Hidden Markov Models: lecture 2

Likelihood computation

Xavier Didelot

HMM definition

- A Hidden Markov Model (HMM) is a Markov chain in which the sequence of states $C_1, ..., C_T$ is not observed but hidden
- Instead of observing the sequence of states, we observe the emissions $X_1, ..., X_T$
- A HMM is defined by two quantities:
  - The transition matrix $\mathbf{\Gamma}$ of elements γ_{ij} where i and j are states:

    $$\gamma_{ij} = p(C_t = j | C_{t-1} = i)$$

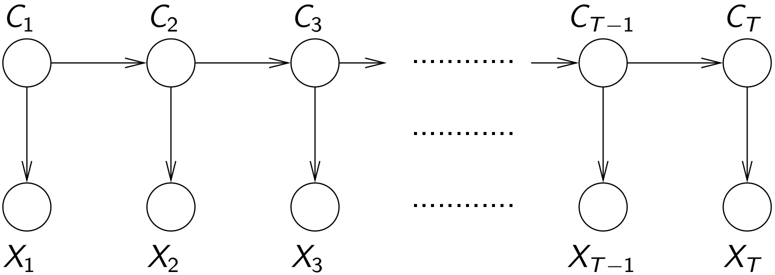
  - The emission probabilities $p_i(x)$ where i is a state and x is an emission:

    $$p_i(x) = p(X_t = x | C_t = i)$$

- The unconditional distribution at t is denoted $\boldsymbol{u}(t)$ and the initial distribution is $\boldsymbol{u}(1)$

$$\boldsymbol{u}(t) = (p(C_t = 1), p(C_t = 2), ..., p(C_t = m))$$

Dependency graph of a hidden Markov model

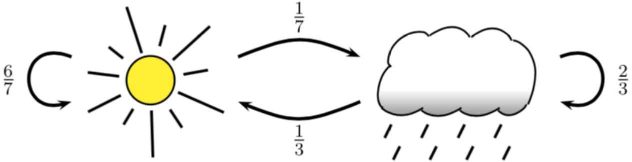


$$p(\boldsymbol{X}^{(T)}, \boldsymbol{C}^{(T)}) = p(C_1) \prod_{k=2}^{T} p(C_k | C_{k-1}) \prod_{k=1}^{T} p(X_k | C_k)$$

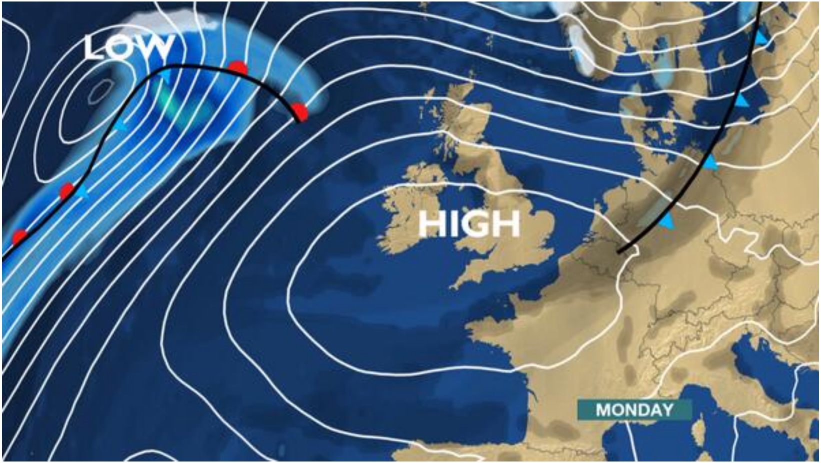
$$p(\boldsymbol{x}^{(T)}, \boldsymbol{c}^{(T)}) = u_{c_1}(1) \prod_{k=2}^{T} \gamma_{c_{k-1} c_k} \prod_{k=1}^{T} p_{c_k}(x_k)$$

