. In other words, the time update step finds the best guess of the behavior of the Life-Wear user based on the model and the past evidence, and the measure update step adjusts the guess with new evidence. Beyond this seemingly similar form of their inference algorithms, the four models work with the probability distributions in different measure spaces, and execute the time/measure update steps differently. As a result, they have different computational performances.

- The latent structure influence process works with the marginal probability distributions of the locations, the speaking/non-speaking status, the postures, and the events, respectively. The time update step computes the marginal probability distributions at the next sample time by linearly combining the marginal probability distributions at the current sample time. When we write the marginal probability distributions into a row vector, the *influence matrix* that is used to update the marginal probability distributions is learned and plotted in Figure 3.3. The measure update step incorporates the new evidence into the marginal probability distributions just computed.

- The coupled hidden Markov process works with the joint probability distribution of the joint states. Each joint state consists of one of the eight locations, one of the six speaking/non-speaking status, one of the seven postures, and one of the eight events. As a result, the latent state space has $8 \times 6 \times 7 \times 8 = 2688$ number of states, and the state transition matrix for time updating is a 2688×2688 square matrix.

The state transition matrix is unnecessarily sparse and can be largely compressed (while preserving its eigen-behavior).

- We can use a single hidden Markov process with several states to model the stochastic process sampled by the Life Wear system and involving a combination of locations, speaking/non-speaking status, postures, and events. However, a single hidden Markov process with several states is not expressive enough for the transitions of the combined states, and the compression of the 2688×2688 state transition matrix of the coupled hidden Markov process involves a more complicated procedure than just adopting a single hidden Markov process. Alternatively, we can use four hidden Markov processes to model the transitions of locations, speaking/non-speaking status, postures, and events, respectively. In this way, we have simplified the coupled hidden Markov process with the cost of a less accurate understanding of a Life-Wear user's behavior. The latter model can be considered as a latent structure influence process with no influence among different perspectives. When we write the marginal probability distributions into a row vector, the influence matrix in this case is the matrix in Figure 3.3 whose off-diagonal sub-matrices are all zeros.

- When we use a dynamic Bayesian network to model a Life-Wear user's behavior, we first need to define a set of random variables and analyze the conditional (in)dependence among those random variables. We then represent the conditional (in)dependence with a graph, and use the message passing algorithm to infer the probability distributions at the sample times. For this approach, constructing the graph representation of a stochastic process is a non-trivial task, the message passing algorithm might be computationally prohibiting depending on the clique size, and manually constructed graph representations do not scale with the modifications of the problem.

Based on the above comparison, we comment that the latent structure influence process models a complex stochastic process in terms of how different perspectives of this stochastic process influence each other linearly and cluster/polarize. With this linearity assumption, we can effectively control the model complexity, and capture useful structural information inherent in the stochastic process at the same time.

In the definition below, the random variable $S_t^{(c)} = \{1, \cdots, m_c\}$ represents the perspective c at sample time t, and the random variable $Y_t^{(c)}$ represents the evidence sampled at time t for the adjustment of $S_t^{(c)}$. We assume that the (marginal) random variables $S_t^{(c)}$ can be updated marginally according to their influences without involving the combined random variable $S(S_t^{(1)} \cdots S_t^{(C)})$, as shown in the equations (1.1, 1.2) below.

Definition 1. *A **latent structure influence process** is a stochastic process $\{S_t^{(1)}, \cdots, S_t^{(C)}, Y_t^{(1)}, \cdots, Y_t^{(C)}\}$. In this process, the **latent variables** $S_t^{(1)}, \cdots, S_t^{(C)}$ are finite-state $S_t^{(c)} \in \{1, \cdots, m_c\}$ and their (marginal) probability mass functions are defined as the following:*

$$P(S_1^{(c)} = s) = \pi_s^{(c)} \tag{1.1}$$

$$P(S_{t+1}^{(c)} = s_{t+1}^{(c)} | S_t^{(1)} = s_t^{(1)} \cdots S_t^{(1)} = s_t^{(1)}) = \sum_{c_1=1}^{C} h^{(c_1,c)}_{s_t^{(c_1)}, s_{t+1}^{(c)}} \tag{1.2}$$

*where $1 \leq s \leq m_c$, $h^{(c_1,c)}_{s_1,s} = d^{(c_1,c)} \cdot a^{(c_1,c)}_{s_1,s}$, $\sum_c d^{(c_1,c)} = 1$, and $\sum_{j=1}^{m_c} a^{(c_1,c)}_{i,j} = 1$. The **observations** $\vec{Y}_t = \{Y_t^{(1)}, \cdots, Y_t^{(C)}\}$ are coupled with the latent states $\vec{S}_t = \{S_t^{(1)}, \cdots, S_t^{(C)}\}$ through a memoryless channel:*

$$Y_t^{(c)} \sim P(Y_t^{(c)} | S_t^{(c)}) \tag{1.3}$$

Several comments concerning this definition are listed below.

- The restriction$\sum_c d^{(c_1,c)} = 1$, and $\sum_{j=1}^{m_c} a_{i,j}^{(c_1,c)} = 1$ in the definition is necessary to guarantee that the *influence* of any state $S_t^{(c)}$ at time t on the states $S_{t+1}^{(c_1)}$ at time $t+1$ sum up to 1. The definition also implicitly gives the formula for $P(S_{t+1}^{(c_1)})$ as a function of $\left(P(S_t^{(c)})\right)^{1\le c\le C}$:

$$\begin{aligned}
& P(S_{t+1}^{(c)} = s_{t+1}^{(c)}) \\
= & \sum_{s_t^{(1)}\cdots s_t^{(C)}} P(S_{t+1}^{(c)} = s_{t+1}^{(c)} | S_t^{(1)}\cdots S_t^{(C)} = s_t^{(1)}\cdots s_t^{(C)}) \cdot P(S_t^{(1)}\cdots S_t^{(C)} = s_t^{(1)}\cdots s_t^{(C)}) \\
= & \sum_{s_t^{(1)}\cdots s_t^{(C)}} \left(\sum_{c1} d^{(c1,c)} a^{(c1,c)}_{s_t^{(c1)} s_{t+1}^{(c)}} \right) \cdot P(S_t^{(1)}\cdots S_t^{(C)} = s_t^{(1)}\cdots s_t^{(C)}) \\
= & \sum_{c1}\sum_{s_t^{(c1)}} d^{(c1,c)} \left(a^{(c1,c)}_{s_t^{(c1)} s_{t+1}^{(c)}} \cdot \sum_{\text{fix } s_t^{(c1)}} P(S_t^{(1)}\cdots S_t^{(C)} = s_t^{(1)}\cdots s_t^{(C)}) \right) \\
= & \sum_{c1}\sum_{s_t^{(c1)}} d^{(c1,c)} a^{(c1,c)}_{s_t^{(c1)} s_{t+1}^{(c)}} \cdot P(s_t^{(c1)})
\end{aligned}$$

  In the same way, any row of the state transition matrix of a hidden Markov process sums up to one (equation 2), and the state transition matrix also implicitly gives the formula for $P(S_{t+1})$ as a function of $P(S_t)$.

- For a latent structure influence process, we can characterize its state transition with an **influence matrix**:

$$\begin{aligned}
A^{(c_1,c_2)} &= \begin{pmatrix} a_{1,1}^{(c_1,c_2)} & \cdots & a_{1,m_{c_2}}^{(c_1,c_2)} \\ \vdots & \ddots & \vdots \\ a_{m_{c_1},1}^{(c_1,c_2)} & \cdots & a_{m_{c_1},m_{c_2}}^{(c_1,c_2)} \end{pmatrix} \\
H &= \begin{pmatrix} d^{(1,1)}A^{(1,1)} & \cdots & d^{(1,C)}A^{(1,C)} \\ \vdots & \ddots & \vdots \\ d^{(C,1)}A^{(C,1)} & \cdots & d^{(C,C)}A^{(1,C)} \end{pmatrix} \\
& \left(P(S_{t+1}^{(1)}=1)\cdots P(S_{t+1}^{(1)}=m_1)\cdots P(S_{t+1}^{(C)}=1)\cdots P(S_{t+1}^{(C)}=m_C)\right) \\
&= \left(P(S_t^{(1)}=1)\cdots P(S_t^{(1)}=m_1)\cdots P(S_t^{(C)}=1)\cdots P(S_t^{(C)}=m_C)\right)\cdot H
\end{aligned}$$

  In comparison, for a hidden Markov model, we have

$$(P(S_{t+1}=1)\cdots P(S_{t+1}=m)) = (P(S_t=1)\cdots P(S_t=m))\cdot A$$

- *The definition does not completely characterize a latent structure influence process*! The definition gives the time-update formula to estimate $P(S_{t+1}^{(c)}|\vec{y}_{1\le t_1\le t})$ from $P(S_t^{(c1)}|\vec{y}_{1\le t_1\le t})$. However, it says nothing about the measure-update formula that incorporates new evidence into the latent state estimates $P(S_{t+1}^{(c)}|\vec{y}_{1\le t_1\le t+1})$. In chapter 2, we will consider the latent structure influence process with two different ways to incorporate new evidence.

The latent structure influence process is used to fit given observation sequences involving human-related behaviors. The influence $d^{(c_1,c_2)}$ reflects how one sequence c_1 agrees with the other c_2. The tasks involved with a latent structure influence model are generally

- latent state inference, i.e., to infer the latent states $\left(S_t^{(c)}\right)$ from the observations $\left(Y_t^{(c)}\right)$ and model parameters $\left(\pi_{s^{(c)}}^{(c)}\right)$, $\left(d^{(c_1,c_2)}\right)$, $\left(a_{s^{(c_1)}s^{(c_2)}}^{(c_1,c_2)}\right)$, and $\left(P(Y_t^{(c)}|S_t^{(c)})\right)$.
- parameter estimation, i.e., to infer the parameters given the latent states and the observations.
- joint latent state inference and parameter estimation. In this case, we are only given the observations, and we are required to find out the parameters, as well as the latent states, that best fit the observations.

To conclude this chapter, we point out that the human-related behaviors have normally strongly clustered/polarized perspectives. The latent structure influence process models these behaviors by emulating the interaction of a group of humans/agents, and computing how different perspectives of the human-related behaviors coincide with each other. When influence modeling is applied to a general complex stochastic processes, there are pros and cons concerning whether it outperforms the other models. In the following chapters, we will derive the algorithms of the latent structure influence process, and inspect its performance in modeling various data sets.

Chapter 2

Inference algorithms for the influence model

When we model the dynamics of a multi-agent stochastic process, we often need to cope with a large probability space, whose states are comprised of the states for the individual agents. This fact has the consequence that we need an exploding number of training samples and need to fix an exploding number of parameters. This is in general not necessary, since the behaviors of any two agents are normally either in agreement with other, or totally unrelated. The latent structure influence process fits such a multi-agent process by emulating how a group of agents influence each other. In this chapter, we present two different ways to incorporate new evidence into the latent states of a latent structure influence process, and derive the inference algorithms for the influence modeling.

2.1 The Linear Approach to the Influence Modeling

The influence model is a tractable approximation of the intractable hidden Markov modeling of multiple interacting dynamic processes. While the number of states for the hidden Markov model is the multiplication of the number of states for individual processes, the number of states for the corresponding influence model is the summation of the number of states for individual processes. The influence model attains this tractability by linearly combining the contributions of the latent state distributions of individual processes at time t to get the latent state distributions of individual processes at time $t+1$.

To illustrate the difficulties of the hidden Markov modeling of a system of multiple interacting processes, and to motivate the influence model approximation, let us consider a system of C interacting processes. In this system, the latent state for process c at time t is denoted as $s_t^{(c)}$ and has a multinomial distribution over $\{1, \cdots, m_c\}$. The latent state for the whole system at time t is denoted as $s_t = s_t^{(1)} \cdots s_t^{(C)}$ and has a multivariate multinomial distribution over $\{\underbrace{1 \cdots 1}_{C}, \cdots, \underbrace{m_1 \cdots m_C}_{C}\}$. The state transition matrix G propagates the system latent state distribution vector $(P(s_t))$ at time t to the system latent state distribution vector $(p(s_{t+1}))$ at time $t+1$:

$$\begin{aligned} (P(s_t)) &\stackrel{\triangle}{=} (P(s_t = \underbrace{1 \cdots 1}_{C}) \cdots P(s_t = \underbrace{m_1 \cdots m_C}_{C})) \\ (P(s_{t+1})) &= (P(s_t)) \cdot G \\ (P(s_{t=1})) &= \pi \end{aligned}$$

The hidden Markov modeling of the whole system has the "curse of dimensionality" and needs to be regularized, since the size of G (which is $\prod_c m_c$) grows exponentially with the number of dynamic processes in the system, while the size of training data only grow polynomially with the number of dynamic

processes. To cope with this issue, we introduce the system event matrix (B) and its Penrose-Moore pseudoinverse (B^+) , and compute in a space related to the system's influence matrix $H = B^+ \cdot G \cdot B$, which is logarithmically smaller, and preserves the eigen-structure of G. In doing so, we confine ourselves to a much restricted space $G_0 = B \cdot H \cdot B^+ = BB^+ \cdot G \cdot BB^+$, whose expressive complexity grows 2^{nd}-order polynomial with the number of processes in the system. The event matrix for the system under discussing has $\prod_c m_c$ number of rows and $\sum_c m_c$ number of columns. It can be constructed in two steps: (1) sorting all possible values the system latent state can take, and (2) filling the rows of the event matrix sequentially by the "one hot" encodings of the corresponding latent state values. The event matrix transforms the system latent state distribution vector $(P(s_t))$ to the system's marginal latent state distribution vector $\left(P(s_t^{(c)})\right)$:

$$\begin{aligned}
\left(P(s_t^{(c)})\right) &\triangleq (\underbrace{P(s_t^{(1)}=1),\cdots,P(s_t^{(1)}=m_1)}_{m_1},\cdots, \\
&\quad \underbrace{P(s_t^{(C)}=1),\cdots,P(s_t^{(C)}=m_C)}_{m_C}) \\
\left(P(s_t^{(c)})\right) &= (P(s_t)) \cdot B \\
H &= B^+ \cdot G \cdot B
\end{aligned}$$

By introducing the above approximation, the latent state propagation from time t to time $t+1$ becomes,

$$\begin{aligned}
(P(s_{t+1})) &= (P(s_t)) \cdot BB^+ \cdot G \cdot BB^+ \\
(P(s_{t=1})) &= \pi \cdot BB^+ \\
\left(P(s_{t+1}^{(c)})\right) &= (P(s_{t+1})) \cdot B \\
&= (P(s_t)) \cdot BB^+ \cdot G \cdot BB^+ \cdot B \\
&= \left(P(s_t^{(c)})\right) \cdot H \\
\left(P(s_{t=1}^{(c)})\right) &= \pi \cdot B
\end{aligned}$$

The connection between $(P(s_t))$ and $\left(P(s_t^{(c)})\right)$ by the event matrix B, and the connection between $\left(P(s_{t+1}^{(c)})\right)$ and $\left(P(s_t^{(c)})\right)$ by the influence matrix H is shown in Figure 2.1. The influence matrix $H = B^+ \cdot G \cdot B$ can always be expressed as the Kronecker product of a transposed stochastic matrix D with a collection of stochastic matrices $\left(A^{(c_1,c_2)}\right)$:

$$H = \begin{pmatrix} d_{1,1} \cdot A^{(1,1)} & \cdots & d_{1,C} \cdot A^{(1,C)} \\ \vdots & \ddots & \vdots \\ d_{C,1} \cdot A^{(C,1)} & \cdots & d_{C,C} \cdot A^{(C,C)} \end{pmatrix}$$

In terms of this property, the marginal state distribution for process c_2 at time $t+1$ is a weighted sum of the "influences" of the marginal state distributions for processes $1 \le c_1 \le C$ at time t, where the influence from process c_1 to process c_2 is computed by right-multiplying the marginal latent state distribution of c_1 at time t with the stochastic matrix $A^{(c_1,c_2)}$.

In both the hidden Markov modeling of system dynamics and its latent structure influence model approximation, the observation $o_t^{(c)}$ of a process c at time t probabilistically depends and only depends on the

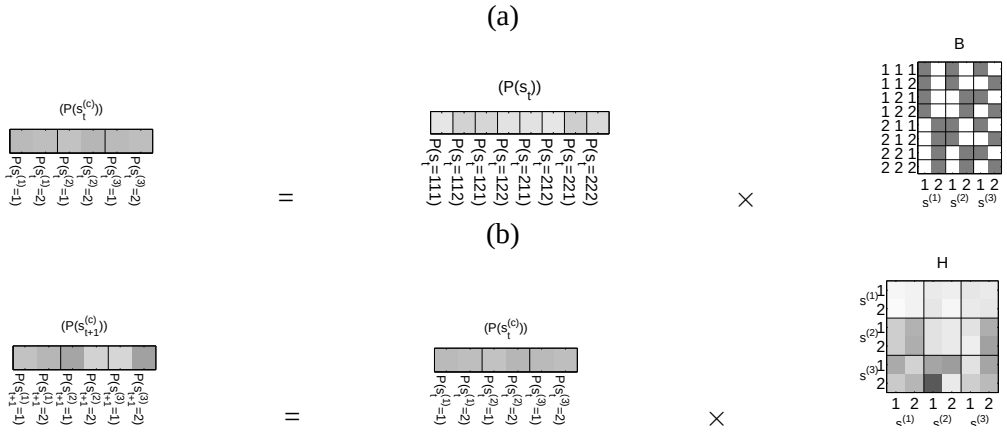

Figure 2.1: (a) The event matrix B produces marginal distributions from joint distributions. (b) The influence matrix H linearly combines the marginal latent state distribution at time t to generate the marginal latent state distribution at time $t+1$.

latent state $s_t^{(c)}$ of process c at time t.

$$\begin{aligned} P(o_t|s_t) &= \prod_c P(o_t^{(c)}|s_t^{(c)}) \\ (P(o_t|s_t)) &\stackrel{\triangle}{=} (P(o_t|s_t = \underbrace{1\cdots 1}_{C})\cdots \\ &\qquad P(o_t|s_t = \underbrace{m_1\cdots m_C}_{C})) \end{aligned}$$

The observations $o_t^{(c)}$ for process c at time t can be either multinomial or Gaussian. When the observations for process c are multinomial, we use n_c to represent the number of possible observation symbols: $o_t^{(c)} \in [1 \dots n_c]$. We define $B^{(c)} = (b_{i,j}^{(c)})$, where $b_{i,j}^{(c)} = p(o^{(c)} = j|s^{(c)} = i)$, as the observation matrix. When the observations for process c are Gaussian, we use n_c to represent the dimensionality of the m_c number of Gaussian distributions corresponding to each latent state $s^{(c)} \in [1 \dots m_c]$: $o^{(c)} \sim \mathcal{N}(\mu_{s^{(c)}}^{(c)}, \sigma_{s^{(c)}}^{(c)})$.

2.1.1 The marginalizing operator B and its inverse B^+

In this section, we formulate the marginalizing operator that maps a joint probability mass function to several marginal probability mass functions. We also derive the best linear estimator (the linear inverse marginalizing operator) that maps several marginal probability mass functions to a joint probability mass function. These two operators are essential to this thesis, since one task of this thesis is to get a good estimate of a joint probability mass function by operate only on the marginal probability mass functions that are logarithmically smaller.

