Hidden Markov Models: lecture 4

Global decoding

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:

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

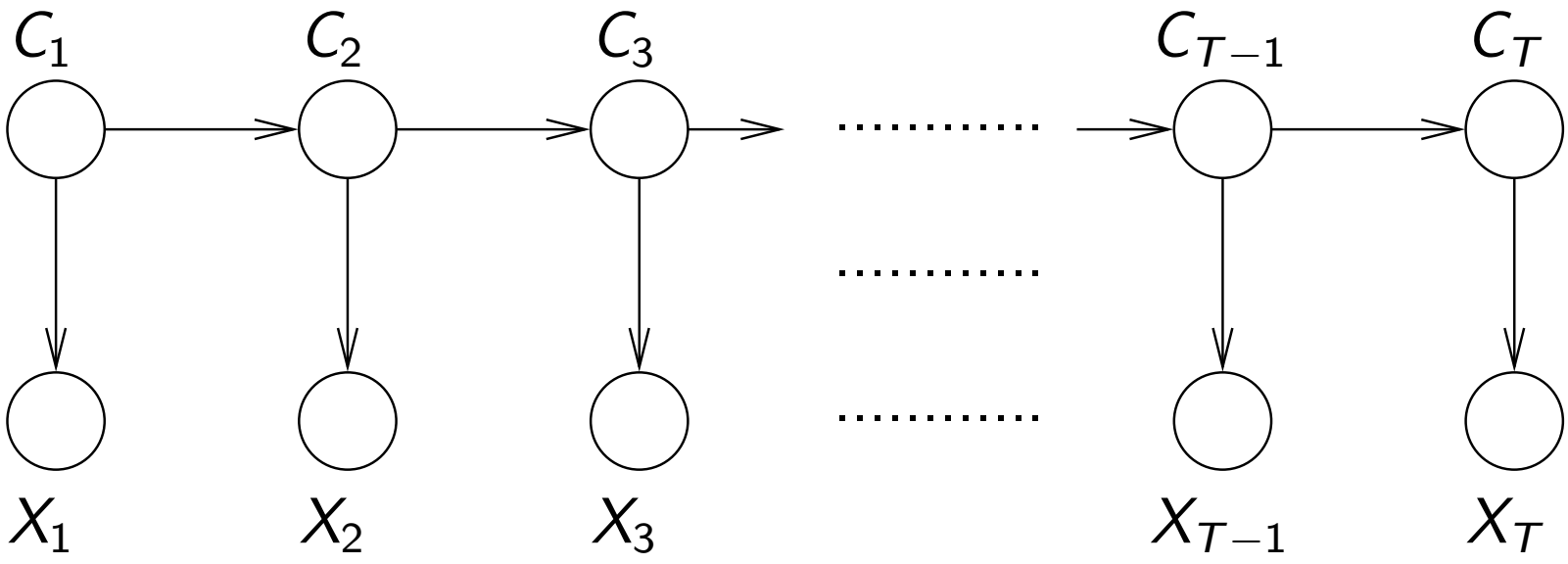
  - The emission probabilities $p_i(x)$ where i is a state and x is an emission:

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

- 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



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

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

Local vs global decoding

- In the last two lectures, we discussed the forward-backward algorithm, which allows us to calculate $p(C_t|\boldsymbol{X}^{(T)})$ and therefore perform local decoding
- Global decoding means to jointly estimate the state for all t
- An obvious first approach is to take the best state for all sites marginally, this is called the maximum accuracy method
- But this path is not always best, and can even have a probability of zero
- A more principled solution is to try and find the path $\boldsymbol{c}^{(T)}$ with maximum probability, ie to maximise:

$$p(\boldsymbol{C}^{(T)} = \boldsymbol{c}^{(T)}|\boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)}) \propto p(\boldsymbol{C}^{(T)} = \boldsymbol{c}^{(T)}, \boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)})$$

Global decoding

- We want to find the most probable state path π^*, ie the one with maximum probability:

$$\pi^* = \text{argmax}_{\boldsymbol{c}^{(T)}}(p(\boldsymbol{X}^{(T)} = \boldsymbol{x}^{(T)}, \boldsymbol{C}^{(T)} = \boldsymbol{c}^{(T)}))$$

- Could we consider all states paths? There are m^T possible paths, so even for the simplest of HMMs with $m = 2$ states this will not be doable for a short sequence of $T = 100$ observations. . .
- Luckily, this can be solved in a similar way as the likelihood calculation: using dynamic programming, ie using a recursive solution
- Let $v_k(i)$ denote the probability of the most probable path ending in state k for the sequence up to the i^{th} observation x_i
- Suppose $v_k(i)$ is known for all k. Then the probabilities can be calculated up to observation x_{i+1} using:

$$v_l(i+1) = p_l(x_{i+1})\text{max}_k(v_k(i)\gamma_{kl})$$

The Viterbi algorithm (1/2)

