隐式马尔可夫模型（HMM）

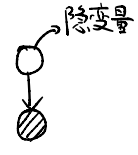
一、背景

概率图模型
- 有向 - Bayesian Network
- 无向 - Markov Random Field (Markov Network)

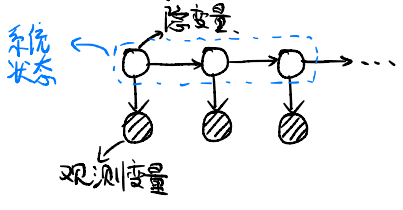
+time → Dynamic Model
- HMM
- Kalman Filter
- Particle Filter

Dynamic Model 的样本之间不是 iid 的

GMM：



如果在 GMM 上加上时间：

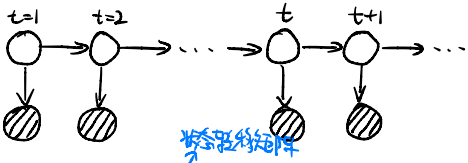


动态模型
- time 横向
- mixture 纵向

根据系统状态的连续与否，state → 离散 = HMM

连续
- 线性 → Kalman Filter
- 非线性 → Particle Filter

HMM



一个模型，两个假设，三个问题

i：模型参数：$\lambda=(\pi, A, B)$

π：初始概率分布　A：状态转移矩阵　B：发射矩阵

取值集合

观测变量：$o_1, o_2, \cdots, o_t, \cdots \longrightarrow V=\{v_1, v_2, \cdots v_m\}$

状态变量：$i_1, i_2, \cdots, i_t, \cdots \longrightarrow Q=\{q_1, q_2, \cdots, q_N\}$

对于模型参数，其中，

$A=[a_{ij}]$，$a_{ij}=P(i_{t+1}=q_j \mid i_t=q_i)$，即 a_{ij} 表示从状态 q_i 转移到状态 q_j 的概率。

$B=[b_j(k)]$，$b_j(k)=P(o_t=v_k \mid i_t=q_j)$，即 $b_j(k)$ 表示 t 时刻从状态 q_j 产生 v_k 的概率，

π 为初始概率分布，即 $i_1=q_1, q_2, \cdots, q_N$

$\pi=(p_1, p_2, \cdots, p_N)$

ii：两个假设：

①齐次 Markov 假设　②观测独立假设

↓

无后效性

① $P(i_{t+1} \mid i_t, i_{t-1}, \cdots i_1, o_t, o_{t-1}, \cdots, o_1)=P(i_{t+1} \mid i_t)$

② $P(o_t \mid i_t, i_{t+1}, \cdots i_1, o_t, o_{t-1}, \cdots, o_1)=P(o_t \mid i_t)$

iii：三个问题

① Evaluation：已知 λ，$P(O \mid \lambda)$ 出现的概率（O：观测序列 $O=o_1, o_2, \cdots, o_T$），常用算法：前向后向

② Learning：λ 如何求？$(\lambda=\arg\max P(O \mid \lambda))$ → EM，Baum Welch

☆ ③ Decoding：已知观测序列 O，状态序列 I 出现的最大概率？（后验概率）

即 $I=\arg\max_I P(I \mid O)$，维特比算法

（I：状态序列 $I=i_1, i_2, \cdots i_T$）

引申问题：{预测：$P(i_{t+1} \mid o_1, o_2, \cdots, o_t)$；滤波：$P(i_t \mid o_1, o_2, \cdots, o_t)$}

问题一. Evaluation：Given λ，求 $P(O|\lambda)$

$$P(O|\lambda)=\sum_I P(I,O|\lambda)=\sum_I P(O|I,\lambda)\cdot P(I|\lambda)$$

其中，$P(I|\lambda)=P(i_1,i_2,\cdots,i_T|\lambda)=P(i_T|i_1,i_2,\cdots,i_{T-1},\lambda)\cdot P(i_1,i_2,\cdots,i_{T-1}|\lambda)$