Before we proceed formally, let us image how the marginalizing operator and the (linear) inverse marginalizing operator look like. The marginalization operator is linear, since we only need to sum over the joint probability mass function of those joint states that has a particular marginal state realization to get the marginal probability mass function of this marginal state realization. For example, if we want to know the probability that a power plant fails in a network of power plants, we can add up all probabilities of the network that says this particular power plant fails.

We normally cannot recover the exact joint probability mass function from several marginal probability functions, since the joint probability mass function contains more information than the marginal probability

mass functions. To be more specific, the information that relates different marginal probability mass functions is not given by a list of probability mass functions. Intuitively, in order to estimate the joint probability mass function by using a linear combination of the marginal probability mass functions, we would assume that all marginal probabilities have equal contribution, and that the joint probability mass function for a joint state is proportional to the sum of the marginal probability mass functions of the constituent states. For example, if we want to know the probability that all power plants in a network are normal, we can assume that each power plant has equal contribution to the fact of their joint state. As a result, we can first add up the probabilities that the individual power plants are normal, then normalize the sum to estimate the joint probability. This intuition is correct, as we will show below.

The rest of this subsection proceeds in the following way. First, we define the marginalizing operator formally. Then, we derive the linear inverse marginalizing operator, and explain the mathematical intuition. Based on our knowledge of the marginalizing operator and its inverse, we will discuss several matrices: $B^{+}GB$ and BHB^{+}. The matrix B is a marginalizing operator, B^{+} and B^{T} are the pseudoinverse and transpose of B, respectively. The matrix G is a Markov matrix corresponding to a joint state $S\left(\begin{pmatrix} S^{(1)} & \cdots & S^{(C)} \end{pmatrix}\right)$, and the matrix H is an influence matrix corresponding to the marginal states $S^{(1)}, \cdots, S^{(C)}$. We will show that those matrix operations map between influence matrices and Markov matrices. The computations have intuitive interpretations. The analysis is important, since we need to search for those matrices to maximize some likelihood functions.

Let us begin with an example of the matrix B.

Example 3. *Let us imagine that we have a network of four power plants numbered 1, 2, 3, 4 respectively. Each power plant takes two states: normal and failed, and the state of the network is a 4-tuple of the states for the individual power plants. By this description, the (joint) states of the network form a probability space $\{\Omega, P\}$, and the (marginal) states for individual power plants form probability spaces $\{\Omega^{(c)}, P^{(c)}\}$, where $c = 1, 2, 3, 4$. We use the random variables $S^{(c)}$ to describe the probability spaces $\{\Omega^{(c)}, P^{(c)}\}$, where*

$$S^{(c)} = \begin{cases} 1 & \text{power plant c normal} \\ 2 & \text{power plant c failed} \end{cases}$$

. We also use the row vectors

$$\begin{aligned} \left(P^{(c)}(S^{(c)})\right)_{S^{(c)}=\{1,2\}} &= \begin{pmatrix} P^{(c)}(S^{(c)}=1) & P(S^{(c)}=2) \end{pmatrix} \\ \left(P^{(c)}(S^{(c)})\right)_{S^{(c)}=\{1,2\}}^{c=\{1,2,3,4\}} &= \begin{pmatrix} P^{(1)}(S^{(1)}=1) & P^{(1)}(S^{(1)}=2) & P^{(2)}(S^{(2)}=1) & P^{(2)}(S^{(2)}=2) \\ & & P^{(3)}(S^{(3)}=1) & P^{(3)}(S^{(3)}=2) & P^{(4)}(S^{(4)}=1) & P^{(4)}(S^{(4)}=2) \end{pmatrix} \end{aligned}$$

to denote the probability mass function for one random variable $S^{(c)}$, and a concatenation of the probability mass functions for random variables, respectively. In the same fashion, we use the random vector

$$\left(S^{(c)}\right)_{1\leq c\leq 4} = \begin{pmatrix} S^{(1)} & S^{(2)} & S^{(3)} & S^{(4)} \end{pmatrix}$$

to describe the probability space $\{\Omega, P\}$*, and we use the row vector*

$$\begin{aligned}\left(P\left(\left(S^{(c)}\right)_{1\leq c\leq 4}\right)\right) &= \left(\begin{array}{cc} P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&1&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&1&2\end{array}\right)\right)\\ P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&2&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&2&2\end{array}\right)\right)\\ P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&1&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&1&2\end{array}\right)\right)\\ P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&2&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&1&2&2\end{array}\right)\right)\\ P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&2&1&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&2&1&2\end{array}\right)\right)\\ P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&2&2&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}1&2&2&2\end{array}\right)\right)\\ P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}2&2&1&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}2&2&1&2\end{array}\right)\right)\\ P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}2&2&2&1\end{array}\right)\right) & P\left(\left(S^{(c)}\right)=\left(\begin{array}{cccc}2&2&2&2\end{array}\right)\right)\end{array}\right)\end{aligned}$$

to describe the probability mass function for the random vector $\left(S^{(c)}\right)_{1\leq c\leq 4}$*. Alternatively, we use the random function*

$$S\left(\left(s^{(c)}\right)_{1\leq c\leq 4}\right) = s^{(1)}+2\cdot s^{(2)}+4\cdot s^{(3)}+8\cdot s^{(4)}$$

to describe the probability space $\{\Omega, P\}$*, and we have*

$$P\left(S\left(\left(S^{(c)}\right)_{1\leq c\leq 4}\right)\right) = P\left(\left(S^{(c)}\right)_{1\leq c\leq 4}\right)$$

A marginalizing operator that maps the joint probability mass function $\left(P\left(\left(S^{(c)}\right)_{1\leq c\leq 4}\right)\right)$ *to the marginal probability mass functions* $\left(P\left(S^{(c)}\right)\right)_{S^{(c)}=\{1,2\}}^{c=\{1,2,3,4\}}$ *is*

$$B = \left(\begin{array}{cccccccc}1&0&1&0&1&0&1&0\\1&0&1&0&1&0&0&1\\1&0&1&0&0&1&1&0\\1&0&1&0&0&1&0&1\\1&0&0&1&1&0&1&0\\1&0&0&1&1&0&0&1\\1&0&0&1&0&1&1&0\\1&0&0&1&0&1&0&1\\0&1&1&0&1&0&1&0\\0&1&1&0&1&0&0&1\\0&1&1&0&0&1&1&0\\0&1&1&0&0&1&0&1\\0&1&0&1&1&0&1&0\\0&1&0&1&1&0&0&1\\0&1&0&1&0&1&1&0\\0&1&0&1&0&1&0&1\end{array}\right)$$

$$\left(P\left(S^{(c)}\right)\right)_{S^{(c)}=\{1,2\}}^{c=\{1,2,3,4\}} = \left(P\left(\left(S^{(c)}\right)_{1\leq c\leq 4}\right)\right)\cdot B$$

This marginalizing operator gets $P^{(c)}\left(s^{(c)}\right)$ *by summing over all joint probability mass functions*

$$P\left(\left(\begin{array}{cccc}S^{(1)}&S^{(2)}&S^{(3)}&S^{(4)}\end{array}\right)\right)$$

for all states such that $S^{(c)}=s^{(c)}$

Let us suppose we have C random variables $S^{(1)}, S^{(2)}, \cdots, S^{(C)}$. Each random variable $S^{(c)}$, where $1 \leq c \leq C$, can take m_c number of states $S^{(c)} = \{1, 2, \cdots, m_c\}$. Let the random function

$$S\left(\left(s^{(c)}\right)_{1\leq c\leq C}\right) \quad = \quad \sum_{c=1}^{C} s^{(c)} \cdot \left(\prod_{c1=1}^{c-1} m_{c1}\right)$$

be the joint state. The marginalizing operator is a matrix that maps from

$$\left(P\left(S\right)\right)_{S=1..\prod m_c}$$

to

$$\left(P\left(S^{(c)}\right)\right)_{S^{(c)}=1..m_c}^{c=1..C}$$

.

Definition 2. *The **marginalizing operator***

$$B\left((m_c)_{1\leq c\leq C}\right) \quad = \quad B\left(\left(\begin{array}{cccc} m_1 & m_2 & \cdots & m_C \end{array}\right)\right)$$

is a $\left(\prod_{c=1}^{C} m_c\right) \times \left(\sum_{c=1}^{C} m_c\right)$ *matrix, where*

$$\begin{aligned} B_{i,j} \quad \leftarrow \quad & c = \texttt{argmax}_{c1} i - \sum_{c2=1}^{c1} m_c > 0, \\ & s = j - \sum_{c1=1}^{c} m_c \\ & B_{i,j} = \begin{cases} 1 & ,\texttt{if } s \equiv \texttt{floor}\left(\frac{i}{\prod_{c1=1}^{c} m_c}\right) \texttt{ mod } m_{c+1} \\ 0 & \texttt{otherwise} \end{cases} \end{aligned}$$

The best estimator B^+ that maps the marginal pmf's to the joint pmf is the "inverse" of the marginalizing matrix B. Since the matrix B is not full-ranked, its inverse does not exist. As a result, we attempt to find the best matrix that takes the effect of inversion (Moore-Penrose pseudoinverse). Let us take a digression and inspect the Bayesian interpretation of pseudoinverse before we give the inverse of the marginalizing operator.

The problem involved with the pseudoinverse operator is solving linear systems $A \cdot \vec{x} = \vec{b}$, where we know the matrix A and the vector $\vec{b}$, and need to find the vector $\vec{x}$ to satisfy this equation. When the matrix A has full rank and is not ill-conditioned, i.e., when $\|A\| \cdot \|A^{-1}\|$ is relatively small, the systems $A \cdot \vec{x} = \vec{b}$ have unique solutions and are not interesting to us. In situations when linear systems do not have numerically stable solutions, do not have solutions, or do not have unique solutions, we would think that the information provided by the linear systems are not enough, and provide more information to the linear system in order to solve it. One way to provide information is to assume that $\vec{x}$ and $\vec{b}$ have Gaussian random noises: $\vec{x} \sim \mathcal{N}(0, \sigma_x^2 I)$, and $\vec{b} \sim \mathcal{N}(0, \sigma_b^2 \cdot I)$. The best linear estimator (in the sense that it has the minimum mean-square error $\|\vec{b} - A \cdot \vec{x}\|^2 + (\sigma_b^2/\sigma_x^2) \cdot \|\vec{x}\|^2$) is thus $\vec{x} = \left(A^{\mathtt{T}}A + (\sigma_b^2/\sigma_x^2) \cdot I\right)^{-1} A^{\mathtt{T}}\vec{b} = \left(A^{\mathtt{T}}A + \alpha^2 \cdot I\right)^{-1} A^{\mathtt{T}}\vec{b}$. The Moore-Penrose pseudoinverse of matrix A is defined as $A^+ = \lim_{\alpha^2 \to 0}(A^{\mathtt{T}}A + \alpha^2 I)^{-1}A^{\mathtt{T}} = \lim_{\alpha^2 \to 0} A^{\mathtt{T}}(A^{\mathtt{T}}A + \alpha^2 I)^{-1}$.

Definition 3. *The **pseudoinverse** A^+ of a matrix A is the unique matrix that satisfies the following criteria:*

1. $AA^+A = A$
2. $A^+AA^+ = A^+$, *A^+ is a weak inverse for the multiplicative semi-group*
3. $(AA^+)^* = AA^+$, *AA^+ is Hermitian*
4. $(A^+A)^* = A^+A$, *A^+A is Hermitian*

Alternatively, A^+ can be defined by the following limiting process:

$$A^+ = \lim_{\alpha^2 \to 0} (A^{\mathrm{T}}A + \alpha^2 I)^{-1} A^{\mathrm{T}} = \lim_{\alpha^2 \to 0} A^{\mathrm{T}} (A^{\mathrm{T}}A + \alpha^2 I)^{-1}$$

The pseudoinverse B^+ of a marginalizing operator $B\left(\left(\begin{array}{cccc} m_1 & m_2 & \cdots & m_C \end{array}\right)\right)$ has a simple form: $B^+ = c_1 \cdot B^{\mathrm{T}} + c_2$, where c_1 and c_2 are constants determined by $m_1, \cdots, m_C$.

Theorem 1. *Given a system of C interacting processes, and m_c number of possible values for the latent state $s^{(c)}$ for process $1 \leq c \leq C$, we have,*

$$\begin{aligned}
B^+ &= \frac{1}{\sum_c \prod_{k \neq c} m_k} \cdot (D_1 \cdot B^{\mathrm{T}} + D_2 \cdot 1_{\sum_c m_c \times \prod_c m_c}) \\
D_1 &= \mathrm{diag}[\underbrace{\sum_k \frac{m_1}{m_k}, \cdots, \sum_k \frac{m_1}{m_k}}_{m_1}, \cdots, \underbrace{\sum_k \frac{m_C}{m_k}, \cdots, \sum_k \frac{m_C}{m_k}}_{m_C}] \\
D_2 &= \mathrm{diag}[\underbrace{-\sum_{k \neq 1} \frac{1}{m_k}, \cdots, -\sum_{k \neq 1} \frac{1}{m_k}}_{m_1}, \cdots, \underbrace{-\sum_{k \neq C} \frac{1}{m_k}, \cdots, -\sum_{k \neq C} \frac{1}{m_k}}_{m_C}]
\end{aligned}$$

where B^{T} represents the transpose of B.

Proof. In order to prove that B^+ is the pseudoinverse of B, we need to show four things: $BB^+B = B$, $B^+BB^+ = B^+$, $(BB^+)^{\mathrm{T}} = BB^+$, and $(B^+B)^{\mathrm{T}} = B^+B$. Below, we use $A[c_1, c_2, i, j]$ to represent the element at the i^{th} row, j^{th} column in the sub-matrix index by (c_1, c_2) of the matrix A (whose size is $\sum m_c \times \sum m_c$).

$$(B^{\mathrm{T}}B)[c_1, c_2, i, j] = \begin{cases} \prod_{k \neq c_1} m_k & , c_1 = c_2, i = j \\ 0 & , c_1 = c_2, i \neq j \\ \prod_{k \neq c_1, c_2} m_k & , c_1 \neq c_2 \end{cases}$$

$$\begin{aligned}
&(\sum_c \prod_{k \neq c} m_k) \cdot (B^+B)[c_1, c_2, i, j] \\
= &\begin{cases} \sum_c \prod_{k \neq c} m_k - \sum_{c \neq c_1} \prod_{k \neq c, c_1} m_k & c_1 = c_2, i = j \\ -\sum_{c \neq c_1} \prod_{k \neq c, c_1} m_k & c_1 = c_2, i \neq j \\ \prod_{k \neq c_1, c_2} m_k & c_1 \neq c_2 \end{cases}
\end{aligned}$$

It follows that $(B^+B)^{\mathrm{T}} = B^+B$, $BB^+B = B$, and $B^+BB^+ = B^+$.

To show $(BB^+)^{\mathrm{T}} = BB^+$, we notice

$$\begin{aligned}
& \frac{1}{\sum_c \prod_{k\neq c} m_k} \cdot (BB^+)^{\mathrm{T}} \\
= \; & (BD_1B^{\mathrm{T}} + BD_2 1_{\sum_c m_c \times \prod_c m_c})^{\mathrm{T}} \\
= \; & BD_1B^{\mathrm{T}} + 1_{\prod_c m_c \times \sum_c m_c} D_2 B \\
= \; & BD_1B^{\mathrm{T}} - (\sum_c \frac{1}{m_c}) \cdot 1_{\prod_c m_c \times \sum_c m_c} \\
= \; & BD_1B^{\mathrm{T}} + BD_2 1_{\sum_c m_c \times \prod_c m_c} \\
= \; & BB^+
\end{aligned}$$

□

The computation of B^+ has a very intuitive interpretation. Suppose $S^{(1)}, \cdots, S^{(C)}$ are C random variables. Each variable takes finite number of values: $S^{(c)} = \{1, \cdots m_c\}$, where $1 \leq c \leq C$. The random function $S\left(\begin{pmatrix} S^{(1)} & \cdots & S^{(C)} \end{pmatrix}\right)$ "encodes" the individual random variables. Let us look at how B^+ maps a joint probability mass function to several marginal probability mass functions. Let us first naively sum up all marginal probability distributions $P(S^{(c)})$, where $1 \leq c \leq C$, to get the joint probability distribution $P\left(S\left(\begin{pmatrix} S^{(1)} & \cdots & S^{(C)} \end{pmatrix}\right)\right)$. In other words, we use the following formula

$$\left(\tilde{P}\left(S\left(\begin{pmatrix} S^{(1)} & \cdots & S^{(C)} \end{pmatrix}\right)\right)\right) \;=\; \left(P\left(S\left(S^{(c)}\right)\right)\right)_{1\leq S^{(c)}\leq m_c}^{1\leq c\leq C} \cdot B^T$$

In the formula for $\tilde{P}$, the marginal probability mass function corresponding to $S^{(c)} = s^{(c)}$ is summed up for $\prod_{c1\neq c} m_{c1}$ times in the joint probability mass function for all possible realizations.

$$\tilde{P}\left(S\left(\begin{pmatrix} S^{(1)} & \cdots & S^{(c)} = s^{(c)} & \cdots & S^{(C)} \end{pmatrix}\right)\right)$$

As a result, we need cancel this effect by scaling the contribution of $S^{(c)}$ by $\frac{P(S^{(c)})}{\prod_{c1\neq c} m_{c1}}$. This operation is reflected by the matrix D_1. The computation of $\tilde{P}$ is also biased, since the information $\sum_{s^{(c)}} P(s^{(c)}) = 1$ is not useful for $\tilde{P}$. As a result, we need to cancel the effect by subtraction $\frac{P(S^{(c)})}{\prod_{c1\neq c} m_{c1}}$ by

$$\frac{\sum_{k\neq c} \frac{1}{m_k}}{(\prod_k m_k)\left(\sum_k \frac{1}{m_k}\right)}$$

The operation of removing the bias is reflected by the matrix D_2. To further illustrate the effect of removing the bias, we can compare the joint probability mass function P via $\frac{1}{\prod_c \sum_{k\neq c} m_k}\left(D_1 \cdot B^{\mathtt{T}} + D_2 \cdot \mathtt{1}\right)$ and the probability function $\tilde{P}$ via $\frac{1}{\prod_c \sum_{k\neq c} m_k} D_1 \cdot B^{\mathtt{T}}$. P might have negative values, while $\tilde{P}$ is always positive. Let us suppose that P is positive thus a true probability mass function. In this case, we get $\tilde{P}$ by adding a constant to all values of P. From our knowledge of information theory, we lose information from P to $\tilde{P}$. On the other hand, the matrix $D_2 \cdot \mathtt{1}$ tries to extract more information from the marginal distributions $P(S^{(c)})$, where $1 \leq c \leq C$.

With the knowledge of B and B^+, we can compute the analytic form of B^+GB and BHB^+, where G is a Markov matrix, and H is an influence matrix. We need to point out that B^+GB (with restriction on G) is an influence matrices, while BHB^+ is a Markov matrices.