From Markov chain to HMM

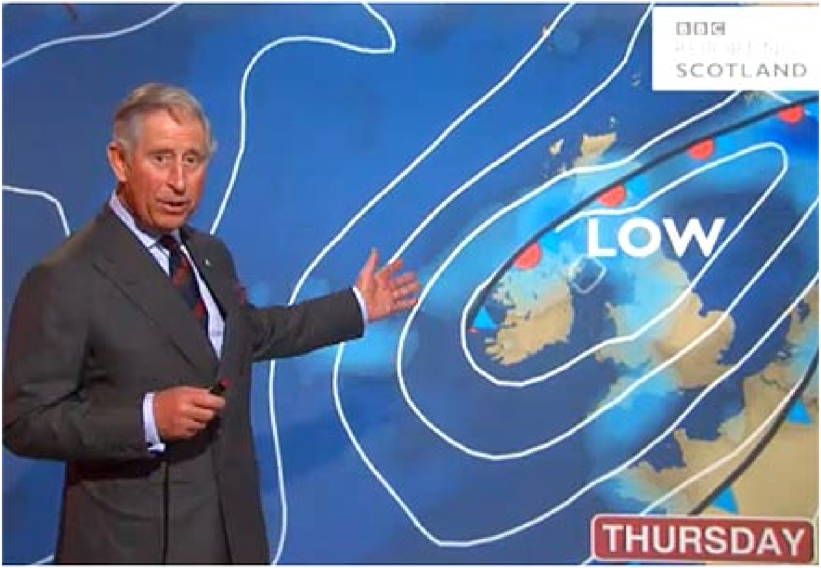
- In the previous lecture we described how a HMM is an extension of a mixture model
- Another way to think about HMM is as an extension of Markov chain model
- For example, let's say we want to build a weather model to forecast the probability of rain on a given day
- We could use a simple Markov chain model:



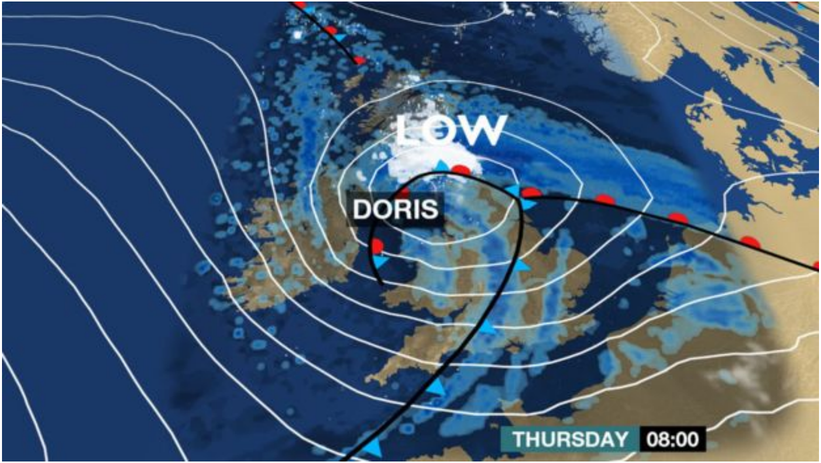
Weather example



Weather example



Weather example

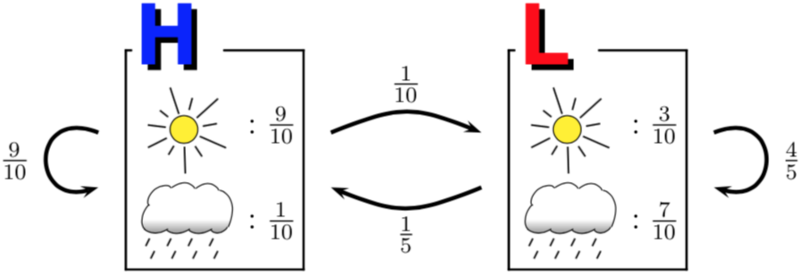


Weather example

- Atmospheric pressure is a strong indicator of rain vs dry weather
- In 1643, Torricelli invented the barometer

Weather example

- We can build a HMM where pressure is the hidden state:



- Since the emissions are distributed according to Bernouilli distributions, this model is called a Bernouilli-HMM

Univariate marginal distribution

- What is $p(X_t = x)$?
- Decompose over states at t:

$$p(X_t = x) = \sum_{i=1}^{m} p(C_t = i) p(X_t = x | C_t = i)$$

- It is convenient to rewrite in matrix notation:

$$p(X_t = x) = \boldsymbol{u}(t)\boldsymbol{P}(x)\mathbf{1}'$$

  where $\boldsymbol{P}(x)$ is a diagonal matrix with i^{th} diagonal element equal to $p_i(x)$
- Since $\boldsymbol{u}(t) = \boldsymbol{u}(t-1)\boldsymbol{\Gamma}$ we have $\boldsymbol{u}(t) = \boldsymbol{u}(1)\boldsymbol{\Gamma}^{t-1}$ and therefore:

$$p(X_t = x) = \boldsymbol{u}(1)\boldsymbol{\Gamma}^{t-1}\boldsymbol{P}(x)\mathbf{1}'$$