由齐次Markov性 → $P(i_T|i_{T-1})=a_{i_{T-1},i_T}$；递归，是T-1规模的，所以是T规模的

$$=a_{i_{T-1},i_T}\cdot a_{i_{T-2},i_{T-1}}\cdots a_{i_1,i_2}\cdot\pi(i_1)$$

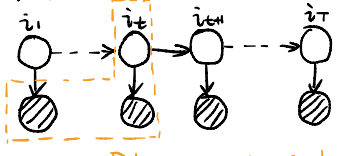
$$=\pi(i_1)\prod_{t=2}^{T}a_{i_{t-1},i_t}$$

$$P(O|I,\lambda)=\prod_{t=1}^{T}b_{i_t}(o_t)$$

$$\therefore P(O|\lambda)=\sum_I\pi(i_1)\prod_{t=2}^{T}a_{i_{t-1},i_t}\cdot\prod_{t=1}^{T}b_{i_t}(o_t)=\sum_{i_1}\sum_{i_2}\cdots\sum_{i_T}\pi(i_1)\prod_{t=2}^{T}a_{i_{t-1},i_t}\cdot\prod_{t=1}^{T}b_{i_t}(o_t)$$

对于每一个求和符号，每一个 i 都有N种选择，共有T个 i，因此时间复杂度为 $O(N^T)$，计算量很大，因此需要一种算法来计算。

Forward Algorithm：



记 $\alpha_t(i)=P(o_1,\cdots o_t,i_t=q_i|\lambda)$

$$\alpha_T(i)=P(O,i_T=q_i|\lambda)$$

则 $P(O|\lambda)=\sum_{i=1}^{N}P(O,i_T=q_i|\lambda)=\sum_{i=1}^{N}\alpha_T(i)$

$$\alpha_{t+1}(j)=P(o_1,o_2,\cdots,o_{t+1},i_{t+1}=q_j|\lambda)$$

$$=\sum_{i=1}^{N}P(o_1,o_2,\cdots,o_{t+1},i_{t+1}=q_j,i_t=q_i|\lambda)$$

$$=\sum_{i=1}^{N}P(o_{t+1}|o_1,o_2,\cdots o_t,i_{t+1}=q_j,i_t=q_i,\lambda)\cdot P(o_1,\cdots o_t,i_t=q_i,i_{t+1}=q_j|\lambda)$$

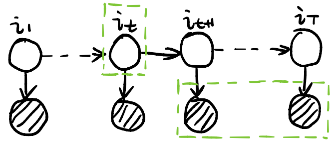
$$=\sum_{i=1}^{N}P(o_{t+1}|i_{t+1}=q_j)\cdot P(o_1,\cdots,o_t,i_t=q_i,i_{t+1}=q_j|\lambda)$$

$$=\sum_{i=1}^{N}P(o_{t+1}|i_{t+1}=q_j)\cdot P(i_{t+1}=q_j|o_1,\cdots,o_t,i_t=q_i,\lambda)\cdot P(o_1,\cdots,o_t,i_t=q_i|\lambda)$$

$$=\sum_{i=1}^{N}\underbrace{P(o_{t+1}|i_{t+1}=q_j)}_{b_j(o_{t+1})}\cdot\underbrace{P(i_{t+1}=q_j|i_t=q_i)}_{a_{ij}}\cdot\alpha_t(i)$$

因此，我们可以从 α_1 一步步求出 α_T，进而由 $P(O|\lambda)=\sum_{i=1}^{N}\alpha_T(i)$ 求出 $P(O|\lambda)$

Backward Algorithm:



记: $\beta_t(i) = P(o_{t+1},\cdots,o_T \mid i_t=q_i,\lambda)$

$\vdots$

$\beta_1(i) = P(o_2,\cdots,o_T \mid i_1=q_i,\lambda)$

$\pi=(\pi_1,\pi_2,\cdots,\pi_N)$, $\sum_{i=1}^{N}\pi_i=1$

$$
\begin{aligned}
P(O|\lambda) &= P(o_1,\cdots,o_T|\lambda) \\
&= \sum_{i=1}^{N} P(o_1,\cdots,o_T,i_1=q_i|\lambda) \\
&= \sum_{i=1}^{N} P(o_1,\cdots,o_T|i_1=q_i,\lambda)\underbrace{P(i_1=q_i|\lambda)}_{\pi_i} \\
&= \sum_{i=1}^{N} P(o_1|o_2,\cdots,o_T,i_1=q_i,\lambda)\cdot P(o_2,\cdots,o_T|i_1=q_i,\lambda)\cdot\pi_i \\
&= \sum_{i=1}^{N} P(o_1|i_1=q_i,\lambda)\cdot\beta_1(i)\cdot\pi_i \\
&= \sum_{i=1}^{N} b_i(o_1)\cdot\pi_i\cdot\beta_1(i)
\end{aligned}
$$

$$
\begin{aligned}
\beta_t(i) &= P(o_{t+1},\cdots,o_T|i_t=q_i,\lambda) \\
&= \sum_{j=1}^{N} P(o_{t+1},\cdots,o_T,i_{t+1}=q_j|i_t=q_i,\lambda) \\
&= \sum_{j=1}^{N} P(o_{t+1},\cdots,o_T|i_{t+1}=q_j,i_t=q_i,\lambda)\cdot\underbrace{P(i_{t+1}=q_j|i_t=q_i,\lambda)}_{a_{ij}} \\
&= \sum_{j=1}^{N} P(o_{t+1},\cdots,o_T|i_{t+1}=q_j,\lambda)\cdot a_{ij} \\
&= \sum_{j=1}^{N} \underbrace{P(o_{t+1}|o_{t+2},\cdots,o_T,i_{t+1}=q_j,\lambda)}_{P(o_{t+1}|i_{t+1}=q_j,\lambda)}\cdot\underbrace{P(o_{t+2},\cdots,o_T|i_{t+1}=q_j,\lambda)}_{\beta_{t+1}(j)}\cdot a_{ij} \\
&= \sum_{j=1}^{N} b_j(o_{t+1})\cdot\beta_{t+1}(j)\cdot a_{ij}
\end{aligned}
$$

b被观测，a与c独立

因此，我们可以从β_T开始，计算$\beta_T,\cdots\beta_1$，从而求出$P(O|\lambda)$

问题二. Learning

由于 $P(O|\lambda)$ 过于复杂，直接使用 MLE 无法求出，因此使用 EM 求解.

EM: $\theta^{(t+1)} = \arg\max_{\theta} \int \log P(X,Z|\theta) P(Z|X,\theta^{(t)}) dZ$

X: 观测数据 $\downarrow O$　　Z: 隐变量 $\downarrow I$　　θ: 参数 $\downarrow \lambda$　　$\lambda^{(t)} = (\pi^{(t)}, A^{(t)}, B^{(t)})$

则 $\lambda^{(t+1)} = \arg\max_{\lambda} \sum_{I} \log P(O,I|\lambda) P(I|O,\lambda^{(t)})$　　$\lambda^{(t+1)} = (\pi^{(t+1)}, A^{(t+1)}, B^{(t+1)})$

$\hookrightarrow \frac{P(I,O|\lambda^{(t)})}{P(O|\lambda^{(t)})}$ → 常数项.

$$= \arg\max_{\lambda} \sum_{I} \log P(O,I|\lambda) P(I,O|\lambda^{(t)})$$

令 $Q(\lambda,\lambda^{(t)}) = \sum_{I} \log P(O,I|\lambda) P(I,O|\lambda^{(t)})$

$$Q(\lambda,\lambda^{(t)}) = \sum_{I} \left[\left(\log \pi_{i_1} + \sum_{t=2}^{T} \log a_{i_{t-1} i_t} + \sum_{t=1}^{T} \log b_{i_t}(o_t) \right) \cdot P(I,O|\lambda^{(t)}) \right]$$