Theorem 2. *Let $1 \leq S^{(c)} \leq m_c$, where $1 \leq c \leq C$, are random variables, $S\left(\left(S^{(c)}\right)^{1\leq c\leq C}\right)$ be the random function, $B\left(\left(\begin{array}{ccc} m_1 & \cdots & m_C \end{array}\right)\right)$, B^+, H, G be respectively the marginalizing matrix, the inverse of the marginalizing matrix, a influence matrix, and a Markov matrix related to $S^{(c)}$, where $1 \leq c \leq C$, and S. We have the following relations:*

- *The analytical form of B^+GB. Each row of the (c_1, c_2)-th sub-matrix sum up to $\frac{1/m_{c1}}{\sum_c 1/m_c}$.*

$$\begin{aligned}
\left(B^+GB\right)(c1,c2,i,j) &= \frac{1}{\prod_{c\neq c1} m_c} \cdot \sum_{\substack{s^{(c1)}=i \\ t^{(c2)}=j \\ s^{(c\neq c1)} \\ t^{(c\neq c2)}}} G\left(s^{(1)}\cdots s^{(C)}, t^{(1)}\cdots t^{(C)}\right) - \\
&\quad \frac{\sum_{c\neq c1}\frac{1}{m_c}}{\left(\prod_c m_c\right)\cdot\left(\sum_c \frac{1}{m_c}\right)} \cdot \sum_{\substack{s^{(1\leq c\leq C)} \\ t^{(c2)}=j \\ t^{(c\neq c2)}}} G\left(s^{(1)}\cdots s^{(C)}, t^{(1)}\cdots t^{(C)}\right) \\
\sum_j \left(B^+GB\right)(c1,c2,i,j) &= \frac{1}{m_{c1}} \cdot \frac{1}{\sum_c \frac{1}{m_c}} \\
\sum_{c1}\sum_j \left(B^+GB\right)\left(c1,c2,s^{(c1)},j\right) &= 1
\end{aligned}$$

- *The analytical form related to BHB^+. Each row of BHB^+ sums up to 1.*

$$\begin{aligned}
\left(BHB^+\right)\left(s^{(1)}\cdots s^{(C)}, t^{(1)}\cdots t^{(C)}\right) &= \sum_{\substack{c1,c2 \\ i=s^{(c1)} \\ j=t^{(c2)}}} \left(H(c1,c2,i,j)\cdot\frac{1}{\prod_{c\neq c2} m_c}\right) - \frac{C-1}{\left(\prod_c m_c\right)} \\
\sum_{t^{(1)}\ldots t^{(C)}} \left(BHB^+\right)\left(s^{(1)}\cdots s^{(C)}, t^{(1)}\cdots t^{(C)}\right) &= 1
\end{aligned}$$

- *The constant matrices $1_{(\sum m_c)\times(\sum m_c)}$ and $1_{(\prod m_c)\times(\prod m_c)}$.*

$$\begin{aligned}
B1_{(\sum m_c)\times(\sum m_c)}B^+ &= 1_{(\prod m_c)\times(\prod m_c)} \\
B^+1_{(\prod m_c)\times(\prod m_c)}B &= 1_{(\sum m_c)\times(\sum m_c)}
\end{aligned}$$

Instead of giving a derivation of the above equalities, we trace the computation and describe informally how we make a best linear estimation of the influence matrix corresponding to a Markov matrix, and how we make a best linear estimation of the Markov matrix corresponding to an influence matrix. In order to compute the entry corresponding to $S_1^{(c_1)} = s_1^{(c_1)}$, $S_2^{(c_2)} = s_2^{(c_2)}$ from the matrix B^+GB (i.e., $\left(B^+GB\right)\left(c_1, c_2, s_1^{(c_1)}, s_2^{(c_2)}\right)$), we sum up all entries of G that are compatible with the fact $S_1^{(c_1)} = s_1^{(c_1)}$ and $S_2^{(c_2)} = s_2^{(c_2)}$, and there are $\left(\prod_{c\neq c_1} m_c\right) \times \left(\prod_{c\neq c_2} m_c\right)$ number of summands. Since we sum up $\prod_{c\neq c_1} m_c$ number of items involving c_1, we need to scale this sum by$1/\prod_{c\neq c_1} m_c$. This leads to

$$\frac{1}{\prod_{c\neq c1} m_c} \cdot \sum_{j=1}^{m_{c_2}} \sum_{s^{(c_1)}=i} \sum_{t^{(c_2)}=j} G\left(s^{(1)}\cdots s^{(C)}, t^{(1)}\cdots t^{(C)}\right) = 1$$

Now the rows of $(c1, c2)$-th sub-matrices all sum up to 1. In comparison, we want to extract one arbitrary row from the $(c1, c2)$-th matrix for all $1 \leq c1 \leq C$ and sum those rows to 1. Since the contribution of each $S^{(c1)}$ to $S^{(c2)}$, where $1 \leq c1 \leq C$, is inverse proportional to the number of states m_c, we need to offset the quantity $\frac{1}{\prod_{c\neq c1} m_c} \sum_{s^{(c1)}=i} \sum_{t^{(c2)}=j} G\left(s^{(1)} \cdots s^{(C)}, t^{(1)} \cdots t^{(C)}\right)$ by $\frac{\sum_{c\neq c1} 1/m_c}{\sum_c 1/m_c}$ of the likelihood that $S^{(c2)} = t^{(c2)}$. Notice that the sum of all entries of the Markov matrix G sum up to $\prod_c m_c$

$$\frac{1}{(\prod_c m_c)} \cdot \sum_{s^{(c1)}} \sum_{t^{(c2)}} G\left(s^{(1)} \cdots s^{(C)}, t^{(1)} \cdots t^{(C)}\right) \quad = \quad 1$$

In order to compute the entry of $(BHB^+)\left(\left(s^{(c)}\right)^{1\leq c\leq C}, \left(t^{(c)}\right)^{1\leq c\leq C}\right)$, we sum up all entries in H compatible with $s^{(c)}$, and $t^{(c)}$. There are $(\sum m_c) \times (\sum m_c)$ summands in total. Since each $t^{(c2)}$ contributes to BHB^+ for $\prod_{c\neq c2} m_c$ number of times as $H(s^{(c1)}, t^{(c2)})$ for some $s^{(c1)}$, we need to scale the contribution of $t^{(c2)}$ down by $\frac{1}{\prod_{c\neq c_2} m_c}$. This leads to

$$\sum_{\substack{c1,c2 \\ i=s^{(c1)} \\ j=t^{(c2)} \\ t^{(1)}\cdots t^{(C)}}} \left(H\left(c1, c2, i, j\right) \cdot \frac{1}{\prod_{c\neq c2} m_c}\right) \quad = \quad C$$

Each $S^{(c)}$ contributes a quantity of 1 to each row of BHB^+ , and all of the C marginal random variables $S^{(c)}$, where $1 \leq c \leq C$, have a total contribution of C to each row of BHB^+. As a result, we want to down-shift the values obtained by $C - 1$ and distribute this quantity among all $\prod_c m_c$ entries. This gives the offset.

We compute below the matrices B^+, and BHB^+ related to the network of four power plants.

Example 4. *Let us compute the pseudoinverse B^+ of the marginalizing operator*

$$B\left(\begin{pmatrix} m_1 & m_2 & m_3 & m_4 \end{pmatrix}\right) \quad = \quad B\left(\begin{pmatrix} 2 & 2 & 2 & 2 \end{pmatrix}\right)$$

for the network of four power plants (Example 3) as well as other interesting objects.

The constants $\frac{1}{\sum_c \prod_{k\neq c} m_k}$, $\sum_k \frac{m_c}{m_k}$, and $-\sum_{k\neq c} \frac{1}{m_k}$ are computed as follows:

$$\begin{aligned}
\frac{1}{\sum_c \prod_{k\neq c} m_k} &= \frac{1}{m_2 \times m_3 \times m_4 + m_1 \times m_3 \times m_4 + m_1 \times m_2 \times m_4 + m_1 \times m_2 \times m_3} \\
&= \frac{1}{2 \times 2 \times 2 + 2 \times 2 \times 2 + 2 \times 2 \times 2 + 2 \times 2 \times 2} \\
&= \frac{1}{32} \\
\sum_k \frac{m_c}{m_k} &= 4, c = 1, 2, 3, 4 \\
-\sum_{k\neq c} \frac{1}{m_k} &= -\frac{3}{2}, c = 1, 2, 3, 4
\end{aligned}$$

The matrix $\frac{1}{\sum_c \prod_{k \neq c} m_k} \cdot D_1$ *is computed as:*

$$\frac{1}{\sum_c \prod_{k \neq c} m_k} \cdot D_1 = \begin{pmatrix} \frac{1}{8} & & & & & & & \\ & \frac{1}{8} & & & & & & \\ & & \frac{1}{8} & & & & & \\ & & & \frac{1}{8} & & & & \\ & & & & \frac{1}{8} & & & \\ & & & & & \frac{1}{8} & & \\ & & & & & & \frac{1}{8} & \\ & & & & & & & \frac{1}{8} \end{pmatrix}$$

The matrix $\frac{1}{\sum_c \prod_{k \neq c} m_k} \cdot D_2$ *is computed as:*

$$\frac{1}{\sum_c \prod_{k \neq c} m_k} \cdot D_2 = \begin{pmatrix} -\frac{3}{64} & & & & & & & \\ & -\frac{3}{64} & & & & & & \\ & & -\frac{3}{64} & & & & & \\ & & & -\frac{3}{64} & & & & \\ & & & & -\frac{3}{64} & & & \\ & & & & & -\frac{3}{64} & & \\ & & & & & & -\frac{3}{64} & \\ & & & & & & & -\frac{3}{64} \end{pmatrix}$$

The matrix $1_{(\sum_c m_c) \times (\prod_c m_c)}$ *is*

$$1_{(\sum_c m_c) \times (\prod_c m_c)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

The pseudoinverse B^+ *is thus:*

$$\begin{aligned} & B^+ \\ = \; & \frac{1}{\sum_c \prod_{k \neq c} m_k} \cdot \left(D_1 \cdot B^{\mathtt{T}} + D_2 \cdot 1_{(\sum_c m_c) \times (\prod_c m_c)} \right) \\ = \; & \frac{1}{64} \cdot \begin{pmatrix} +5 & +5 & +5 & +5 & +5 & +5 & +5 & +5 & -3 & -3 & -3 & -3 & -3 & -3 & -3 & -3 \\ -3 & -3 & -3 & -3 & -3 & -3 & -3 & -3 & +5 & +5 & +5 & +5 & +5 & +5 & +5 & +5 \\ +5 & +5 & +5 & +5 & -3 & -3 & -3 & -3 & +5 & +5 & +5 & +5 & -3 & -3 & -3 & -3 \\ -3 & -3 & -3 & -3 & +5 & +5 & +5 & +5 & -3 & -3 & -3 & -3 & +5 & +5 & +5 & +5 \\ +5 & +5 & -3 & -3 & +5 & +5 & -3 & -3 & +5 & +5 & -3 & -3 & +5 & +5 & -3 & -3 \\ -3 & -3 & +5 & +5 & -3 & -3 & +5 & +5 & -3 & -3 & +5 & +5 & -3 & -3 & +5 & +5 \\ +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 \\ -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 & -3 & +5 \end{pmatrix} \end{aligned}$$

The constants $\sum_c \prod_{k\neq c} m_k$, $\sum_{c\neq c_1} \prod_{k\neq c,c_1} m_k$, $\prod_{k\neq c_1,c_2} m_k$ for computing B^+B are:

$$\begin{aligned}
\sum_c \prod_{k\neq c} m_k &= m_2\times m_3\times m_4 + m_1\times m_3\times m_4 + m_1\times m_2\times m_4 + m_1\times m_2\times m_3 \\
&= 32 \\
\sum_{c\neq c_1} \prod_{k\neq c,c_1} m_k &= 16, c_1 = 1,2,3,4 \\
\prod_{k\neq c_1,c_2} &= 4, c_1, c_2 = 1,2,3,4
\end{aligned}$$

The matrix B^+B is

$$B^+B = \frac{1}{32}\cdot\begin{pmatrix}
20 & -12 & 4 & 4 & 4 & 4 & 4 & 4 \\
-12 & 20 & 4 & 4 & 4 & 4 & 4 & 4 \\
4 & 4 & 20 & -12 & 4 & 4 & 4 & 4 \\
4 & 4 & -12 & 20 & 4 & 4 & 4 & 4 \\
4 & 4 & 4 & 4 & 20 & -12 & 4 & 4 \\
4 & 4 & 4 & 4 & -12 & 20 & 4 & 4 \\
4 & 4 & 4 & 4 & 4 & 4 & 20 & -12 \\
4 & 4 & 4 & 4 & 4 & 4 & -12 & 20
\end{pmatrix}$$

This is the best matrix we can get from the marginalizing matrix$B$ and any other matrix A that most resembles the identity matrix.

Let matrix $H = \left(h^{(c_1,c_2)}_{s^{(c_1)},s^{(c_2)}}\right)^{c_1,c_2=\{1,2,3,4\}}_{s^{(c_1)},s^{(c_2)}=\{1,2\}}$ be the influence matrix of the network of four power plants. In this matrix, $h^{(c_1,c_2)}_{s^{(c_1)},s^{(c_2)}}$ represents the "influence" from state $s^{(c_1)}$ of power plant c_1 to state $s^{(c_2)}$ of power plant c_2, and $h^{(c_1,c_2)}_{s^{(c_1)},s^{(c_2)}}$ is the $\left(s^{(c_1)}, s^{(c_2)}\right)$-th element in the (c_1,c_2)-th sub matrix of H. Let matrix G be the Markov matrix of the hidden Markov process model of the network, corresponding to the latent structure influence process model. Let the random variable $S\left(\left(S^{(1)}S^{(2)}S^{(3)}S^{(4)}\right)\right)$ be the state of the network, "encoded" by the states of the individual power plants, and

$$S\left(\left(s^{(1)}s^{(2)}s^{(3)}s^{(4)}\right)\right) = 8\cdot s^{(1)} + 8\cdot s^{(2)} + 8\cdot s^{(3)} + 8\cdot s^{(4)}$$

With this notation,

$$\begin{aligned}
& B\cdot H\cdot B^+[S\left(\begin{pmatrix} s_1^{(1)} & \cdots & s_1^{(2)}\end{pmatrix}\right), S\left(\begin{pmatrix} s_2^{(1)} & \cdots & s_2^{(2)}\end{pmatrix}\right)] \\
= & \left(h^{(1,1)}_{s_1^{(1)},s_2^{(1)}} + h^{(2,1)}_{s_1^{(2)},s_2^{(1)}} + h^{(3,1)}_{s_1^{(3)},s_2^{(1)}} + h^{(4,1)}_{s_1^{(4)},s_2^{(1)}}\right)\cdot\frac{1}{m_2\times m_3\times m_4} + \\
& \left(h^{(1,2)}_{s_1^{(1)},s_2^{(2)}} + h^{(2,2)}_{s_1^{(2)},s_2^{(2)}} + h^{(3,2)}_{s_1^{(3)},s_2^{(2)}} + h^{(4,2)}_{s_1^{(4)},s_2^{(2)}}\right)\cdot\frac{1}{m_1\times m_3\times m_4} + \\
& \left(h^{(1,3)}_{s_1^{(1)},s_2^{(3)}} + h^{(2,3)}_{s_1^{(2)},s_2^{(3)}} + h^{(3,3)}_{s_1^{(3)},s_2^{(3)}} + h^{(4,3)}_{s_1^{(4)},s_2^{(3)}}\right)\cdot\frac{1}{m_1\times m_2\times m_4} + \\
& \left(h^{(1,4)}_{s_1^{(1)},s_2^{(4)}} + h^{(2,4)}_{s_1^{(2)},s_2^{(4)}} + h^{(3,4)}_{s_1^{(3)},s_2^{(4)}} + h^{(4,4)}_{s_1^{(4)},s_2^{(4)}}\right)\cdot\frac{1}{m_1\times m_2\times m_3} - \\
& \frac{4-1}{m_1\times m_2\times m_3\times m_4}
\end{aligned}$$

The expression for $B^+GB\left(c_1,c_2,s^{(1)},s^{(2)}\right)$ has too many terms, we do not write an explicit expression here.

2.1.2 Estimation of Forward and Backward Parameters

For hidden Markov models whose latent variable S can take hundreds or even thousands of values, its influence modeling is rather intuitive: We attempt to find another set of latent variables $S^{(c)}$, where $1 \leq c \leq C$, and establish the connection between $S^{(c)}$ and S, as well as the connection between $S^{(c)}$ and the observations Y. The inferences involved with $S^{(c)}$ can be computationally much cheaper and more stable than the inferences involved with S. In this section, we show that if the latent variable S of the original hidden Markov model can be mapped linearly from a set of variables $S^{(c)}$, and if for each random variable $S^{(c)}$, we know the conditional probability $P(Y^{(c)}|S^{(c)})$ for the observation $Y^{(c)}$, then the inferences with $S^{(c)}$ is much cheaper and more stable. In addition, our understanding of the original random variable S can come from the mapping from $S^{(c)}$ to S. This section proceeds in the following order. We first state how the conditional probability of an observation conditioned on the random variable S be mapped to the conditional probability of this observation conditioned on the random variables $S^{(c)}$. Then we give the forward-backward algorithm of the influence modeling based on linear mapping. Last, we state how to connect the probabilities involved with S and the probabilities involved with $S^{(c)}$.

Let us first review the forward-backward algorithm in matrix form, and inspect how we reduce a complex hidden Markov model inference problem involved with a large number of latent states to a much simpler one via the influence modeling. The matrix form of the forward-backward algorithm is the most natural form for this formulation.