- The probability that the chain starts in state k and emits x_1 is $v_k(1) = u_k(1)p_k(x_1)$
- Wecalculate recursively the $v_k(i)$ for i from 2 to T using the equation on the previous slide
- At the end of this recursion, we obtain the $v_k(T)$ for all k and we have that $\pi^*_T = \max_k(v_k(T))$ is the probability of the optimal path, and we know that this optimal path finishes in state $\operatorname{argmax}_k(v_k(T))$
- We then trace our steps back from T down to 1 to reconstruct the chain of states that gave this optimal probability (to do this, we have to record pointers in the forward step)

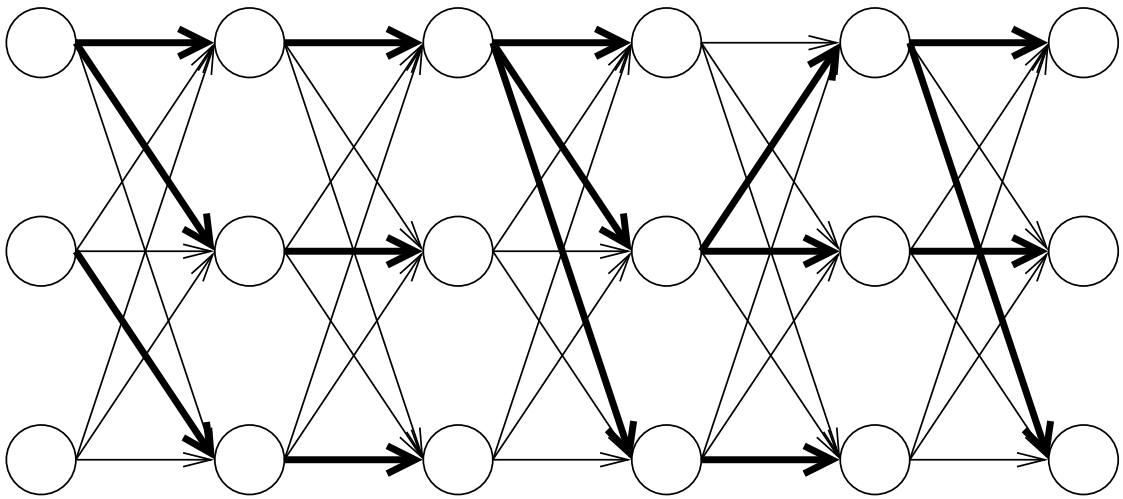
The Viterbi algorithm (2/2)

We deduce the famous Viterbi algorithm:

- Initialisation ($i = 1$):
  - $v_k(1) = u_k(1)p_k(x_1)$ for all $k = 1..m$
- Recursion ($i = 2..T$):
  - $v_l(i) = p_l(x_i)\max_k(v_k(i-1)\gamma_{kl})$
  - $\mathrm{ptr}_i(l) = \mathrm{argmax}_k(v_k(i-1)\gamma_{kl})$
- Termination:
  - $p(x, \pi^*) = \max_k(v_k(T))$
  - $\pi^*_T = \mathrm{argmax}_k(v_k(T))$
- Traceback ($i = T..1$):
  - $\pi^*_{i-1} = \mathrm{ptr}_i(\pi^*_i)$

Pointers

The pointer $\mathrm{ptr}_i(l)$ stores from which state in step $i-1$ did we reach state l in step i. Recursively, this identifies a unique optimal path from the start (step 1) to state l in step i.



History of the Viterbi algorithm

- This algorithm is named after Andrew Viterbi, an Italian-born American electrical engineer
- He discovered the algorithm in 1967
- However, others discovered the algorithm independently soon before or after
- Discovered for military applications and kept secret?
- The Viterbi algorithm is an example of dynamic programming, where a complicated problem is simplified by breaking it down into simpler sub-problems in a recursive manner
- Dynamic programming was developed by Richard Bellman in the 1950s

Example: the occasionally dishonest casino

- State H is honest, state D is dishonest
- $p_H(1) = p_H(2) = p_H(3) = p_H(4) = p_H(5) = p_H(6) = 1/6$
- $p_D(1) = p_D(2) = p_D(3) = p_D(4) = p_D(5) = 1/9$ and $p_D(6) = 4/9$
- $\gamma_{HD} = \gamma_{DH} = 0.1$ and $\gamma_{HH} = \gamma_{DD} = 0.9$
- Observed values:

