Statistical and Algorithmic Foundation of Hidden Markov Models

Zicheng Yu

November 26, 2025

1 Setup & Definitions

1.1 Data Structure

We start with a feature matrix M with dimensions $T \times F$.

- T: Time steps (rows, e.g., $t_0, t_1, \ldots, t_{T-1}$)
- F: Features (columns)

Step 1: Standardization
Turn matrix M into a standardized matrix X by computing z-scores per column (mean 0, variance 1).

$$X \in \mathbb{R}^{T \times F}$$

1.2 Goal

The objective is to learn K hidden regimes (states) $S_t \in \{1, \ldots, K\}$.

1.3 HMM Assumptions

A Gaussian HMM makes two primary assumptions:

1. **Emissions (Observations):** For a given state k, the observation X_t is drawn from a Multivariate Normal distribution:
$$X_t \mid S_t = k \sim \mathcal{N}(\mu_k, \Sigma_k)$$
2. **Transitions:** The probability of moving from state i to state j is defined by the transition matrix A:
$$A_{ij} = P(S_{t+1} = j \mid S_t = i)$$

1.4 Parameters to Learn (θ)

We need to learn the set $\theta = \{\pi, A, \mu, \Sigma\}$.

- $\pi \in \mathbb{R}^K$: Initial state probabilities $P(S_1 = k)$.
- $A \in \mathbb{R}^{K \times K}$: Transition probability matrix.
- $\mu \in \mathbb{R}^{K \times F}$: Means for each state.
- $\Sigma \in \mathbb{R}^{K \times F \times F}$: Covariance matrices for each state.

2 Initialization

Before running the EM algorithm, we need reasonable initial guesses for our parameters. Choose K (e.g., $K = 4$).

- **Means (μ):** Run K-Means clustering on X. Use the resulting centroids as the initial μ_k.
- **Covariance (Σ):** Take the empirical covariance of the points assigned to each cluster. *Note: Add a tiny ridge ϵI to the diagonal for numerical stability.*
- **Priors (π):** The fraction of days spent in each cluster during the initial time period.
- **Transitions (A):** In macro data, states are persistent (diagonal probabilities are high).

Persistence Logic: Under a Markov Model, the run length L of state k is geometrically distributed:

$$P(L = l) = A_{kk}^{l-1}(1 - A_{kk})$$

The expected run length is $E[L] = \frac{1}{1-A_{kk}}$. If we assume an expected duration d_k (e.g., 30 days), we can initialize the diagonal:

$$A_{kk} = 1 - \frac{1}{d_k}$$

The off-diagonal probabilities split the remaining mass equally:

$$A_{k \to j} = \frac{1 - A_{kk}}{K - 1}$$

3 Learning: The EM Algorithm (Python Package: GaussianHMM)

We use the Expectation-Maximization (EM) algorithm to fit the model.

3.1 E-Step (Expectation)

1. Logarithmic likelihoods:
For each t and state k, compute the emission log-likelihood:

$$\log B_{t,k} = \log \mathcal{N}(x_t \mid \mu_k, \Sigma_k)$$

2. Forward Pass (α):
Initialize: $\log \alpha_{1,k} = \log \pi_k + \log B_{1,k}$.
Recursion:

$$\log \alpha_{t,k} = \log B_{t,k} + \log \sum_i \exp(\log \alpha_{t-1,i} + \log A_{i,k})$$

3. Backward Pass (β):
Initialize: $\beta_{T,k} = 1$ (or 0 in log-space, as "nothing to observe" after T).
Recursion:

$$\log \beta_{t,i} = \log \sum_j \exp(\log A_{ij} + \log B_{t+1,j} + \log \beta_{t+1,j})$$

4. Posterior over States (γ):

$$\gamma_{t,k} = P(S_t = k \mid X_{1:T}) \propto \exp(\log \alpha_{t,k} + \log \beta_{t,k})$$

5. Pairwise Posteriors (ξ):
Expected transitions at each step:

$$\xi_{t,i \to j} \propto \exp(\log \alpha_{t,i} + \log A_{ij} + \log B_{t+1,j} + \log \beta_{t+1,j})$$

3.2 M-Step (Maximization)

Use γ and ξ as weights to recompute parameters. Iterate until convergence.

- **New μ_k:**
$$\mu_k \leftarrow \frac{\sum_t \gamma_{t,k} x_t}{\sum_t \gamma_{t,k}}$$
- **New Σ_k:**
$$\Sigma_k \leftarrow \frac{\sum_t \gamma_{t,k}(x_t - \mu_k)(x_t - \mu_k)^T}{\sum_t \gamma_{t,k}}$$
- **New π_k:**
$$\pi_k \leftarrow \gamma_{1,k}$$
- **New A_{ij}:**
$$A_{ij} \leftarrow \frac{\sum_{t=1}^{T-1} \xi_{t,i\to j}}{\sum_{t=1}^{T-1} \gamma_{t,i}}$$

4 Forward-Only Filtering (Forecasting)

Once the model is fitted $(\hat{\pi}, \hat{A}, \hat{\mu}, \hat{\Sigma})$, we can perform filtering on new data.
Recursion:

1. **Init:** $\log \alpha_{0,k} = \log \pi_k + \log b_k(x_0)$
2. **Step:** $\log \alpha_{t,k} = \log b_k(x_t) + \text{LogSumExp}_i(\log \alpha_{t-1,i} + \log A_{i,k})$
3. **Normalize:** Calculate current regime probability:
$$y_t = \text{Softmax}(\log \alpha_t)$$
4. **Forecast:** Predict the probability of the next state:
$$P(S_{t+1} \mid X_{1:t}) = \gamma_t A$$

5 Implementation Details & Tricks

5.1 Multivariate Normal Log-PDF

$$\log \mathcal{N}(x; \mu, \Sigma) = -\frac{1}{2}\left[F \log(2\pi) + \log|\Sigma| + (x-\mu)^T \Sigma^{-1}(x-\mu)\right]$$

5.2 The LogSumExp Trick

To prevent numerical underflow/overflow when summing probabilities in log-space (computing $\log \sum_i e^{v_i}$):
Let $m = \max(v_i)$.
$$\log \sum_i e^{v_i} = \log\left(e^m \sum_i e^{v_i - m}\right) = m + \log \sum_i e^{v_i - m}$$