In the below, we let $S_t^{(c)}$, where $1 \leq c \leq C$, be random variables, $\vec{S}_t = \left(S_t^{(c)}\right)^{1\leq c\leq C}$ be a random vector, and $S\left(\left(S_t^{(c)}\right)^{1\leq c\leq C}\right)$ be the index of $\left(S_t^{(c)}\right)^{1\leq c\leq C}$. The index random variable S can take $\prod_c m_c$ number of different values. Let $\left\{\left(S_t, \left(Y_t^{(c)}\right)^{1\leq c\leq C}\right)\right\}$ be a hidden Markov process characterized by the initial state distribution $\pi_s = P(S_1 = s)$, the state transition probability $P(S_{t+1}|S_t)$, and the emission probability $P\left(\left(Y_t^{(c)}\right)^{1\leq c\leq C}\big|S_t\right) = \prod_c P(Y_t^{(c)}|S_t^{(c)})$. Given $\left(Y_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}$, the statistical values $\alpha_t(s_t)$, $\beta_t(s_t)$, $\gamma_t(s_t)$, and $\xi_{t\rightarrow t+1}(s_t, s_{t+1})$ are the forward parameters, the backward parameters, the one-slice parameters, and the two-slice parameters respectively:

$$\begin{aligned}
\alpha_t(s_t) &= P\left(s_t, \left(y_{t_1}^{(c)}\right)_{1\leq t_1\leq t}^{1\leq c\leq C}\right) \\
\beta_t(s_t) &= P\left(\left(y_{t_1}^{(c)}\right)_{t+1\leq t_1\leq T}^{1\leq c\leq C}\big|s_t\right) \\
\gamma_t(s_t) &= P\left(s_t, \left(y_{t_1}^{(c)}\right)_{1\leq t_1\leq T}^{1\leq c\leq C}\right) \\
\xi_{t\rightarrow t+1}(s_t, s_{t+1}) &= P\left(s_t, s_{t+1}, \left(y_{t_1}^{(c)}\right)_{1\leq t_1\leq T}^{1\leq c\leq C}\right)
\end{aligned}$$

. Let $\left\{\left(S_t^{(1)}, \cdots, S_t^{(C)}, Y_t^{(1)}, \cdots, Y_t^{(C)}\right)\right\}$ be a latent structure influence process characterized by the marginal initial state distributions $\pi_s^{(c)} = P(S_1^{(c)} = s)$, the marginal state transition probabilities $P(s_{t+1}^{(c)}|s_t^{(1)}\cdots s_t^{(C)}) = \sum_{c_1=1}^{C} h_{s_t^{(c_1)}, s_{t+1}^{(c)}}^{(c_1,c)}$, and the emission probabilities $P(Y_t^{(c)}|S_t^{(c)})$. Given $\left(Y_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}$, the statistical values $\alpha_t^{(c)}(s_t)$, $\beta_t^{(c)}(s_t)$, $\gamma_t^{(c)}(s_t)$, and $\xi_{t\rightarrow t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)})$ are the marginal forward parameters, the marginal backward parameters, the marginal one-slice parameters, and the marginal two-slice

parameters respectively:

$$\begin{aligned}
\alpha_t^{(c)}(s_t^{(c)}) &= P\left(s_t^{(c)}, \left(y_{t_1}^{(c)}\right)_{1\le t_1\le t}^{1\le c\le C}\right) \\
\beta_t^{(c)}(s_t^{(c)}) &= P\left(\left(y_{t_1}^{(c)}\right)_{t+1\le t_1\le T}^{1\le c\le C} \Big| s_t^{(c)}\right) \\
\gamma_t^{(c)}(s_t^{(c)}) &= P\left(s_t^{(c)}, \left(y_{t_1}^{(c)}\right)_{1\le t_1\le T}^{1\le c\le C}\right) \\
\xi_{t\to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)}) &= P\left(s_t^{(c_1)}, s_{t+1}^{(c_2)}, \left(y_{t_1}^{(c)}\right)_{1\le t_1\le T}^{1\le c\le C}\right)
\end{aligned}$$

We express the above parameters and the statistics related to the hidden Markov model with the following matrix form:

$$\begin{aligned}
(\pi) &= \begin{pmatrix} \pi_1 & \cdots & \pi_{\prod_c m_c} \end{pmatrix} \\
(\alpha_t) &= \begin{pmatrix} \alpha_t(1) & \cdots & \alpha_t(\prod_c m_c) \end{pmatrix} \\
(\beta_t) &= \begin{pmatrix} \beta_t(1) \\ \vdots \\ \beta_t(\prod_c m_c) \end{pmatrix} \\
(p\,(y_t|s_t)) &= \begin{pmatrix} P(y_t^{(1)}\cdots y_t^{(C)}|S=1) & & \\ & \ddots & \\ & & P(y_t^{(1)}\cdots y_t^{(C)}|S=\prod_c m_c) \end{pmatrix}
\end{aligned}$$

We express the above parameters and the statistics related to the latent structure influence model with the following matrix form:

$$\begin{aligned}
\left(\pi^{(c)}\right) &= \begin{pmatrix} \underbrace{\pi_1^{(1)} \cdots \pi_{m_1}^{(1)}}_{m_1} & \cdots & \underbrace{\pi_1^{(C)} \cdots \pi_{m_C}^{(C)}}_{m_C} \end{pmatrix} \\
\left(\alpha_t^{(c)}\right) &= \begin{pmatrix} \underbrace{\alpha_t^{(1)}(1) \cdots \alpha_t^{(1)}(m_1)}_{m_1} & \cdots & \underbrace{\alpha_t^{(C)}(1) \cdots \pi_t^{(C)}(m_C)}_{m_C} \end{pmatrix} \\
\left(\beta_t^{(c)}\right) &= \begin{pmatrix} \left.\begin{matrix} \beta_t^{(1)}(1) \\ \vdots \\ \beta_t^{(1)}(m_1) \end{matrix}\right\} m_1 \\ \vdots \\ \left.\begin{matrix} \beta_t^{(C)}(1) \\ \vdots \\ \beta_t^{(C)}(m_C) \end{matrix}\right\} m_C \end{pmatrix} \\
\left(P(y^{(c)}|s^{(c)})\right) &= \texttt{diag}\Big(\underbrace{P(y_t^{(1)}|S_t^{(1)}=1) \cdots P(y_t^{(1)}|S_t^{(1)}=m_1)}_{m_1} \cdots \\
&\qquad \underbrace{P(y_t^{(C)}|S_t^{(C)}=1) \cdots P(y_t^{(C)}|S_t^{(C)}=m_C)}_{m_C} \Big)
\end{aligned}$$

With this notation, we can verify that the statistical parameters for a hidden Markov model can be expressed in the following non-recursive form

$$\begin{aligned}
(\alpha_t) &= (\pi)\cdot(P(y_1|s_1))\cdot G\cdot(P(y_2|s_2))\cdots G\cdot(P(y_t|s_t)) \\
(\beta_t) &= G\cdot(P(o_{t+1}|s))\cdots G\cdot(P(o_T|s))\cdot 1_{\prod_c m_c\times 1} \\
(\gamma_t) &= (\alpha_t).*(\beta_t) \\
(\xi_{t\to t+1}) &= G.*((\beta_t)\cdot(\alpha_t)) \\
P(y_1\cdots y_t) &= (\alpha_T)\cdot 1_{(\prod_c m_c)\times 1} = 1\cdot(\beta_1) = \alpha_t\cdot\beta_t
\end{aligned}$$

The forward and backward parameters (α_t) and (β_t) can also be expressed in the following recursive form:

$$\begin{aligned}
(\alpha_{t=1}) &= (\pi)\cdot(P(y_1|s_1)) \\
(\alpha_{t+1}) &= (\alpha_t)\cdot G\cdot(P(y_t|s)) \\
(\beta_{t=T}) &= 1_{\prod_c m_c\times 1} \\
(\beta_t) &= G\cdot(P(y_{t+1}|s))\cdot(\beta_{t+1})
\end{aligned}$$

Since we have assumed that the probability mass functions for $S_t^{(c)}$ and the probability mass functions for S_t can be mapped to each other with the marginalizing operator B and its inverse B^+, the space of valid probability mass functions are restricted, and the transition matrix has the restriction that $G = BB^+GBB^+$. With this assumption, we can recombine the terms and write the marginal forward parameters $\left(\alpha_t^{(c)}\right)$ as the following:

$$\begin{aligned}
(\alpha_t) &= \pi\cdot BB^+\cdot(P(y_1|s_1))\cdot BB^+\cdot G\cdot BB^+\cdots BB^+\cdot G\cdot BB^+\cdot(P(y_t|s_t)) \\
\left(\alpha_t^{(c)}\right) &\triangleq (\alpha_t)\cdot B \\
&= (\pi\cdot B)\cdot\left(B^+\cdot(P(y_1|s_1))\cdot B\right)\cdot H\cdot\cdots\cdot H\cdot\left(B^+\cdot(P(y_t|s_t))\cdot B\right)
\end{aligned}$$

We can also recombine the terms and write the marginal backward parameters $\left(\beta_t^{(c)}\right)$ as the following:

$$\begin{aligned}
(\beta_t) &= BB^+\cdot G\cdot BB^+\cdot(P(y_{t+1}|s_t))\cdots BB^+\cdot G\cdot BB^+\cdot(P(y_T|s_T))\cdot 1_{\prod_c m_c\times 1} \\
\left(\beta_t^{(c)}\right) &\triangleq H\cdot\left(B^+\cdot(P(y_{t+1}|s_{t+1}))\cdot B\right)\cdots H\cdot\left(B^+\cdot(P(y_T|s_T))\cdot 1_{\prod_c m_c\times 1}\right) \\
&= H\cdot\left(B^+\cdot(P(y_{t+1}|s_{t+1}))\cdot B\right)\cdots H\cdot\left(B^+\cdot(P(y_T|s_T))\cdot B\right)\cdot(B^+\cdot 1_{\prod_c m_c\times 1}) \\
&= H\cdot\left(B^+\cdot(P(y_{t+1}|s_{t+1}))\cdot B\right)\cdots H\cdot\left(B^+\cdot(P(y_T|s_T))\cdot B\right)\cdot\frac{1_{\sum m_c\times 1}}{\sum m_c} \\
(\beta_t) &= B\cdot\left(\beta_t^{(c)}\right)
\end{aligned}$$

The likelihood of observing sequence $y_1\cdots y_T$ can be expressed as

$$P(y_1\cdots y_T) = \left(\alpha_t^{(c)}\right)\cdot\left(\beta_t^{(c)}\right)$$

The recursive formulas for the marginal forward and backward parameters are thus:

$$\begin{aligned}
\left(\alpha_{t=1}^{(c)}\right) &= \left(\pi^{(c)}\right)\cdot(B^+\cdot(P(y_1|s))\cdot B) \\
\left(\alpha_{t+1}^{(c)}\right) &= \left(\alpha_t^{(c)}\right)\cdot H\cdot(B^+\cdot(P(y_{t+1}|s))\cdot B) \\
\left(\beta_{t=T}^{(c)}\right) &= \frac{1}{\sum m_c}\cdot 1_{\sum m_c\times 1} \\
\left(\beta_t^{(c)}\right) &= H\cdot\left(B^+\cdot(P(y_{t+1}|s_{t+1}))\cdot B\right)\cdot\left(\beta_{t+1}^{(c)}\right)
\end{aligned}$$

As a result, if we can compute $(B^+ \cdot (P(y_t|s_t)) \cdot B)$ in terms of $S_t^{(c)}$, we can make the inference without involving S_t. This is actually true, as we will demonstrate below.

Theorem 3. *Let $1 \leq S^{(c)} \leq m_c$, where $1 \leq c \leq C$, be random variables, and the random variable $S\left(\left(S_t^{(c)}\right)^{1\leq c\leq C}\right)$ be the index of $\left(S_t^{(c)}\right)^{1\leq c\leq C}$. The random variables $S^{(c)}$ are independent:*

$$P\left(S\left(\left(s^{(c)}\right)^{1\leq c\leq C}\right)\right) = \prod_c P\left(s^{(c)}\right)$$

. Let $B\left(\begin{pmatrix} m_1 & \cdots & m_C \end{pmatrix}\right)$ and B^+ be the marginalizing matrix and its inverse, and

$$\texttt{obslik} = B^+ \cdot \begin{pmatrix} P(S=1) & & \\ & \ddots & \\ & & P(S=\prod_c m_c) \end{pmatrix} \cdot B$$

We have

$$\texttt{obslik}(\texttt{c}_1, \texttt{c}_2, \texttt{i}, \texttt{j}) = -\frac{\sum_{k\neq c_1} \frac{1}{m_k}}{(\prod_k m_k)\cdot\left(\sum_k \frac{1}{m_k}\right)} \cdot P(S^{(c_2)}=j) + \begin{cases} \frac{1}{\prod_{k\neq c_1} m_k} P(S^{(c_1)}=i)\cdot P(S^{(c_2)}=j) & , c_1 \neq c_2 \\ \frac{1}{\prod_{k\neq c_1} m_k} P(S^{(c_1)}=i) & , c_1 = c_2, i = j \\ 0 & , c_1 = c_2, i \neq j \end{cases}$$

2.1.3 Parameter Estimation

In this section, we give the likelihood function for a latent state influence model, with the assumption that the marginal probability mass functions can be mapped linearly to a joint probability mass function. We consider two situations: the latent states $\left(S_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}$ are known exactly, and only a probabilistic estimate of $\left(S_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}$ is known. Based on the likelihood function, we derive the parameter re-estimation formula for the latent state influence model. It should be noticed that the correlations of $\left(S_t^{(c)}\right)^{1\leq c\leq C}$ within the same time slice t are not captured by the influence matrix. They are provided by the observations, when we combine the evidence from new observations $P(\vec{y}_t|\vec{s}_t)$ with the estimation from old observations $P(\vec{s}_t|\vec{y}_{1\cdots t-1})$. The influence matrix only captures the correlations between $\left(S_{t-1}^{(c)}\right)^{1\leq c\leq C}$ and $\left(S_t^{(c)}\right)^{1\leq c\leq C}$.

When we know the latent states exactly, $\left(S_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C} = \left(s_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}$ corresponding to the observations $\left(Y_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C} = \left(y_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}$, the likelihood function can be computed as follows:

$$\begin{aligned} p\left(\left(y_t^{(c)}\right)_{1\leq t\leq T}^{1\leq c\leq C}\right) &= p(\vec{s}_1)\cdot\left(\prod_{t=1}^{T-1} p(\vec{s}_{t+1}|\vec{s}_t)\right)\cdot\left(\prod_{t=1}^{T} p(\vec{y}_t|\vec{s}_t)\right) \\ &= \left(\prod_c p(\pi^{(c)}_{s_1^{(c)}})\right)\cdot\left(\prod_{t=1}^{T-1}\sum_{c_1=1}^{C}\sum_{c_2=1}^{C} h^{(c_1,c_2)}_{s_t^{(c_1)} s_{t+1}^{(c_2)}}\right)\left(\prod_{t=1}^{T}\prod_{c=1}^{C} p(y_t^{(c)}|s_t^{(c)})\right) \end{aligned}$$

We can find a new estimate of the parameters to try to maximize the log likelihood function:

$$
\begin{aligned}
& \log p\left(\left(y_t^{(c)}\right)_{1\le t\le T}^{1\le c\le C}\right) \\
= & \sum_{c=1}^{C} p(\pi^{(c)}_{s_1^{(c)}}) + \sum_{t=1}^{T-1} \log \sum_{c_1=1}^{C}\sum_{c_2=1}^{C} h^{(c_1,c_2)}_{s_t^{(c_1)} s_{t+1}^{(c_2)}} + \sum_{t=1}^{T}\sum_{c=1}^{C} \log p(y_t^{(c)}|s_t^{(c)}) \\
\ge & \sum_{c=1}^{C} p(\pi^{(c)}_{s_1^{(c)}}) + \sum_{t=1}^{T-1} \sum_{c_1=1}^{C}\sum_{c_2=1}^{C} \log h^{(c_1,c_2)}_{s_t^{(c_1)} s_{t+1}^{(c_2)}} + \sum_{t=1}^{T}\sum_{c=1}^{C} \log p(y_t^{(c)}|s_t^{(c)}) \qquad (2.1) \\
= & \sum_{c=1}^{C}\sum_{i=1}^{m_c} \delta(s_1^{(c)}, i)\cdot \log \pi_i^{(c)} + \\
& \sum_{t=1}^{T-1}\sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}} \delta(s_t^{(c1)}, i)\cdot\delta(s_{t+1}^{(c2)}, j)\cdot \log h_{i,j}^{(c1,c2)} + \sum_{t=1}^{T}\sum_{c=1}^{C}\sum_{i=1}^{m_c} \delta(s_t^{(c)}, i)\cdot \log p(y_t^{(c)}|i) \\
\stackrel{\triangle}{=} & \sum_{c=1}^{C}\sum_{i=1}^{m_c} \tilde{\pi}_i^{(c)}\cdot \log \pi_i^{(c)} + \\
& \sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}} \tilde{\xi}^{(c1,c2)}(i,j)\cdot \log h_{i,j}^{(c1,c2)} + \sum_{c=1}^{C}\sum_{i=1}^{m_c} \tilde{\gamma}^{(c)}(i)\cdot \log p(y_t^{(c)}|i)
\end{aligned}
$$

where the step 2.1 is according to the Jensen's inequality, the function

$$\delta(i,j) = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

is the Kronecker delta function, and

$$
\begin{aligned}
\tilde{\pi}_i^{(c)} &= \delta(s_1^{(c)}, i) \\
\tilde{\xi}^{(c1,c2)}(i,j) &= \sum_{t=1}^{T-1} \delta(s_t^{(c1)}, i)\cdot\delta(s_{t+1}^{(c2)}, j) \\
\tilde{\gamma}^{(c)}(i) &= \sum_{t=1}^{T} \delta(s_t^{(c)}, i)
\end{aligned}
$$

are the sufficient statistics for $\pi_i^{(c)}$, $h_{i,j}^{(c1,c2)}$, and $p(y_t^{(c)}|i)$ respectively. We can maximize the parameters involved in the influence matrix H by equaling them to the corresponding sufficient statistics:

$$\pi_i^{(c)} = \tilde{\pi}_i^{(c)} \qquad (2.2)$$

$$h_{i,j}^{(c1,c2)} = \frac{1}{C}\cdot\frac{\tilde{\xi}_{i,j}^{(c1,c2)}}{\sum_{j=1}^{m_{c2}} \tilde{\xi}_{i,j}^{(c1,c2)}} \qquad (2.3)$$

We can maximize the parameters that map from the latent states to the observations in the same way as in an ordinary hidden Markov model.

When the latent states at time $t = 1..T$ are not known. We can choose parameters that maximize the expected log likelihood function:

$$
\begin{aligned}
& \mathtt{E}_{\vec{s}_1 \cdots \vec{s}_T} \left[\log p \left(\left(y_t^{(c)} \right)_{1 \le t \le T}^{1 \le c \le C} \right) \right] \\
= \; & \mathtt{E}_{\vec{s}_1 \cdots \vec{s}_T} \left[\sum_{c=1}^{C} \log \pi_{s_1^{(c)}}^{(c)} + \sum_{t=1}^{T} \log \sum_{c1=1}^{C} \sum_{c2=1}^{C} h_{s_t^{(c1)}, s_{t+1}^{(c2)}}^{(c1,c2)} + \sum_{t=1}^{T} \sum_{c=1}^{C} \log p(y_t^{(c)} | s_t^{(c)}) \right] \\
\ge \; & \mathtt{E}_{\vec{s}_1 \cdots \vec{s}_T} \left[\sum_{c=1}^{C} \log \pi_{s_1^{(c)}}^{(c)} + \sum_{t=1}^{T} \sum_{c2=1}^{C} \sum_{c1=1}^{C} \log h_{s_t^{(c1)}, s_{t+1}^{(c2)}}^{(c1,c2)} + \sum_{t=1}^{T} \sum_{c=1}^{C} \log p(y_t^{(c)} | s_t^{(c)}) \right] \\
= \; & \sum_{c=1}^{C} \sum_{i=1}^{m_c} \mathtt{E}_{s_1^{(c)}} \left[\delta(s_1^{(c)}, i) \right] \cdot \log \pi_i^{(c)} + \\
& \sum_{c=1}^{C} \sum_{c1=1}^{C} \sum_{i=1}^{m_{c1}} \sum_{j=1}^{m_{c2}} \sum_{t=1}^{T-1} \mathtt{E}_{s_t^{(c1)} s_{t+1}^{(c2)}} \left[\delta(s_t^{(c1)}, i) \cdot \delta(s_{t+1}^{(c2)}, j) \right] \cdot \log h_{i,j}^{(c1,c2)} + \\
& \sum_{c=1}^{C} \sum_{i=1}^{m_c} \sum_{t=1}^{T} \mathtt{E}_{s_t^{(c)}} \left[\delta(s_t^{(c)}, i) \right] \cdot \log p(y_t^{(c)} | i) \\
\triangleq \; & \sum_{c=1}^{C} \sum_{i=1}^{m_c} \tilde{\pi}_i^{(c)} \cdot \log \pi_i^{(c)} + \\
& \sum_{c=1}^{C} \sum_{c1=1}^{C} \sum_{i=1}^{m_{c1}} \sum_{j=1}^{m_{c2}} \tilde{\xi}^{(c1,c2)}(i,j) \cdot \log h_{i,j}^{(c1,c2)} + \sum_{c=1}^{C} \sum_{i=1}^{m_c} \tilde{\gamma}^{(c)}(i) \cdot \log p(y_t^{(c)} | i)
\end{aligned}
$$

According to the attributes of the expectation operator and the Kronecker delta operator, the sufficient statistics are given in the following way, and the parameters related to the state transitions are maximized by equations 2.2 and 2.3:

$$
\begin{aligned}
\tilde{\pi}_i^{(c)} &= \gamma_i^{(c)} \\
\tilde{\xi}^{(c1,c2)}(i,j) &= \sum_{t=1}^{T=1} \xi_{t \to t+1}^{(c1,c2)}(i,j) \\
\tilde{\gamma}^{(c)}(i) &= \sum_{t=1}^{T} \gamma_t^{(c)}(i)
\end{aligned}
$$

We summarize the EM algorithm for the latent structure influence model in Algorithm 1.