- If the chain has initial distribution equal to the stationary distribution $\boldsymbol{\delta}$, then:

$$p(X_t = x) = \boldsymbol{\delta}\boldsymbol{P}(x)\mathbf{1}'$$

Bivariate marginal distribution

- What is $p(X_t = v, X_{t+k} = w)$?
- Decompose over states at both t and $t + k$:

$$p(X_t = v, X_{t+k} = w) = \sum_{i=1}^{m} \sum_{j=1}^{m} p(X_t = v, X_{t+k} = w, C_t = i, C_{t+k} = j)$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{m} p(C_t = i) p_i(v) \gamma_{ij}(k) p_j(w)$$

where $\gamma_{ij}(k)$ denotes the (i, j) element of $\mathbf{\Gamma}^k$

- In matrix format:

$$p(X_t = v, X_{t+k} = w) = \boldsymbol{u}(t) \boldsymbol{P}(v) \mathbf{\Gamma}^k \boldsymbol{P}(w) \mathbf{1}'$$

- If the Markov chain is stationary:

$$p(X_t = v, X_{t+k} = w) = \boldsymbol{\delta} \boldsymbol{P}(v) \mathbf{\Gamma}^k \boldsymbol{P}(w) \mathbf{1}'$$

Likelihood

The likelihood of a HMM is given by:

$$L_T = p(\boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)}) = \boldsymbol{u}(1)\boldsymbol{P}(x_1)\boldsymbol{\Gamma}\boldsymbol{P}(x_2)\boldsymbol{\Gamma}\boldsymbol{P}(x_3)...\boldsymbol{\Gamma}\boldsymbol{P}(x_T)\mathbf{1}'$$

Likelihood proof 1, using linear algebra

$$L_T = \sum_{c_1, c_2, \ldots, c_T = 1}^{m} p(\boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)}, \boldsymbol{C}^{(T)} = \boldsymbol{c}^{(T)})$$

Since:

$$p(\boldsymbol{X}^{(T)}, \boldsymbol{C}^{(T)}) = p(C_1) \prod_{k=2}^{T} p(C_k | C_{k-1}) \prod_{k=1}^{T} p(X_k | C_k)$$

It follows that:

$$L_T = \sum_{c_1, c_2, \ldots, c_T = 1}^{m} p(C_1 = c_1) p_{c_1}(x_1) \gamma_{c_1 c_2} p_{c_2}(x_2) \ldots \gamma_{c_{T-1} c_T} p_{c_T}(x_T)$$

which can be rewritten in matrix format to give the likelihood equation.

Likelihood proof 2, using induction

Define the vector $\boldsymbol{\alpha}_t$ such that

$$\alpha_t(j) = p(\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)}, C_t = j)$$

In particular:

$$\alpha_1(j) = p(X_1 = x_1, C_1 = j) = u_1(j) p_j(x_1)$$

In matrix format: $\boldsymbol{\alpha}_1 = \boldsymbol{u}(1)\boldsymbol{P}(x_1)$

We have the recursion:

$$\alpha_t(j) = \sum_{k=1}^{m} p(\boldsymbol{X}^{(t)} = \boldsymbol{x}^{(t)}, C_t = j, C_{t-1} = k) = \sum_{k=1}^{m} \alpha_{t-1}(k)\gamma_{kj} p_j(x_t)$$

In matrix format: $\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1}\boldsymbol{\Gamma}\boldsymbol{P}(x_t)$

Finally, note that $L_T = \sum_{k=1}^{m} \alpha_T(k) = \boldsymbol{\alpha}_T \mathbf{1}'$ and the likelihood equation follows.