以计算 $\pi^{(t)}$ 为例：

$$\pi^{(t+1)} = \arg\max_{\pi} Q(\lambda,\lambda^{(t)})$$
$$= \arg\max_{\pi} \sum_{I} [\log \pi_{i_1} \cdot P(I,O|\lambda^{(t)})]$$
$$= \arg\max_{\pi} \sum_{i_1} \cdots \sum_{i_T} [\log \pi_{i_1} \cdot P(O,i_1,\cdots,i_T|\lambda^{(t)})]$$
$$= \arg\max_{\pi} \sum_{i_1} [\log \pi_{i_1} \cdot P(O,i_1|\lambda^{(t)})]$$
$$= \arg\max_{\pi} \sum_{i=1}^{N} [\log \pi_i \cdot P(O,i_1=q_i|\lambda^{(t)})] \quad (s.t. \sum_{i=1}^{N} \pi_i = 1)$$

由拉格朗日乘子法：

$$L(\pi,\gamma) = \sum_{i=1}^{N} [\log \pi_i \cdot P(O,i_1=q_i|\lambda^{(t)}] + \gamma(\sum_{i=1}^{N} \pi_i - 1)$$

令 $\frac{\partial L}{\partial \pi_i} = \frac{1}{\pi_i} \cdot P(O,i_1=q_i|\lambda^{(t)}] + \gamma = 0 \quad \cdots ①$

即 $P(O,i_1=q_i|\lambda^{(t)}) + \pi_i \gamma = 0$

则 $\sum_{i=1}^{N} [P(O,i_1=q_i|\lambda^{(t)}) + \pi_i \gamma] = 0$

$P(O|\lambda^{(t)}) + \gamma = 0$

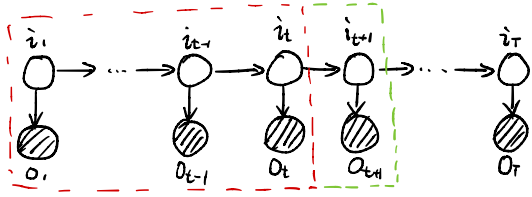
$\therefore \gamma = -P(O|\lambda^{(t)})$

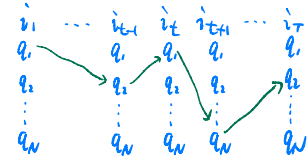
将 γ 代入式①，求得 $\pi_i^{(t+1)} = \frac{P(O,i_1=q_i|\lambda^{(t)})}{P(O|\lambda^{(t)})}$

$\therefore \pi^{(t+1)} = (\pi_1^{(t+1)}, \pi_2^{(t+1)}, \cdots, \pi_N^{(t+1)})$

问题三：Decoding

Viterbi：

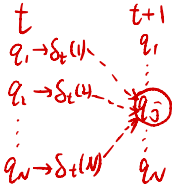




设 $\delta_t(i) = \max_{i_1,\cdots,i_{t-1}} P(o_1, o_2, \cdots o_t, i_1, \cdots i_{t-1}\ i_t = q_i)$ 表示 t 时刻时到达状态 q_i 的最大概率，

则 $\delta_{t+1}(j) = \max_{i_1,\cdots i_t} P(o_1, o_2, \cdots o_{t+1}, i_1, \cdots i_t, i_{t+1} = q_j)$

$= \max_{1 \le i \le N} \delta_t(i) \cdot a_{ij} \cdot b_j(o_{t+1})$



因此，我们可以计算 δ_1，δ_2 直到 δ_T，δ_T 就是我们要求得的那个概率值。

考虑记录路径，可以令 $\psi_{t+1}(j) = \arg\max_{1 \le i \le N} \delta_t(i) \cdot a_{ij}$，即记录使得 t+1 时刻到达 q_j 的 t 时刻所在的状态 i，$\psi_{t+1}(j)$ 返回的就是使这个概率最大的状态，所以每走一步就可以记录一个状态。

四. 隐式马尔可夫模型小结.

HMM (mixture + time) 混合+序列.