2.2 The Non-linear Approach to the Influence Modeling

We need to notice that the probability space of a joint random variable $S\left(\left(S^{(c)}\right)_{1 \le c \le C}\right)$ is exponentially larger than the probability spaces of its constituent random variables, and there are many mappings between the probability space of the joint random variable and the marginal random variables. In Section 2.1, we defined a class of linear mappings between the joint random variable and the marginal random variables, formulated the latent structure influence model based on this mapping, and showed that the inference algorithms for an latent structure influence model is equivalent to the inference algorithms for the corresponding

Algorithm 1 The EM algorithm for the latent structure influence model (linear)

E-Step

$$
\begin{aligned}
\left(\alpha_t^{(c)}\right) &= \begin{cases} \left(\pi^{(c)}\right) \cdot B^{+}\left(p\left(\vec{y}_1|\vec{s}_1\right)\right) B & t=1 \\ \left(\alpha_{t-1}^{(c)}\right) \cdot H \cdot B^{+}\left(p\left(\vec{y}_{t-1}|\vec{s}_{t-1}\right)\right) B & t>1 \end{cases} \\
\left(\beta_t^{(c)}\right) &= \begin{cases} \frac{1}{\sum_c m_c}\vec{1}_{\sum m_c \times 1} & t=T \\ H \cdot B^{+}\left(p\left(\vec{y}_{t-1}|\vec{s}_{t-1}\right)\right) B \cdot \left(\beta_{t+1}^{(c)}\right) & t<T \end{cases} \\
\left(\gamma_t^{(c)}\right) &= \left(\alpha_t^{(c)}\right) \cdot \mathtt{diag}[\left(\beta_t^{(c)}\right)] \\
\xi_{t-1\to t} &= \mathtt{diag}[\left(\alpha_{t-1}^{(c)}\right)] \cdot H \cdot B^{+}\left(p\left(\vec{y}_{t-1}|\vec{s}_{t-1}\right)\right) B \cdot \mathtt{diag}[\left(\beta_t^{(c)}\right)] \\
p(\vec{y}) &= \left(\alpha_t^{(c)}\right) \cdot \left(\beta_t^{(c)}\right)
\end{aligned}
$$

M-step

- Parameters related to the latent state transitions

$$
\begin{aligned}
A_{ij} &= \mathtt{normalize}[\sum_{t=2}^{T} \xi_{t-1\to t}^{(i,j)}] \\
S &= \begin{pmatrix} \vec{1}_{1\times m_1} & & \\ & \ldots & \\ & & \vec{1}_{1\times m_C} \end{pmatrix} \\
d_{ij} &= \mathtt{normalize}[S \sum_{t=2}^{T} \xi_{t-1\to t} S^{\mathtt{T}}] \\
\vec{\pi}^{(c)} &= \mathtt{normalize}[\vec{\gamma}_1^{(c)}]
\end{aligned}
$$

Parameters related to multinomial observations

$$
p^{(c)}(\vec{o}^{(c)}|\vec{s}^{(c)}) = \mathtt{normalize}[\sum_t \vec{\gamma}_t^{(c)\,\mathtt{T}} \cdot \vec{\delta}_{y_t,\vec{o}^{(c)}}]
$$

Parameters related to Gaussian observations

$$
\begin{aligned}
\mu^{(c)} &= \frac{\sum_t \vec{\gamma}_t^{(c)\,\mathtt{T}} \cdot \vec{y}_t^{(c)}}{\sum_t \vec{\gamma}_t^{(c)} \cdot \vec{1}_{m_c\times 1}} \\
\Sigma_i^{(c)\,2} &= \frac{\sum_t \gamma_{t,i}^{(c)} \vec{y}_t^{(c)} \vec{y}_t^{(c)\,\mathtt{T}}}{\sum_t \vec{\gamma}_t^{(c)} \cdot \vec{1}_{m_c\times 1}} - \vec{\mu}_i^{(c)} \cdot \vec{\mu}_i^{(c)\,\mathtt{T}}
\end{aligned}
$$

hidden Markov model. In this section, we show that a non-linear mapping from the marginal random variables to the joint random variable also exists (the mapping from the joint random variable to the marginal random variables is always linear). We also formulate the latent structure influence model based on the non-linear mapping, and derive the inference algorithms. We will see shortly why we need to map the marginal probability distributions to the joint probability distribution, and why we call it non-linear.In the non-linear case, the forward-backward algorithm for an influence model is an approximation to the forward-backward algorithm for the corresponding hidden Markov model.

In the following paragraphs, we describe the influence parameters, the evolution of the marginal latent state distributions for individual processes, and the observations for individual processes as probabilistic functions of the latent states. The usage with an influence model is generally: inference of latent states given parameters and observations, estimation of parameters given latent states and observations, or simultaneous latent state inference and parameter estimation from observations. A graphical model representation of the influence model is plotted in Figure 2.2. In this figure, the left column represents basis step, and the right column represents the induction step. Black squares are observable, and white squares represent latent states. Our task is to learn the parameters and latent states from observations. The two-column convention is adopted from Murphy [2].

Let us assume that we have C interacting stochastic processes and $m_c (1 \leq c \leq C)$ number of latent states corresponding to process c in the system's behavior space. Following Asavathiratham [12] we use $D_{C \times C}$ as the network (influence) matrix, whose columns each add up to 1, and $A^{(c_1,c_2)} (1 \leq c_1, c_2 \leq C)$ as the inter-process state transition matrix, whose rows adds up to 1. The influence matrix is defined as the Kronecker product $H = D \otimes A = (d_{c_1,c_2} A^{(c_1,c_2)}) = (h_{i,j}^{(c_1,c_2)})$, where H is a block matrix whose sub-matrix at row c_1 and column c_2 is $d_{c_1,c_2} A^{(c_1,c_2)}$. The influence matrix is used to generate the marginal latent state distributions for individual processes at time $t+1$ from the marginal latent state distributions at time t. The marginal latent state distributions for individual processes c at time 1 is given as $\vec{\pi}^{(c)} = (p(s_1^{(c)} = 1), \ldots, p(s_1^{(c)} = m_c)) \triangleq (\pi_1^{(c)}, \ldots, \pi_{m_c}^{(c)})$, and we concatenate the row vectors $\vec{\pi}^{(c)}$ into a longer row vector $\vec{\pi} = (\vec{\pi}^{(1)}, \ldots, \vec{\pi}^{(C)})$.

We express the marginal probabilistic distributions of states $s_t^{(c)}$ for processes c at time t as a row vector $\vec{p}(s_t^{(c)}) = (p(s_t^{(c)} = 1), p(s_t^{(c)} = 2), \ldots, p(s_t^{(c)} = m_c))$, where $1 \leq c \leq C, \sum_j p(s_t^{(c)} = j) = 1$, and concatenate these row vectors together into a longer row vector: $\vec{p}(\vec{s}_t) = (\vec{p}(s_t^{(1)}), \vec{p}(s_t^{(2)}), \ldots, \vec{p}(s_t^{(C)}))$. Using this notation, the influence marginal latent state distributions for individual interacting processes are evolved as

$$\begin{aligned} p(\vec{s}_{t+1}) &= p(\vec{s}_t) \cdot H \\ p(\vec{s}_1) &= \vec{\pi} \end{aligned}$$

There are many ways to establish the equivalence relation between an influence process represented by the influence matrix H and a Markov process represented by the state transition matrix G. We follow Asavathiratham's formulation in [12] Section 5.11: $G = (g_{i,j})$, where $g_{\vec{s}_t,\vec{s}_{t+1}} = \prod_c \sum_{c_1} h^{(c_1,c)}_{s_t^{(c_1)},s_{t+1}^{(c)}}$. Intuitively, this means that we can linearly combine the marginal latent state distributions at time t to get the marginal latent state distributions at time $t+1$, and that a joint latent state distribution for a system can be factored into the product of the marginal latent state distributions of individual processes for this system.

The observations $o_t^{(c)}$ for process c at sample times $1 \leq t \leq +\infty$ can be either finite symbolic or Gaussian and are statistically determined by the corresponding latent state $s_t^{(c)}$. When the observations for process c are symbolic, we use n_c to represent the number of observation symbols $o_t^{(c)} \in [1 \ldots n_c]$. We define $B^{(c)} = (b_{i,j}^{(c)})$, where $b_{i,j}^{(c)} = p(o^{(c)} = j | s^{(c)} = i)$, as the observation matrix. When the observations for process c are Gaussian, we use n_c to represent the dimensionality of the m_c number of Gaussian

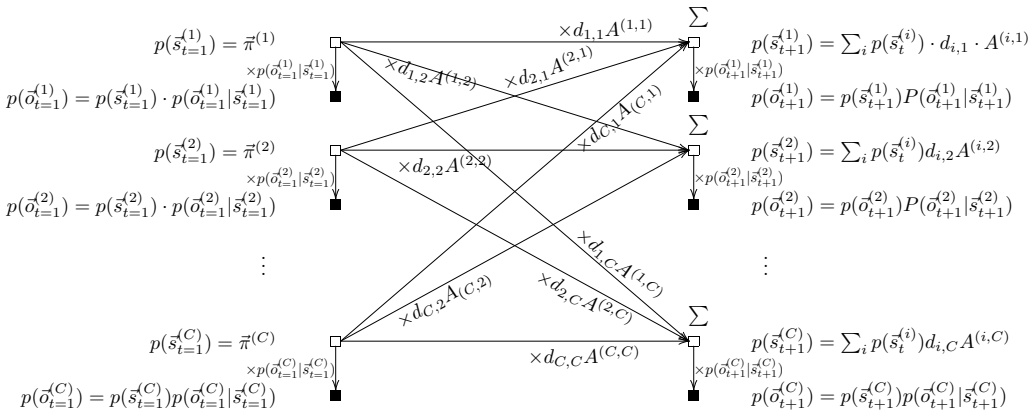

Figure 2.2: A graphical model representation of the influence model.

distributions corresponding to each latent state $s^{(c)} \in [1 \dots m_c]$: $o^{(c)} \sim \mathcal{N}(\mu_{s^{(c)}}^{(c)}, \sigma_{s^{(c)}}^{(c)})$.

2.2.1 Latent State Inference

In the inference algorithm below with a non-linear mapping interpretation of H, the computation of the forward parameters needs special care. be the conditional marginal probability for $s_t^{(c)}$ given the observations $y_{t_1}^{(c_1)}$, where $1 \leq c_1 \leq C$ and time $1 \leq t_1 \leq t$. Let the marginal backward parameters be the conditional probability of $\left(y_{t_1}^{(c_1)}\right)_{t+1 \leq t_1 \leq T}^{1 \leq c_1 \leq C}$ given $s_t^{(c)}$

Theorem 4. *Let the marginal forward parameters $\alpha_t^{(c)}(s_t^{(c)})$, the marginal backward parameters $\beta_t^{(c)}(s_t^{(c)})$, the marginal one-slice parameters $\gamma_t^{(c)}(s_t^{(c)})$, the marginal two-slice parameters $\xi_{t \to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)})$ of a latent structure influence model be*

$$\begin{aligned}
\alpha_t^{(c)}(s_t^{(c)}) &= p\left(s_t^{(c)}, \left(y_{t_1}^{(c_1)}\right)_{1 \leq t_1 \leq t}^{1 \leq c_1 \leq C}\right) \\
\beta_t^{(c)}(s_t^{(c)}) &= P\left(\left(y_{t_1}^{(c_1)}\right)_{t+1 \leq t_1 \leq T}^{1 \leq c_1 \leq C} \Big| s_t^{(c)}\right) \\
\gamma_t^{(c)}(s_t^{(c)}) &= P\left(s_t^{(c)} \Big| \left(y_{t_1}^{(c_1)}\right)_{1 \leq t_1 \leq T}^{1 \leq c_1 \leq C}\right) \\
\xi_{t \to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)}) &= P\left(s_t^{(c_1)} s_{t+1}^{(c_2)} \Big| \left(y_{t_1}^{(c_1)}\right)_{1 \leq t_1 \leq T}^{1 \leq c_1 \leq C}\right)
\end{aligned}$$

They can be computed recursively in the following way:

$$
\begin{aligned}
\alpha_1^{(c)}(s_1^{(c)}) &= p\left(\left(y_t^{(c)}\right)^{1\le c\le C}|s_1^{(c)}\right)\cdot \pi^{(c)}_{s_1^{(c)}} \\
\alpha_t^{(c)}(s_{2\le t}^{(c)}) &= p\left(\left(y_t^{(c)}\right)^{1\le c\le C}|s_t^{(c)}\right)\sum_{c_1,s_{t-1}^{(c_1)}}\alpha(s_{t-1}^{(c_1)})h^{(c_1,c)}_{s_{t-1}^{(c_1)}s_t^{(c)}} \\
\beta_T^{(c)}(s_T^{(c)}) &= 1 \\
\beta_{t<T}^{(c)}(s_t^{(c)}) &= \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s_{t+1}^{(c_1)}=1}^{m_{c_1}} h^{(c,c_1)}_{s_t^{(c)},s_{t+1}^{(c_1)}}\cdot P\left(\left(y_{t+1}^{(c)}\right)^{1\le c\le C}\left|s_{t+1}^{(c_1)}\right.\right)\beta_{t+1}^{(c)}(s_{t+1}^{(c)}) \\
\gamma_t^{(c)}(s_t^{(c)}) &= \alpha_t^{(c)}(s_t^{(c)})\cdot\beta_t^{(c)}(s_t^{(c)}) \\
\xi^{(c_1,c_2)}_{t\to t+1}(s_t^{(c_1)},s_{t+1}^{(c_2)}) &= \alpha_t^{(c_1)}(s_t^{(c_1)})\cdot h^{(c_1,c_2)}_{s_t^{(c_1)},s_{t+1}^{(c_2)}}\cdot\beta_{t+1}^{(c_2)}(s_{t+1}^{(c_2)})\cdot P\left(\left(y_{t+1}^{(c)}\right)^{1\le c\le C}|s_{t+1}^{(c_2)}\right)
\end{aligned}
$$

Proof. In the following, we demonstrate that we can solve for the marginal forward parameters without first solving the joint marginal forward parameters.

- Basis Step

$$
\begin{aligned}
& \alpha(s_1^{(c)}) \\
= \;& p\left(s_t^{(c)},\left(y_1^{(c_1)}\right)^{1\le c_1\le C}\right) \\
= \;& p\left(\left(y_1^{(c_1)}\right)^{1\le c_1\le C}|s_1^{(c)}\right)\cdot p\left(s_1^{(c)}\right) \\
= \;& p\left(\left(y_1^{(c_1)}\right)^{1\le c_1\le C}|s_1^{(c)}\right)\cdot\pi^{(c)}_{s_1^{(c)}}
\end{aligned}
$$

- Induction Step

$$
\begin{aligned}
& \alpha(s_{t\ge 2}^{(c)}) \\
= \;& p\left(s_t^{(c)},\left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le t}\right) \\
= \;& p\left(\left(y_t^{(c_1)}\right)^{1\le c_1\le C}|s_t^{(c)}\right)\cdot p\left(s_t^{(c)},\left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le t-1}\right) \\
= \;& p\left(\left(y_t^{(c_1)}\right)^{1\le c_1\le C}|s_t^{(c)}\right)\cdot\sum_{c_1=1}^{C}\sum_{s_{t-1}^{(c_1)}=1}^{m_{c_1}} p\left(s_{t-1}^{(c_1)},\left(y_{t_1}^{(c_1)}\right)^{1\le c_1\le C}_{1\le t_1\le t-1}\right)\cdot h^{(c_1,c)}_{s_{t-1}^{(c_1)}s_t^{(c)}} \\
= \;& p\left(\left(y_t^{(c_1)}\right)^{1\le c_1\le C}|s_t^{(c)}\right)\cdot\left(\sum_{c_1=1}^{C}\sum_{s_{t-1}^{(c_1)}=1}^{m_{c_1}}\alpha(s_{t-1}^{(c_1)})\cdot h^{(c_1,c)}_{s_{t-1}^{(c_1)}s_t^{(c)}}\right)
\end{aligned}
$$

In the following, we show that we can get the marginal backward parameters without the knowledge of the joint backward parameters.