The forward algorithm

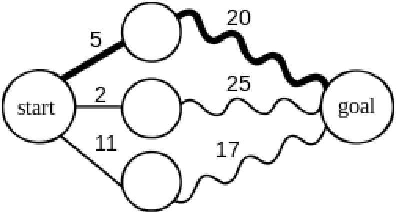
- An important consequence of the likelihood equation is that the likelihood can be calculated using the **forward algorithm**:
  - Set $\boldsymbol{\alpha}_1 = \boldsymbol{u}(1)\boldsymbol{P}(x_1)$
  - For t from 2 to T, calculate $\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1}\boldsymbol{\Gamma}\boldsymbol{P}(x_t)$
  - Return the likelihood $L_T = \boldsymbol{\alpha}_T \mathbf{1}'$
- This algorithm calculates the likelihood using a number of operations of order Tm^2
- This is much more efficient than the brute force approach of calculating the likelihood by summing over all m^T possible values for $\boldsymbol{C}^{(T)}$

Dynamic programming

- The forward algorithm, and other algorithms we will see in subsequent lectures, is an example of **dynamic programming**
- In dynamic programming, a costly computation is replaced with a simpler one by exploiting a recursive form
- If we calculated all m^T combinations of $\boldsymbol{C}^{(T)}$, many subcalculations would be done over and over again
- By rearranging terms of the summation, or in other words reusing rather than recalculating certain terms, the algorithm becomes much more efficient

Dynamic programming

- Dynamic programming was developed by Richard Bellman in the 1950s
- Simplifying a complicated problem by breaking it down into simpler sub-problems in a recursive manner
- Divide and conquer
- Don't recalculate the same thing multiple times - memoisation



The forward algorithm in R

- Example of R code for the forward algorithm in the case of an HMM with emissions from a Poisson distribution
- The vector `x` is observed
- The matrix `Gamma` contains the transition probabilities
- Each state emits from a Poisson distribution, with parameters stored in the vector `lambda`
- The initial probabilities are stored in the vector `u1`
- Code:

```
alpha <- u1*dpois(x[1],lambda)
for (i in 2:T)
  alpha <- alpha %*% Gamma*dpois(x[i],lambda)
sum(alpha)
```

Example

- Consider a hidden Markov model with transition matrix:

$$\boldsymbol{\Gamma} = \begin{pmatrix} 0.5 & 0.5 \\ 0.25 & 0.75 \end{pmatrix}$$

- The emissions are binary, with:

$$p(X_t = 0 | C_t = 1) = 0.5 \text{ and } p(X_t = 1 | C_t = 1) = 0.5$$

$$p(X_t = 0 | C_t = 2) = 0 \text{ and } p(X_t = 1 | C_t = 2) = 1$$

- We observe the data $X_1 = 1, X_2 = 1, X_3 = 1$
- Calculate the likelihood $L_T = 29/48$ using both brute force and the forward algorithm

Likelihood with missing data

- It is often difficult to deal with missing data in time series analysis
- Calculating the likelihood of a HMM with missing data is straightforward though
- If x_t is missing, the term $\boldsymbol{P}(x_t)$ is removed from the matrix multiplicative form of the likelihood expression
- The forward algorithm can also be adjusted by removing the term $\boldsymbol{P}(x_t)$
- In the special case where all data is missing except one or two points, we find again the formula at the start of this lecture for univariate and bivariate marginal distribution
- Interval-censored data, where for example instead of observing x_t we observe that $a \leq x_t \leq b$ can be dealt with similarly by replacing the i-th diagonal element of the matrix $\boldsymbol{P}(x_t)$ with $p(a \leq x_t \leq b | C_t = i)$

Filtering

- The distribution $p(C_T|\boldsymbol{X}^{(T)})$ is often of interest
- This is the distribution of the last hidden state, at the end of the observed sequence
- The forward algorithm is a solution to this problem called **filtering**
- By definition $\alpha_T(j) = p(\boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)}, C_T = j)$ and therefore:

$$p(C_T = j|\boldsymbol{X}^{(T)}) = \frac{\alpha_T(j)}{\sum_{i=1}^{m} \alpha_T(i)} = \frac{\alpha_T(j)}{L_T}$$

- More generally, we might want to know the distribution $p(C_t|\boldsymbol{X}^{(T)})$ of hidden state at some point in the observed sequence
- This problem is called **smoothing** but can't be solved just using the forward algorithm...

Conclusions

- The likelihood of a HMM can be calculated efficiently using the forward algorithm
- The computational complexity of the forward algorithm is $O(Tm^2)$
- The forward algorithm is an example of dynamic programming
- Missing or censored data is not an issue
- The forward algorithm also allows us to solve the filtering problem, ie to find the distribution of the HMM state at the end of the observed sequence
- But the forward algorithm does not solve the smoothing problem, ie to find the distribution of the HMM state in the middle of the observed sequence...