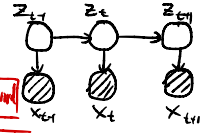
在 HMM 中，只对隐变量有要求，要求其为离散变量，不对观测变量有要求.

HMM 是一种动态模型 (Dynamic Model)，也叫状态空间模型 (state space Model)

对于这种模型，寻其它概率图模型或贝叶斯法一样，都是要解决两个问题：

即 { Learning: Baum Welch (EM) 对于时间序列的状态空间模型，Inference 可细分为：
Inference.

Inference {
- decoding: $P(z_1,\cdots z_t|x_1,\cdots x_t)$ Viterbi
- 求 evidence 的概率: $P(X|\theta)=P(x_1,\cdots,x_t|\theta)$ Forward/Backward
- filtering: $P(z_t|x_1,x_2,\cdots,x_t)\to$ online Forward
- smoothing: $P(z_t|x_1,x_2,\cdots,x_T)\to$ offline Forward-Backward Algorithm
- prediction: { $P(z_{t+1},z_{t+2}|x_1,\cdots,x_t)$; $P(x_{t+1},x_{t+2}|x_1,\cdots,x_t)$ }



Filtering: $P(z_t|x_1,\cdots,x_t)=\frac{P(z_t,x_1,\cdots,x_t)}{P(x_1,\cdots,x_t)}=\frac{P(x_1,\cdots,x_t,z_t)}{\sum_{z_t}P(x_1,\cdots x_t,z_t)}\propto P(x_1,\cdots,x_t,z_t)=\alpha_t$

即 Filtering 问题用的方法是 Forward 算法.

Smoothing: $P(z_t|x_1,\cdots,x_T)=\frac{P(x_1,\cdots,x_T,z_t)}{P(x_1,\cdots,x_T)}=\frac{P(x_1,\cdots,x_T,z_t)}{\sum_{z_t}P(x_1,\cdots x_T,z_t)}$

$$P(x_1,\cdots x_T,z_t)=P(x_1,\cdots,x_t,x_{t+1},\cdots,x_T,z_t)$$
$$=\underbrace{P(x_{t+1},\cdots,x_T|x_1,\cdots x_t,z_t)}\cdot\underbrace{P(x_1,\cdots,x_t,z_t)}_{\alpha_t}$$

使用 D-划分，设 $x_1,\cdots,x_t$ 为 A，z_t 为 B，$x_{t+1},\cdots,x_T$ 为 C，则由 A 到 C 的路径为：

A → (z_t) → z_{t+1} → C，因此，在给定 B 的情况下，$A\perp C|B$.

B.

$\therefore P(x_{t+1},\cdots,x_T|x_1,\cdots x_t,z_t)=P(x_{t+1},\cdots,x_T|z_t)=\beta_t$

因此 $P(x_1,\cdots,x_T,z_t)=\beta_t\cdot\alpha_t$. $P(z_t|x_1,\cdots,x_T)\propto P(x_1,\cdots,x_T,z_t)=\alpha_t\cdot\beta_t$

(注意，有时文献中称 Baum Welch 为 Forward-Backward 算法，这是不准确的，原因是 Baum Welch 算法在实现过程中使用了 Forward-Backward 算法)

Prediction: $P(z_{t+1} | x_1, \cdots, x_t) = \sum_{z_t} P(z_{t+1}, z_t | x_1, \cdots, x_t) = \sum_{z_t} \underbrace{P(z_{t+1} | z_t, x_1, \cdots x_t)}_{P(z_{t+1} | z_t)} \cdot \underbrace{P(z_t | x_1, \cdots x_t)}_{\text{filtering}}$

$P(x_{t+1} | x_1, \cdots, x_t) = \sum_{z_{t+1}} P(x_{t+1}, z_{t+1} | x_1, \cdots, x_t) = \sum_{z_{t+1}} \underbrace{P(x_{t+1} | z_{t+1}, x_1, \cdots x_t)}_{P(x_{t+1} | z_{t+1})} \cdot P(z_{t+1} | x_1, \cdots x_t)$