- Basis Step. We have $\beta(s_T^{(c)}) = 1$ trivially, and

$$\begin{aligned}
\sum_{s_T^{(c)}=1}^{m_c} \alpha(s_T^{(c)}) \cdot \beta(s_T^{(c)}) &= \sum_{s_T^{(c)}=1}^{m_c} p\left(s_T^{(c)}, \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le T}^{1\le c_1\le C}\right) \\
&= p\left(\left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le T}^{1\le c_1\le C}\right) \\
\frac{1}{C}\cdot\sum_{c=1}^{C}\sum_{s_T^{(c)}=1}^{m_c} \alpha(s_T^{(c)}) \cdot \beta(s_T^{(c)}) &= p\left(\left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le T}^{1\le c_1\le C}\right)
\end{aligned}$$

- Induction Step

$$\begin{aligned}
&\beta(s_{t<T}^{(c)}) \\
= \; & p\left(\left(y_{t_1}^{(c_1)}\right)_{t+1\le t_1\le T}^{1\le c_1\le C} \big| s_t^{(c)}\right) \\
= \; & \sum_{s_{t+1}^{(c_1)}=1}^{m_C} p\left(s_{t+1}^{(c_1)}, \left(y_{t_1}^{(c_1)}\right)_{t+1\le t_1\le T}^{1\le c_1\le C} \big| s_t^{(c)}\right), 1 \le c_1 \le C \\
= \; & \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s_{t+1}^{(c_1)}=1}^{m_C} p\left(s_{t+1}^{(c_1)}, \left(y_{t_1}^{(c_1)}\right)_{t+1\le t_1\le T}^{1\le c_1\le C} \big| s_t^{(c)}\right) \\
= \; & \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s_{t+1}^{(c_1)}=1}^{m_C} p\left(\left(y_{t_1}^{(c_1)}\right)_{t+1\le t_1\le T}^{1\le c_1\le C} \big| s_{t+1}^{(c_1)}\right)\cdot p(s_{t+1}^{(c_1)}|s_t^{(c)}) \\
= \; & \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s_{t+1}^{(c_1)}=1}^{m_C} p\left(\left(y_{t_1}^{(c_1)}\right)_{t+2\le t_1\le T}^{1\le c_1\le C} \big| s_{t+1}^{(c_1)}\right)\cdot p\left(\left(y_{t+1}^{(c_1)}\right)^{1\le c_1\le C} \big| s_{t+1}^{(c_1)}\right)\cdot p(s_{t+1}^{(c_1)}|s_t^{(c)}) \\
= \; & \frac{1}{C}\cdot\sum_{c_1=1}^{C}\sum_{s_{t+1}^{(c_1)}=1}^{m_{c_1}} \beta(s_{t+1}^{(c_1)})\cdot h_{s_t^{(c)} s_{t+1}^{(c_1)}}^{(c_1,c)}\cdot p\left(\left(y_t^{(c_1)}\right)^{1\le c_1\le C} | s_{t+1}^{(c_1)}\right)
\end{aligned}$$

The one-slice parameters $\gamma_t^{(c)}(s_t^{(c)})$ can be computed from the marginal forward parameters and the marginal backward parameters

$$\begin{aligned}
\gamma_t^{(c)}(s_t^{(c)}) &= P\left(s_t^{(c)}, \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le T}^{1\le c_1\le C}\right) \\
&= P\left(s_t^{(c)}, \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le t}^{1\le c_1\le C}\right) P\left(\left(y_{t_1}^{(c_1)}\right)_{t+1\le t_1\le T}^{1\le c_1\le C} \big| s_t^{(c)}\right) \\
&= \alpha_t^{(c)}(s_t^{(c)})\cdot\beta_t^{(c)}(s_t^{(c)})
\end{aligned}$$

The two-slice parameters $\xi_{t\to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)})$ can also be computed from the marginal forward parameters

$\alpha_t^{(c)}(s_t^{(c)})$ and the marginal backward parameters $\beta_t^{(c)}(s_t^{(c)})$:

$$\begin{aligned}
\xi_{t\to t+1}^{(c_1,c_2)}(s_t^{(c_1)}, s_{t+1}^{(c_2)}) &= P\left(s_t^{(c_1)} s_{t+1}^{(c_2)}, \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le T}^{1\le c_1\le C}\right) \\
&= P\left(s_t^{(c_1)} \left(y_{t_1}^{(c_1)}\right)_{1\le t_1\le t}^{1\le c_1\le C}\right) \cdot P\left(s_{t+1}^{(c_2} | s_t^{(c_1)}\right) \cdot \\
&\quad P\left(\left(y_{t+1}^{(c_1)}\right)^{1\le c_1\le C} | s_{t+1}^{(c_2)}\right) \cdot P\left(\left(y_{t_1}^{(c_1)}\right)_{t+2\le t_1\le T}^{1\le c_1\le C} | s_{t+1}^{(c_2)}\right) \\
&= \alpha_t^{(c_1)} \cdot h_{s_t^{(c_1)} s_{t+1}^{(c_2)}}^{(c_1,c_2)} \cdot P\left(\left(y_{t+1}^{(c_1)}\right)^{1\le c_1\le C} | s_{t+1}^{(c_2)}\right) \cdot \beta_{t+1}^{(c_2)}
\end{aligned}$$

□

2.2.2 Parameter Estimation

Suppose the latent states at time $t = 1..T$ is already known $S_t = s_t^{(1)} \cdots s_t^{(C)}$. The likelihood function is

$$\begin{aligned}
& p\left(\left(y_t^{(c)}\right)_{1\le t\le T}^{1\le c\le C}\right) \\
= \; & \pi_{\vec{S_1}} \cdot \left(\prod_{t=1}^{T-1} g_{\vec{s}_t \to s_{\vec{t+1}}}\right) \cdot \left(\prod_{t=1}^{T} p(\vec{y}_t | \vec{s}_t)\right) \\
= \; & \left(\prod_{c=1}^{C} \pi_{s_1^{(c)}}^{(c)}\right) \cdot \left(\prod_{t=1}^{T}\prod_{c2=1}^{C}\sum_{c1=1}^{C} h_{s_t^{(c1)}, s_{t+1}^{(c2)}}^{(c1,c2)}\right) \cdot \left(\prod_{t=1}^{T}\prod_{c=1}^{C} p(y_t^{(c)} | s_t^{(c)})\right)
\end{aligned}$$

We can find new parameters and try to maximize the log likelihood function:

$$\begin{aligned}
& \log p\left(\left(y_t^{(c)}\right)_{1\le t\le T}^{1\le c\le C}\right) \\
= \; & \sum_{c=1}^{C} \log \pi_{s_1^{(c)}}^{(c)} + \sum_{t=1}^{T}\sum_{c2=1}^{C} \log \sum_{c1=1}^{C} h_{s_t^{(c1)}, s_{t+1}^{(c2)}}^{(c1,c2)} + \sum_{t=1}^{T}\sum_{c=1}^{C} \log p(y_t^{(c)} | s_t^{(c)}) \\
\ge \; & \sum_{c=1}^{C} \log \pi_{s_1^{(c)}}^{(c)} + \sum_{t=1}^{T}\sum_{c2=1}^{C}\sum_{c1=1}^{C} \log h_{s_t^{(c1)}, s_{t+1}^{(c2)}}^{(c1,c2)} + \sum_{t=1}^{T}\sum_{c=1}^{C} \log p(y_t^{(c)} | s_t^{(c)}) \qquad (2.4) \\
= \; & \sum_{c=1}^{C}\sum_{i=1}^{m_c} \delta(s_1^{(c)}, i) \cdot \log \pi_i^{(c)} + \qquad (2.5) \\
& \sum_{t=1}^{T-1}\sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}} \delta(s_t^{(c1)}, i) \cdot \delta(s_{t+1}^{(c2)}, j) \cdot \log h_{i,j}^{(c1,c2)} + \sum_{t=1}^{T}\sum_{c=1}^{C}\sum_{i=1}^{m_c} \delta(s_t^{(c)}, i) \cdot \log p(y_t^{(c)} | i) \\
\triangleq \; & \sum_{c=1}^{C}\sum_{i=1}^{m_c} \tilde{\pi}_i^{(c)} \cdot \log \pi_i^{(c)} + \\
& \sum_{c=1}^{C}\sum_{c1=1}^{C}\sum_{i=1}^{m_{c1}}\sum_{j=1}^{m_{c2}} \tilde{\xi}^{(c1,c2)}(i,j) \cdot \log h_{i,j}^{(c1,c2)} + \sum_{c=1}^{C}\sum_{i=1}^{m_c} \tilde{\gamma}^{(c)}(i) \cdot \log p(y_t^{(c)} | i)
\end{aligned}$$

where the step 2.4 is according to the Jensen's inequality, and the function $\delta(i,j) = \begin{cases} 1 & i = j \\ 0 & i \ne j \end{cases}$ is the Kronecker delta function. From 2.5, we know that $\tilde{\pi}_i^{(c)} = \delta(s_1^{(c)}, i)$, $\tilde{\xi}^{(c1,c2)}(i,j) = \sum_{t=1}^{T-1} \delta(s_t^{(c1)}, i) \cdot \delta(s_{t+1}^{(c2)}, j)$,

and $\tilde{\gamma}^{(c)}(i) = \sum_{t=1}^{T} \delta(s_t^{(c)}, i)$ are the sufficient statistics for $\pi_i^{(c)}$, $h_{i,j}^{(c1,c2)}$, and $p(y_t^{(c)}|i)$ respectively. We can maximize the parameters involved in the influence matrix H by equaling them to the corresponding sufficient statistics:

$$\pi_i^{(c)} = \tilde{\pi}_i^{(c)} \tag{2.6}$$

$$h_{i,j}^{(c1,c2)} = \frac{1}{C} \cdot \frac{\tilde{\xi}_{i,j}^{(c1,c2)}}{\sum_{j=1}^{m_{c2}} \tilde{\xi}_{i,j}^{(c1,c2)}} \tag{2.7}$$

We can maximize the parameters that map the latent states to the observations in the same way as in an ordinary hidden Markov model.

When the latent states at time $t = 1..T$ are not known. We can choose parameters that maximize the expected log likelihood function:

$$\begin{aligned}
& \mathtt{E}_{\vec{s}_1 \cdots \vec{s}_T} \left[\log p \left(\left(y_t^{(c)} \right)_{1 \le t \le T}^{1 \le c \le C} \right) \right] \\
= \; & \mathtt{E}_{\vec{s}_1 \cdots \vec{s}_T} \left[\sum_{c=1}^{C} \log \pi_{s_1^{(c)}}^{(c)} + \sum_{t=1}^{T} \sum_{c2=1}^{C} \log \sum_{c1=1}^{C} h_{s_t^{(c1)}, s_{t+1}^{(c2)}}^{(c1,c2)} + \sum_{t=1}^{T} \sum_{c=1}^{C} \log p(y_t^{(c)}|s_t^{(c)}) \right] \\
\ge \; & \mathtt{E}_{\vec{s}_1 \cdots \vec{s}_T} \left[\sum_{c=1}^{C} \log \pi_{s_1^{(c)}}^{(c)} + \sum_{t=1}^{T} \sum_{c2=1}^{C} \sum_{c1=1}^{C} \log h_{s_t^{(c1)}, s_{t+1}^{(c2)}}^{(c1,c2)} + \sum_{t=1}^{T} \sum_{c=1}^{C} \log p(y_t^{(c)}|s_t^{(c)}) \right] \\
= \; & \sum_{c=1}^{C} \sum_{i=1}^{m_c} \mathtt{E}_{s_1^{(c)}} \left[\delta(s_1^{(c)}, i) \right] \cdot \log \pi_i^{(c)} + \\
& \sum_{c=1}^{C} \sum_{c1=1}^{C} \sum_{i=1}^{m_{c1}} \sum_{j=1}^{m_{c2}} \sum_{t=1}^{T-1} \mathtt{E}_{s_t^{(c1)} s_{t+1}^{(c2)}} \left[\delta(s_t^{(c1)}, i) \cdot \delta(s_{t+1}^{(c2)}, j) \right] \cdot \log h_{i,j}^{(c1,c2)} + \\
& \sum_{c=1}^{C} \sum_{i=1}^{m_c} \sum_{t=1}^{T} \mathtt{E}_{s_t^{(c)}} \left[\delta(s_t^{(c)}, i) \right] \cdot \log p(y_t^{(c)}|i) \\
\triangleq \; & \sum_{c=1}^{C} \sum_{i=1}^{m_c} \tilde{\pi}_i^{(c)} \cdot \log \pi_i^{(c)} + \\
& \sum_{c=1}^{C} \sum_{c1=1}^{C} \sum_{i=1}^{m_{c1}} \sum_{j=1}^{m_{c2}} \tilde{\xi}^{(c1,c2)}(i,j) \cdot \log h_{i,j}^{(c1,c2)} + \sum_{c=1}^{C} \sum_{i=1}^{m_c} \tilde{\gamma}^{(c)}(i) \cdot \log p(y_t^{(c)}|i)
\end{aligned}$$

According to the attributes of the expectation operator and the Kronecker delta operator, the sufficient statistics are given in the following way, and the parameters related to the state transitions are maximized by Equations 2.6 and 2.7:

$$\begin{aligned}
\tilde{\pi}_i^{(c)} &= \gamma_i^{(c)} \\
\tilde{\xi}^{(c1,c2)}(i,j) &= \sum_{t=1}^{T=1} \xi_{t \to t+1}^{(c1,c2)}(i,j) \\
\tilde{\gamma}^{(c)}(i) &= \sum_{t=1}^{T} \gamma_t^{(c)}(i)
\end{aligned}$$

The parameters are re-estimated in the same way as in the known latent state case.

Algorithm 2 The EM algorithm for the latent structure influence model

E-Step

$$
\begin{aligned}
\vec{\alpha}_t^* &= \begin{cases} \vec{\pi}_{1\times\sum m_c} & t=1 \\ \vec{\alpha}_{t-1}\cdot H\cdot \texttt{diag}[\vec{b}_t] & t>1 \end{cases} \\
\mathcal{N}_t &= \begin{pmatrix} \frac{1}{\sum_{i=1}^{m_1}\vec{\alpha}_{t,i}^{*(1)}} & & \\ & \cdots & \\ & & \frac{1}{\sum_{i=1}^{m_C}\vec{\alpha}_{t,i}^{*(C)}} \end{pmatrix} \\
\vec{\alpha}_t &= \vec{\alpha}_t^*\cdot\mathcal{N}_t \\
\vec{\beta}_t &= \begin{cases} \vec{1}_{\sum m_c\times 1} & t=T \\ H\cdot\texttt{diag}[\vec{b}_t]\mathcal{N}_{t+1}\vec{\beta}_{t+1} & t<T \end{cases} \\
\vec{\gamma}_t &= \vec{\alpha}_t\cdot\texttt{diag}[\vec{\beta}_t] \\
\xi_{t-1\to t} &= \texttt{diag}[\vec{\alpha}_{t-1}]\cdot H\cdot\texttt{diag}[\vec{b}_t]\cdot\mathcal{N}_t\cdot\texttt{diag}[\vec{\beta}_t] \\
p(\vec{o}) &= \prod_{t,c}(\sum_{i=1}^{m_c}\vec{\alpha}_{t,i}^{*(c)})
\end{aligned}
$$

M-step

- Parameters related to the latent state transitions

$$
\begin{aligned}
A_{ij} &= \texttt{normalize}[\sum_{t=2}^{T}\xi_{t-1\to t}^{(i,j)}] \\
S &= \begin{pmatrix} \vec{1}_{1\times m_1} & & \\ & \cdots & \\ & & \vec{1}_{1\times m_C} \end{pmatrix} \\
d_{ij} &= \texttt{normalize}[S\sum_{t=2}^{T}\xi_{t-1\to t}S^{\mathtt{T}}] \\
\vec{\pi}^{(c)} &= \texttt{normalize}[\vec{\gamma}_1^{(c)}]
\end{aligned}
$$

Parameters related to multinomial observations

$$
p^{(c)}(\vec{o}^{(c)}|\vec{s}^{(c)}) = \texttt{normalize}[\sum_t \vec{\gamma}_t^{(c)\,\mathtt{T}}\cdot\vec{\delta}_{y_t,\vec{o}^{(c)}}]
$$

Parameters related to Gaussian observations

$$
\begin{aligned}
\mu^{(c)} &= \frac{\sum_t \vec{\gamma}_t^{(c)\,\mathtt{T}}\cdot\vec{y}_t^{(c)}}{\sum_t\vec{\gamma}_t^{(c)}\cdot\vec{1}_{m_c\times 1}} \\
\Sigma_i^{(c)\,2} &= \frac{\sum_t \gamma_{t,i}^{(c)}\vec{y}_t^{(c)}\vec{y}_t^{(c)\,\mathtt{T}}}{\sum_t\vec{\gamma}_t^{(c)}\cdot\vec{1}_{m_c\times 1}} - \vec{\mu}_i^{(c)}\cdot\vec{\mu}_i^{(c)\,\mathtt{T}}
\end{aligned}
$$

2.2.3 Viterbi-decoding of the Influence Modeling

Theorem 5. *Finding a Viterbi path for an influence process is NP-complete*

Proof. We prove this theorem in 2 steps. We first show that given a hard

assignment for all latent variables on all processes at time $t+1$ (i.e. $s^{(c)}_{t+1,\,i},\ 1 \le c \le C,\ 1 \le i \le m_c$ is either 0 or 1), finding a hard assignment for all latent variables on all processes at time t (i.e. $s^{(c)}_{t,\,i},\ 1 \le c \le C,\ 1 \le i \le m_c$ is either 0 or 1) is NP-complete. We then show that we have output observations together with output matrices that fix $\vec{s}_{t+1}$ while do not fix $\vec{s}_t$.

(Given $\vec{s}_{t+1}$, finding $\vec{s}_t$ to maximize $p(\vec{s}_{t+1}|\vec{s}_t)$ is NP-complete) We reduce from the SATISFIABILITY problem to an influence process with 2 time steps (step t, and step $t+1$), such that when the conjunction normal form under consideration is satisfiable, we have a path $\vec{s}_t\vec{s}_{t+1},\ where\ s^{(c)}_{t+1} = 2$ from some unknown start state $\vec{s}_t$ to a specific state $\vec{s}_{t+1}$ with probability greater than 0 for the constructed influence process.

Following the notation of [15] A9.1[L01] SATISFIABILITY, we have Set $U = \{u_1, .., u_M\}$ of variables, collection $C = \{\{\tilde{u}_{i_1}, \tilde{u}_{i_2}, .., \tilde{u}_{i_m}\}_{i=1..N}\}$ of clauses over U, where $\tilde{u}$ means u or or its negation $\bar{u}$. We know that the SATISFIABILITY problem in general is NP-complete.

We construct the influence process $H = D \otimes A = (d_{ij}A_{ij})$ as follows: H has $max(M, N)$ chains, where M is the number of boolean variables and N is the number of clauses. Each $d_{ij}A_{ij}$ is a 2×2 matrix. The evaluation of each boolean variable $u_i$to true or false is represented by selecting either the first row or the second row of $d_{ij}A_{ij}$. After we have chosen one row for each i , we sum the rows together and get a row matrix. Then we choose one entry for each j ,and multiply the entries together. The result is the probability ([12] (5.11)). We arrange the influence matrix in the following way such that when the conjunction normal form is satisfiable, we have a path through: When the number clauses is less than the number of variables, i.e., $M > N$, we fill all entries in $d_{ij}A_{ij}$ with positive values, such that evaluation of boolean variables won't "block" the path. When variable u_i does not appear in clause c_j , we put 0 into $d_{ij}A_{ij}$ such that evaluation of variable u_i does not provide a path through. when $\bar{u_i}$ appears in clause c_j , we set $d_{ij}A_{ij}$ to be $d_{ij}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},\ d_{ij} > 0$,such that the evaluation $u_i = 0$ provides a path through.when u_i appears in clause c_j , we set $d_{ij}A_{ij}$ to be $d_{ij} \cdot \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},\ d_{ij} > 0$,such that the evaluation $u_i = 1$ provides a path through.

To show that there exists an non-deterministic polynomial solution, we note that we can solve all $p(\vec{s}_{t+1}|\vec{s}_t)$ for all $\vec{s}_t$ and get the maximum of these joint probabilities.

(We can provide observations $\vec{o}_t, \vec{o}_{t+1}$, and output matrix $B^{(c)}$ to fix the latent state $\vec{s}_{t+1}$ while leaving $\vec{s}_t$ undetermined) We set the output matrix $B^{(c)} = \begin{pmatrix} 1 & 0 \\ .5 & .5 \end{pmatrix}$. That is to say, if $s^{(c)} = 1$, then $o^{(c)} = 1$. If $s^{(c)} = 2$, then there is a 50% chance that $o^{(c)} = 1$, and a 50% chance that $o^{(c)} = 2$. To look it in another way, if $o^{(c)} = 2$, then $s^{(c)} = 2$. If $o^{(c)} = 1$, then there is a 67% chance that $s^{(c)} = 1$, and 33% chances that $s^{(c)} = 2$. Now we stipulate that $o^{(c)}_t = 1,\ o^{(c)}_{t+1} = 2$. □

We use the following example to illustrate how the corresponding influence matrix for a given conjunction normal form is constructed.

Example 5. *Let* $U = \{u_1, u_2, u_3, u_4\}$ *, and* $C = \{\{u_1 \vee \bar{u_2} \vee u_3\}, \{\bar{u_1}\}, \{u_2 \vee \bar{u_2}\}, \}$

. The constructed influence matrix looks like this:

$$H = \begin{pmatrix} \frac{1}{3} & & & 1 & & & \frac{1}{4} & \frac{1}{4} \\ & \frac{1}{3} & 1 & & & & \frac{1}{4} & \frac{1}{4} \\ & \frac{1}{3} & & & \frac{1}{2} & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{3} & & & & \frac{1}{2} & \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{3} & & & & & & \frac{1}{4} & \frac{1}{4} \\ & \frac{1}{3} & & & & & \frac{1}{4} & \frac{1}{4} \\ & & & & & & \frac{1}{4} & \frac{1}{4} \\ & & & & & & \frac{1}{4} & \frac{1}{4} \end{pmatrix}.$$

Let's set $\vec{s}_1 = [1\ 0|0\ 1|0\ 1|0\ 1]$ *, i.e. we set* $u_1 = 0, u_2 = 1, u_3 = 1, u_4 = 1$ *, after we add the corresponding rows together, we have* $\sum_{i=1,4,6,8} H_{i,*} = [\frac{1}{3}\ \frac{2}{3}|0\ 1|\frac{1}{2}\ \frac{1}{2}|1\ 1]$ *. This evaluation satisfies* C *since* $\vec{s}_2 = [0\ 1|0\ 1|0\ 1|0\ 1]$ *is a solution. When we set* $\vec{s}_1 = [0\ 1|0\ 1|0\ 1|0\ 1]$ *, we have* $\sum_{i=1,4,6,8} H_{i,*} = [\frac{1}{3}\ \frac{2}{3}|1\ 0|\frac{1}{2}\ \frac{1}{2}|1\ 1]$ *, we know it is not a solution since* $s^{(2)}(2) = 0$ *.*

Chapter 3

Experimental Results

In this section, we illustrate how an influence model can capture the correlations among different dynamic processes and thus improve the overall dynamic inference performance. We give three examples. In the example of interacting processes with noisy observations, we illustrate the structure that an influence model tries to capture, and how an influence model can be used to improve classification precision. We then extend the noisy body sensor net example and compare the training errors and the testing errors of different dynamic models. In the social network example, we illustrate how the cell phone usage information can be used to recover geographical information as well as the social structure of the cellphone users.

3.1 Interacting Processes with Noisy Observations

Let us suppose that we have six stochastic processes, and we sample these six processes with six sensors. Each process can be either signaled (one) or non-signaled (zero) at any time, and the corresponding sensor has approximately 10% of its samples flipped. The interaction of the six stochastic processes behind the scene looks like this: processes one through three tend to have the same states; processes four through six tend to have the same states; the processes are more likely to be non-signaled than to be signaled; and the processes tend to stick to their states for a stretch of time. The parameters of the model are given as the following and are going to be estimated: $A_{ij} = \begin{pmatrix} .99 & .01 \\ .08 & .92 \end{pmatrix}, 1 \leq i, j \leq 6$, $B_i = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}, 1 \leq i \leq 6$, $d_{ij} = .33, 1 \leq i, j \leq 3$, and $d_{ij} = .33, 4 \leq i, j \leq 6$.

In Figure 3.1, (a) shows the sampled latent state sequences, (b) shows the corresponding observation sequences, (c) shows the influence matrix reconstructed from sampled observation sequences, and (d) shows the reconstructed latent state sequences after 300 observations. The $(i, j)^{th}$ entry of the $(c_1, c_2)^{th}$ sub-matrix of an influence matrix determines how likely that process c_1 is in state i at time t and process c_2 is in state j at time $t + 1$. It can be seen from Figure 3.1 (c) that the influence model computation recovers the structure of the interaction.

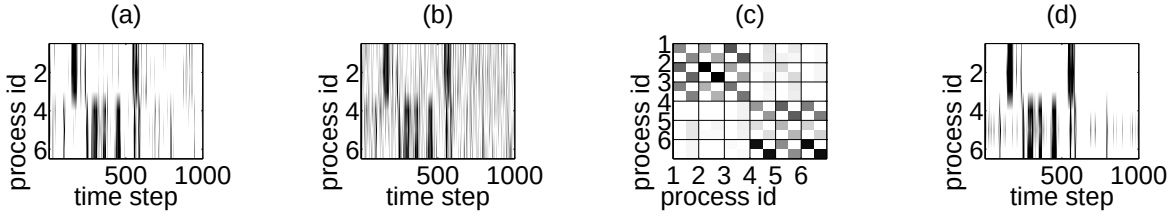

Figure 3.1: Inference from observations of interacting dynamic processes.

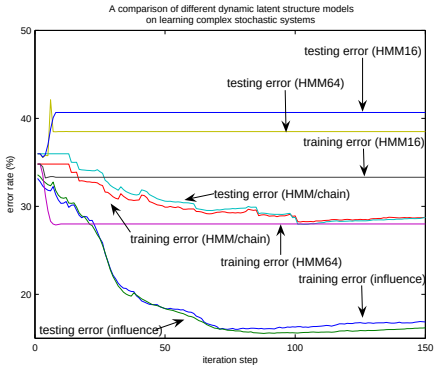

Figure 3.2: Comparison of dynamic models.

The influence model can normally attain around 95% accuracy in predicting the latent states for each process. The reconstructed influence matrix has only 9% relative differences with the original one. Using only observations of other chains we can predict a missing chain's state with 87% accuracy.

We then constructed a more complex experimental setting to compare the performances of different types of hidden Markov models. In this setting, we have a Markov process with 2^C, where $C = 10$, number of states and a randomly generated state transition matrix. Each system state $\vec{s}_t$ is encoded into a binary $s_t^{(1)} \cdots s_t^{(C)}$. Each of the $m_c = 2$ evaluations of "bit" $s_t^{(c)}$corresponds a different 1-d Gaussian observation $o_t^{(c)}$: Digit $s_t^{(c)} = 1$ corresponds to $o_t^{(c)} \sim \mathcal{N}[\mu_1 = 0, \sigma_1^2 = 1]$; Digit $s_t^{(c)} = 2$ corresponds to $o_t^{(c)} \sim \mathcal{N}[\mu_2 = 1, \sigma_2^2 = 1]$. Figure 3.2 compares the performances of several dynamic latent structure models applicable to multi-sensor systems. Of the 1000 samples $(\vec{o}_t)_{1 \le t \le 100}$, we use the first 250 for training and all 1000 for validation.

There are two interesting points. First, the logarithmically scaled number of parameters of the influence model allows us to attain high accuracy based on a relatively small number of observations. This is because the eigenvectors of the master Markov model we want to approximate are either mapped to the eigenvectors of the corresponding influence model, or mapped to the null space of the corresponding event matrix thus is not observable from the influence model, and that in addition the eigenvector with the largest eigenvalue (i.e., 1) is mapped to the eigenvector with the largest eigenvalue of the influence matrix [12]. Secondly, both the influence model and the hidden Markov model applied to individual processes are relatively immune to over-fitting, at the cost of low convergence rates. This situation is intuitively the same as the numerical analysis wisdom that a faster algorithm is more likely to converge to a local extremum or to diverge.

3.2 Real-time Context Recognition

An early version of the Life-Wear real-time context recognition system was developed by Blum [10], and is comprised of a Sharp Zaurus PDA, an ambient audio recorder, and two accelerometers, worn on hip and wrist. This system is designed to classify in real time eight locations, six speaking/non-speaking status, six postures, and eight activities. The classification is carried out in two steps: A pre-classifier (single Gaussian, mixture of Gaussians, or C4.5) is first invoked on the audio and accelerometer features to get a moderate pre-classification result of the above four categories.

The pre-classification result of different categories is then fed into an influence model to learn inter-sensor structure, and then this learned structure is used to generate an improved post-classification result. In this example the influence model learns the conditional probabilities that relate the four categories (location, audio, posture, and activity) and then uses this learned influence matrix to improve the overall performance.

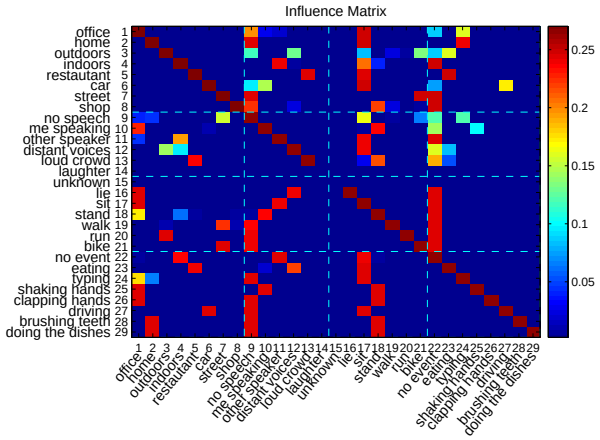

Figure 3.3: Influence matrix learned by the EM algorithm.

For example, given that the Life-Wear user is typing, we can inspect the row of the influence matrix corresponding to "typing" and see that this person is very likely to be either in the office or at home, to not be speaking, and to be sitting. As a result, the action of typing can play a critical role to disambiguating confusions between sitting and standing, or between speaking vs not-speaking, but not between office and home.

By combining evidence across different categories using the influence model, the classification errors for locations, speaking/non-speaking, postures, and activities decreased by an average of 23%, from 38%, 22%, 8% and 27% to 28%, 19%, 8%, and 17% respectively. The post-classification for postures does not show significant improvement because of two reasons: (1) it is already precise enough considering that we have labeling imprecision in our training data and testing data, and (2) it is the driving force for improving the other categories, and no other categories are more certain than the posture category.

3.3 Speaker identification of a group meeting

Automatic speaker identification is important for documenting the meeting audio recordings. When combined with the speech recognition technique, we can generate automatic meeting reviews.

The speaker identification algorithm to be discussed in this section uses the audio recordings for individual meeting participants. It assumes that a meeting proceeds as a finite state Markov chain involving its interacting participants, and each state describes a different configuration of who is speaking. The Markov chain is latent and to be inferred. It also assumes that the energy distributions for the audio recordings are random variables conditioned on the latent states of the Markov chain.

The specification of the latent Markov chain needs careful consideration, since the number of states for even a moderate-sized meeting can be very large. For example, let us assume that we have n participants in a meeting, and each participant can only be in two states: speaking, and non-speaking. The number of possible states for the meeting is then 2^n. On the other hand, we noticed that human interactions such as in a meeting is highly structured: there is normally one person speaking in a meeting session. The situations where all people speak are rare and normally unimportant. Thus, instead of working with a hidden Markov process with all 2^n possible sates, we work with a dual latent structure influence process with only $n + 1$ number of states: one for each speaker and the rest for all other situations.

The meeting data we are going to analysis was recorded from a rather casual group meeting in 12/21/2004. It lasts for 44 minutes, from a clapping at the beginning of the meeting to another clapping at the end of

the meeting. More than 99% of the meeting time was occupied by seven persons: one advisor (labeled as sandy1221), and six students (labeled as msung1221, ron1221, wen1221, juan, jon, and anmol). We assigned the four microphones to the following people: msung1221, ron1221, sandy1221, and wen1221, and recorded the meeting from their angle. When one of the four persons was speaking, we would expect his microphone to pick up a louder voice than the other three microphones. When one of the other three speakers was speaking, we would expect different energy distributions among the four microphones due to the relative positions of the three speakers and the four microphones.

The speaking/non-speaking status of different persons are hand-labeled and illustrated in Figure 3.4. The normalized energy distributions among the four audio channels in one- second-window bins (the features we use) are plotted in Figure 3.5. Our goal in this section is only to illustrate an application of the latent state influence model. The readers can choose more advanced features, such as the pitches for different speakers, in other similar applications to attain better accuracy.

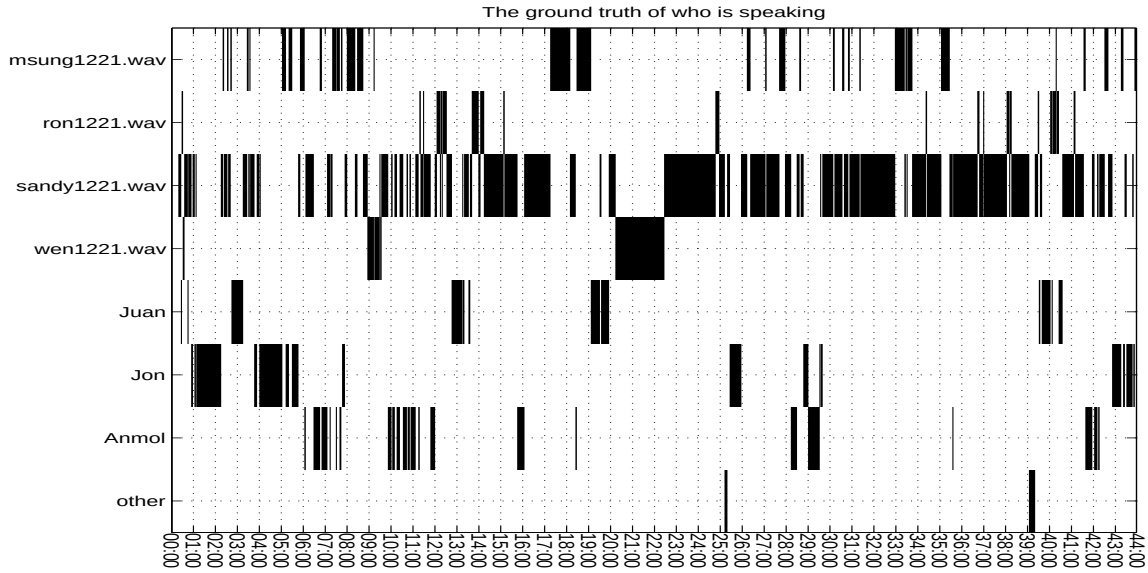

Figure 3.4: The ground truth of the speaking/non-speaking status of different persons in a meeting.

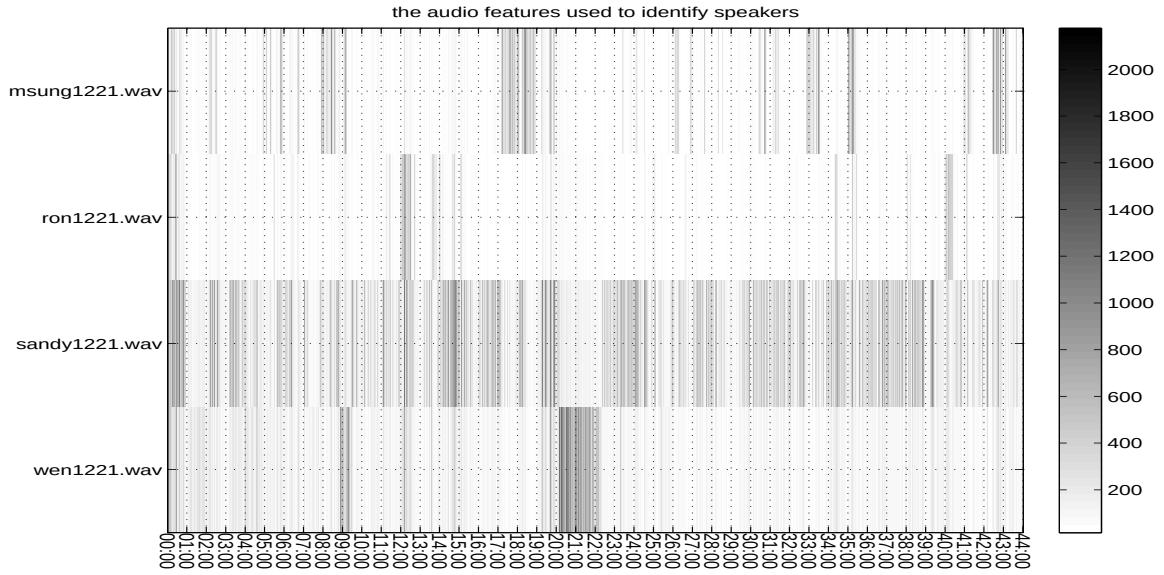

Figure 3.5: The features used to identify the speakers.

This section proceeds in the following order. First, we discuss and compare different approaches to align the audio sequences. Then, we segment the meeting by speakers with various unsupervised methods (k-means, mixture of Gaussians, hidden Markov modeling, and the influence modeling). The influence model out-performs the other models, since it better captures the structure of the speaker identification problem. Next, we segment the meeting by speakers with various supervised methods (mixture of Gaussians, hidden Markov modeling, and the influence modeling). Again, the influence model out-performs the other models. We conclude this section by a discussion of why the influence modeling fits the structure of meeting interactions better than the other models used in this section.

In situations where the signal sequences have a common reference point (such as a global time stamp), and the signal sequences do not have jitters, the sequence alignment issue is simple. However, in many interesting applications, we do not have such a global time stamp for us to precisely and trivially alignment data. For example, the signal sequences collected by embedded devices are often not synchronized, due to the simplicity requirement of those devices. In such scenarios, we need to consider the cross-sequence statistical characteristics to design an aligning algorithm.

We assume that the sound signals picked up by different microphones are linear combinations of the sound sources (the persons). The weights of the sound sources are different from microphone to microphone, due to the different relative positions of the sound sources and the microphones. We segment each of the four sequences into five-minute segments, and shift corresponding segments to attain maximum cross correlations. The cross correlation of either the low-pass filtered sound waves or the spectrogram amplitudes of the sound waves are used. We attain very close results in both approaches. The aligned wave sequences are then overlapped (summed) together and played back to judge the quality of the alignment. They are aligned very well and we cannot hear discrepancies in the 44 minutes span.

We first show how to separate the four speakers wearing microphones using unsupervised learning techniques. The assumption is that when one/none of the four speakers who wore a microphone was speaking, the energy distributions among the four audio channels were significantly different to reveal who was speaking. The speaking/non-speaking classifier can work in the following way. The samples were first captured with five (number of speakers who wore microphones + 1) clusters. A cluster can then be assigned to a speaker if the channel corresponding to this speaker has the most energy statistically.

The (unsupervised) classification results with the k-means algorithm, the Gaussian mixture models, the

hidden Markov model for all audio channels, one hidden Markov model per audio channel, and the latent state influence model are respectively plotted in Figures 3.6, 3.7, 3.8, 3.9, 3.10, and 3.11.

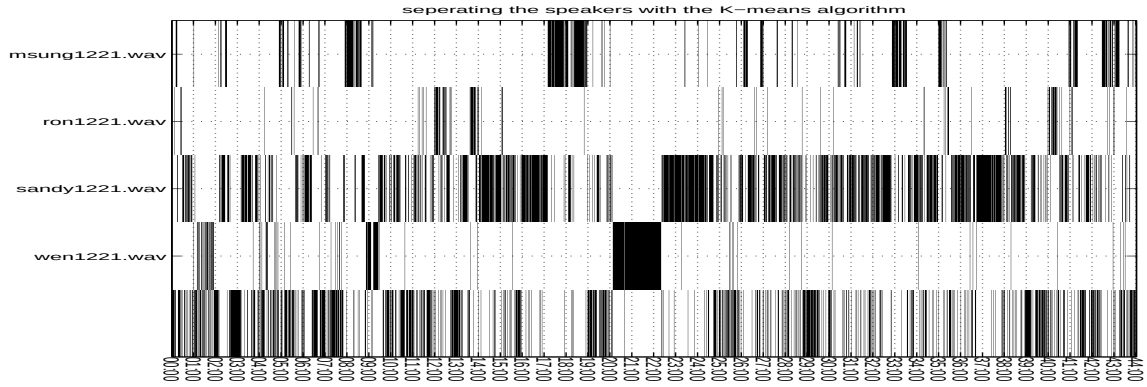

Figure 3.6: Unsupervised speaking/non-speaking classification result with the k-means algorithm

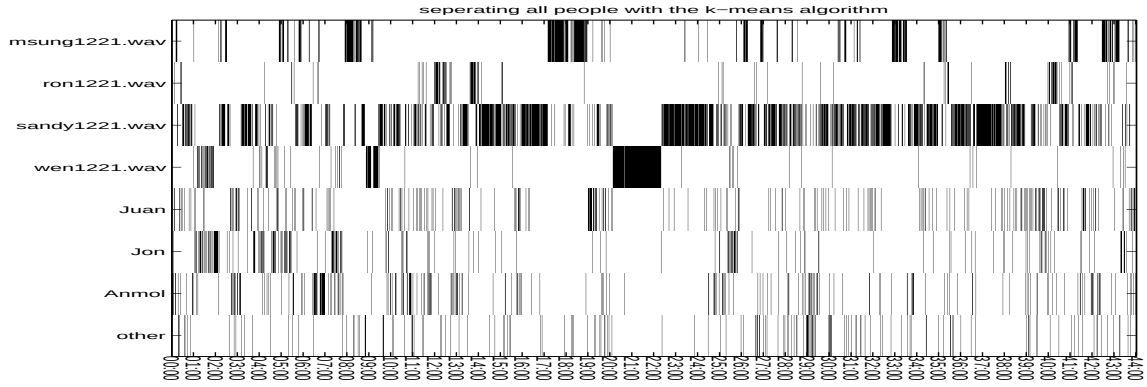

Figure 3.7: Unsupervised speaking/non-speaking classification for all speakers with the k-means algorithm.

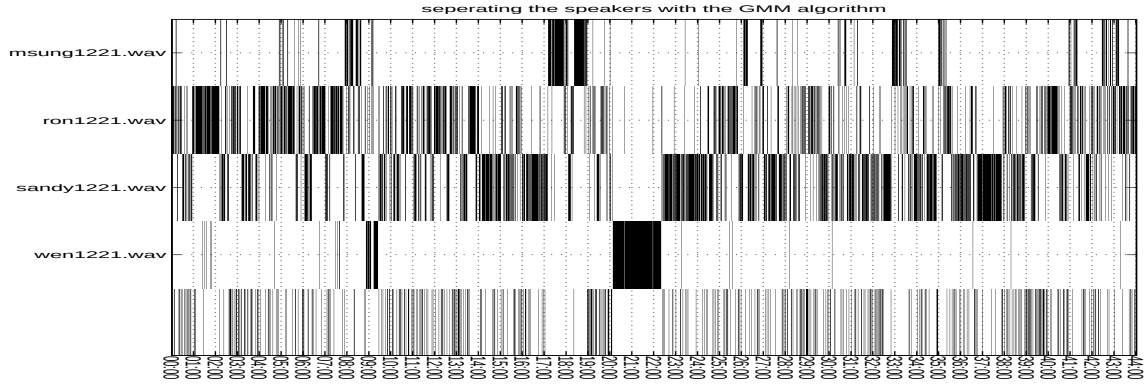

Figure 3.8: Unsupervised speaking/non-speaking classification result with the Gaussian mixture model

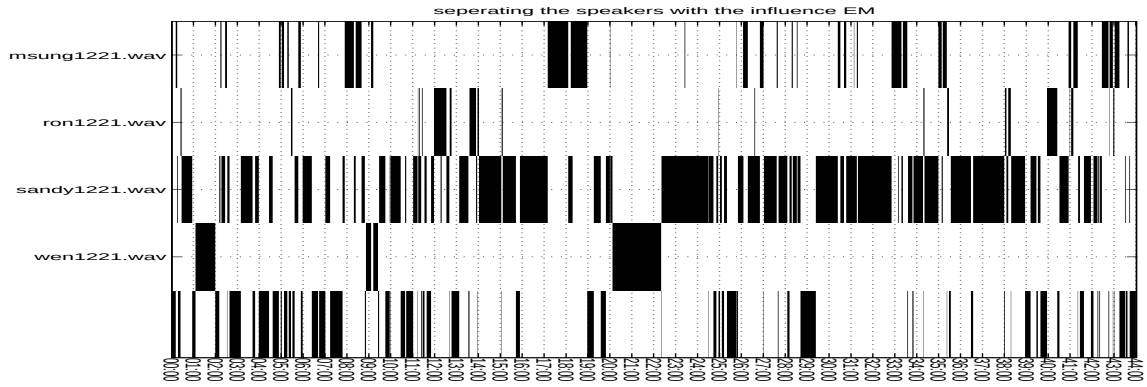

Figure 3.11: Unsupervised speaking/non-speaking classification result with the latent state influence process

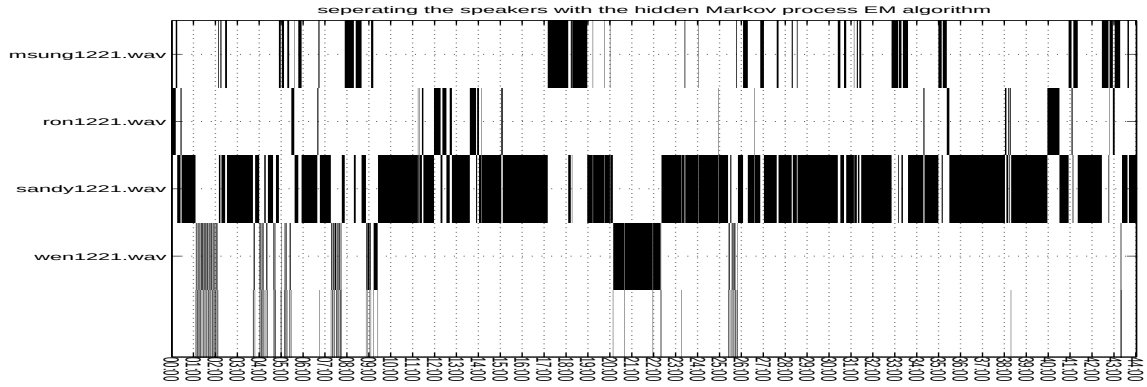

Figure 3.9: Unsupervised speaking/non-speaking classification result with one hidden Markov process

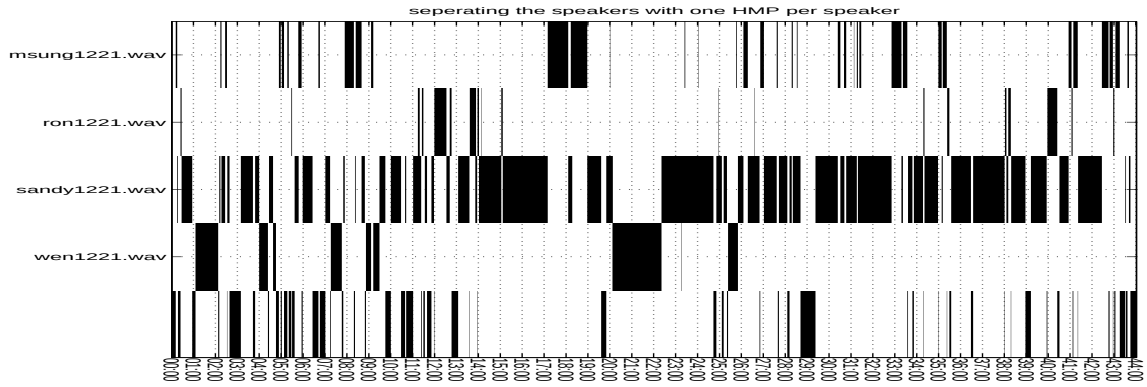

Figure 3.10: Unsupervised speaking/non-speaking classification with one HMP per speaker

The (supervised) classification results are plotted in Figure 3.12 and 3.13.

In order to understand why the influence model based classifier out-performs the other classifiers, it is important to notice that the influence model combines the evidence from different channels in the "concept" level, i.e., when speaker A is speaking, speaker B is very likely to be non-speaking. In comparison, the other classifiers either combine the evidence from different channels in the feature level or do not combine evidence from different channels at all.

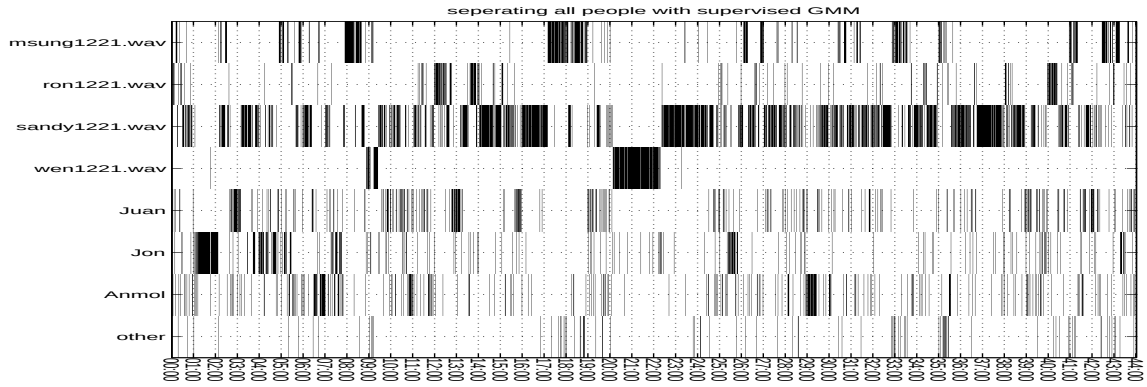

Figure 3.12: Supervised speaking/non-speaking classification result with the GMM algorithm

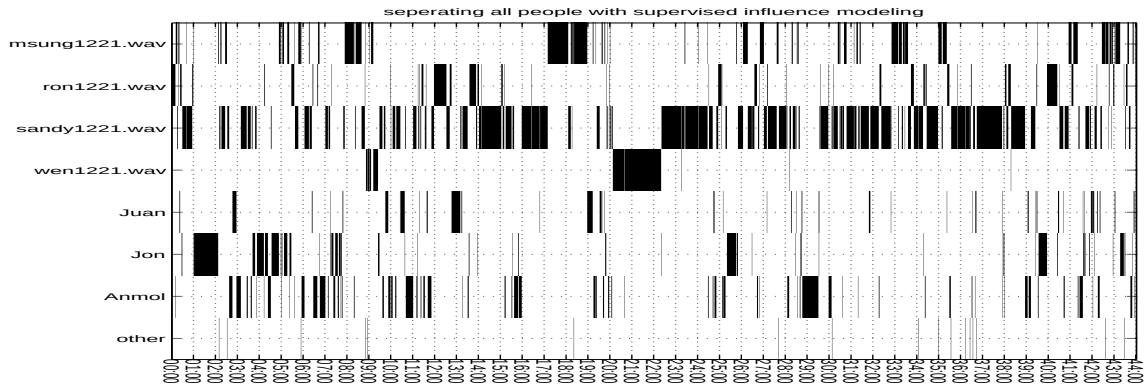

Figure 3.13: Supervised speaking/non-speaking classification result with the influence model

3.4 Social Network Example

The social network example demonstrates reconstructing the social structure of a set of subjects from their cellphone-collected data [8]. In this data 81 subjects wore Bluetooth-enabled mobile telephones that recorded which cell towers were visible to the telephone, thus allowing coarse estimation of the wearers' location, and which Bluetooth devices are nearby, thus allowing inference of proximity to other subjects. Note that Bluetooth signals include a unique identifier, and are typically detectable at a range of only a few meters. In this study fixed Bluetooth beacons were also employed, allowing fairly precise estimation of subjects location even within buildings. Over the nine months of the study 350,000 hours of data were recorded.

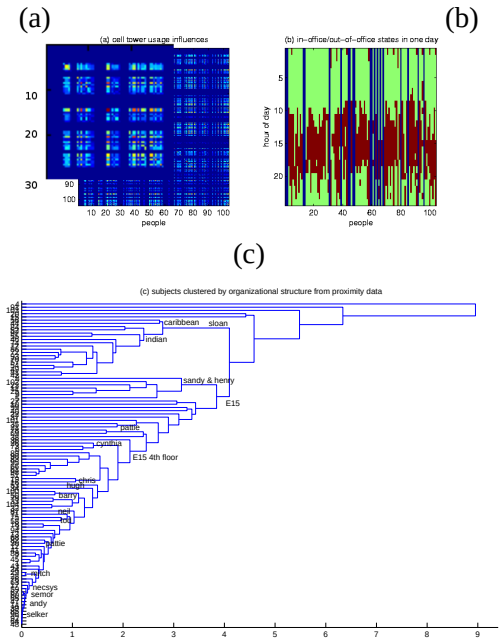

Figure 3.14: Finding social structures from cellphone-collected data. (a) New students and faculty are outliers in the influence matrix, appearing as red dots due to large self-influence values. (b) Most People follow regular patterns (red: in office, green: out of office, blue: no data), (c) clustering influence values recovers work-group affiliation with high accuracy (labels show name of group).

The temporal evolution of these observations was analyzed using the influence model with 81 chains, corresponding to the 81 subjects. Each subjects' chain was constrained to have two latent states ("work", "home") but with no restriction on social network connectivity.

In our first experiment with this data the observation vector for each chain was restricted to the cell tower visibility of each subjects' 10 most commonly seen cell towers. In the resulting model, the two states for each subject corresponded accurately to 'in the office' and 'at home', with other locations being misclassified. The resulting influence matrix, shown in Figure 3.14 (a), demonstrated that most people follow very regular patterns of movement and interpersonal association, and consequently we can predict their actions with substantial accuracy from observations of the other subjects. A few of the chains were highly independent and thus not predictable. These chains corresponded to new students, who had not yet picked up the rhythm of the community, and the faculty advisers, whose patterns are shown to determine the patterns of other students.

In another setup, we used the Bluetooth proximity distribution as our observations. Again, the latent states accurately reflect whether a person is at home of in office. However with this data the resulting influence matrix shows precisely the social and geometrical structure of the subjects. The dendrogram from the proximity influence matrix shown in Figure 3.14 (b) captures the actual organization of the laboratory, clustering people into their actual work groups with only three errors. For comparison, a clustering based on direct correlations in the data has six errors.

Chapter 4

Conclusions

The current thesis has described my work in modeling and understanding the dynamics of human related behaviors. Those dynamics are characterized by large state spaces and highly structured interactions. The latent structure influence process models the dynamics of human related behaviors by emulating how a group of persons help each other to understand complex processes.

The latent structure influence process is a state-space model, and its inference algorithm alternates between two steps: time updating (guessing what is going to happen next), and measure updating (incorporating new evidence). In this thesis, I have inspected two different approaches for measure updating. In the linear approach, the individuals in the team of persons update their guessing by averaging over each other's evidence. In the non-linear approach, the individuals only care about their own evidence. I derived the forward-backward algorithm, and the parameter maximization algorithm for each measure updating method. I also proved that finding a Viterbi path for a latent structure influence process with the non-linear measure updating strategy is NP-complete.

The influence modeling is applied to an imaginary network of power plants. With the latent structure influence process, we can have a more accurate understanding of the individual power plants' normal/failed states by looking at each other's states. We can compute how the power plants are connected at the same time.

We also used the latent structure influence process to segment speakers in a group meeting, to infer a person's location, posture, speaking/non-speaking status, and event, and to infer the structure of a cellphone user community. In all three applications, we either get better accuracy, or get invaluable insights that could not be derived without the influence modeling.

There are several unanswered questions, and I am willing to pursue their answers. (1) When monitoring a general complex process, in what situation(s) does a group of cooperating persons perform better than an individual? (2) The current latent structure influence process only models the degree of agreement of any two persons in a in a cooperating group. How can we model the situation that two persons agree in one thing but disagree in another, and what computational performance does the extension incur? (3) Theoretically, what dynamic processes are suitable the hidden Markov process and the latent structure influence process, and how many information do they capture respectively ?

Bibliography

[1] Sumit Basu. *Conversational Scene Analysis*. PhD thesis, MIT, 2002.

[2] Kevin Murphy. The bayes net toolbox for matlab, 2001. URL `http://bnt.sourceforge.net/`.

[3] Yariv Ephraim and Neri Merhav. Hidden markov process. *IEEE Transactions on Information Theory*, 48(6):1518–1569, 2002.

[4] Oliver Cappe. Ten years of hmms. URL `http://www.tsi.enst.fr/~cappe/docs/hmmbib.html`.

[5] L. E. Baum and J. A. Eagon. Statistical inference for probabilistic functions of finite state markov chains. *Ann. Math. Statist.*, 37:1554–1563, 1966.

[6] L. E. Baum, T. Petri, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *Ann. Math. Statist.*, 41:164–171, 1970.

[7] Brian D. O. Anderson. The realization problem for hidden markov models. *Mathematics of Control, Signals, and Systems*, 12:80–120, 1999.

[8] Nathan Eagle and Alex Pentland. Reality mining: Sensing complex social systems. *Journal of Personal and Ubiquitous Computing*, 10:255–268, 2006.

[9] DARPA. Assist proposer information pamphlet, 2004. URL `http://www.darpa.mil/ipto/solicitations/open/04-38_PIP.htm`.

[10] Mark Blum. Real-time context recognition. Master's thesis, Department of Information Technology and Electrical Engineering, Swiss Federal Institute of Technology Zurich (ETH), 2005.

[11] Nuria M. Oliver, Barbara Rosario, and Alex Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):831–843, 2000.

[12] Chalee Asavathiratham. *The Influence Model: A Tractable Representation for the Dynamics of Networked Markov Chains*. PhD thesis, MIT, 1996.

[13] Sumit Basu, Tanzeem Choudhury, Brian Clarkson, and Alex Pentland. Learning human interactions with the influence model. Technical report, MIT Media Laboratory Vision & Modeling Technical Report #539, 2001. URL `http://vismod.media.mit.edu/tech-reports/TR-539.pdf`.

[14] James Stoner. A comparison of individual and group decisions involving risk. Master's thesis, MIT, 1961.

[15] Michael Garey and David Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY: W. H. Freeman and Company, 1979.