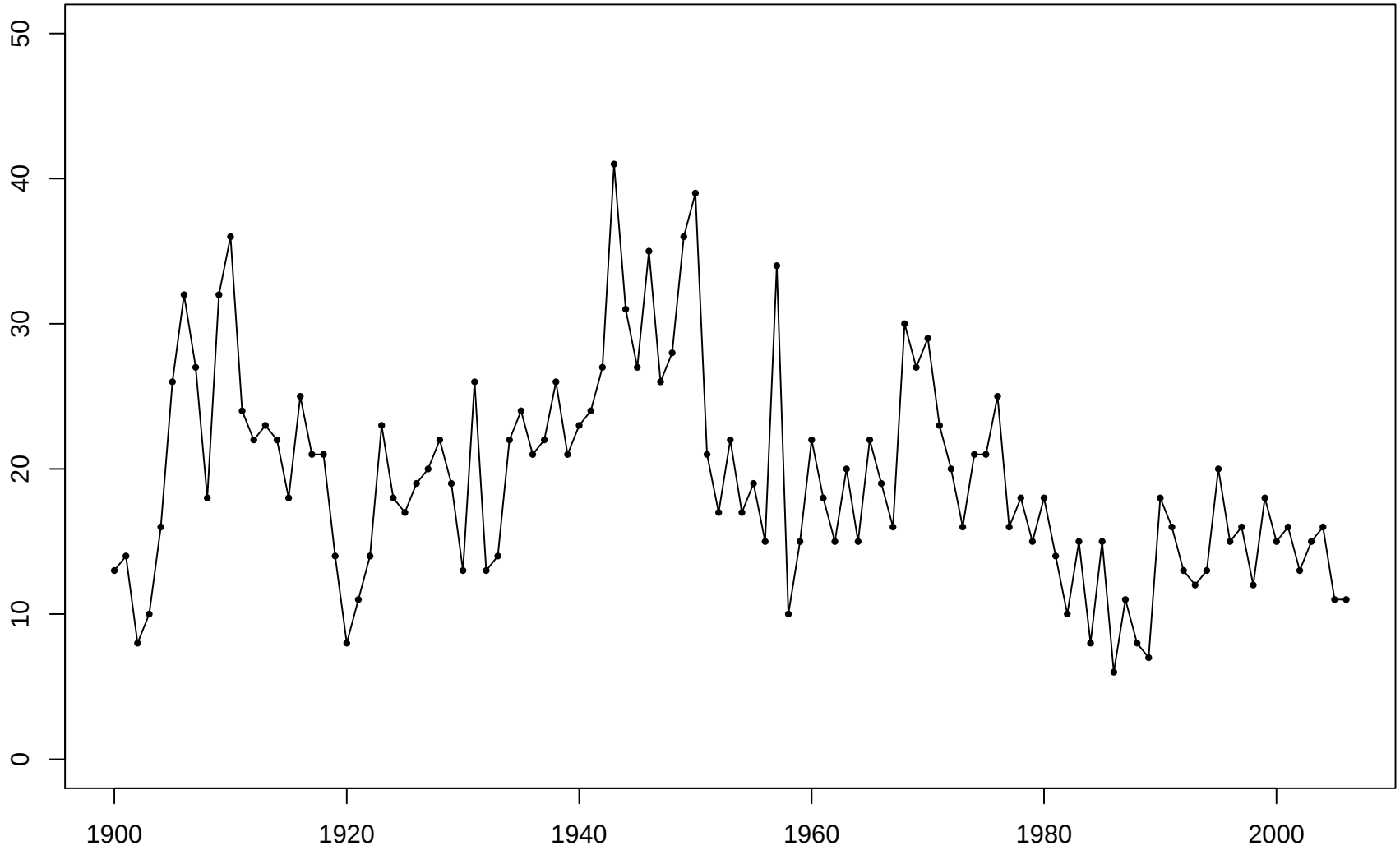
  5 3 1 5 3 4 6 5 6 3 2 6 6 2 1 2 4 3 2 5 3 2 2 6
- True unobserved sequence of states:

  H H H H H H D D D D D D D H H H H H H H H H H H

Example: the occasionally dishonest casino

- Observed values:
  5 3 1 5 3 4 6 5 6 3 2 6 6 2 1 2 4 3 2 5 3 2 2 6
- True unobserved sequence of states:
  H H H H H H D D D D D D D H H H H H H H H H H H
- Local decoding result:
  H H H H H H D D D D D D D H H H H H H H H H H H
- Global decoding result:
  H H H H H H H H H H H H H H H H H H H H H H H H
- In this case, the local decoding approach finds the correct path whereas the global decoding does not
- The global decoding path π^* still has highest probability though
- But we have $p(\pi^*, x^{(T)})/p(x^{(T)}) = 0.07$ so even the most likely path is not that likely and it is not surprising that it is not the correct one

Earthquake data



Earthquake model

- Transition matrix:

$$\mathbf{\Gamma} = \begin{pmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{pmatrix}$$

- We use as starting distribution the stationary distribution:

$$\boldsymbol{u}(1) = \boldsymbol{\delta} = (0.5, 0.5)$$

- Emission probabilities:

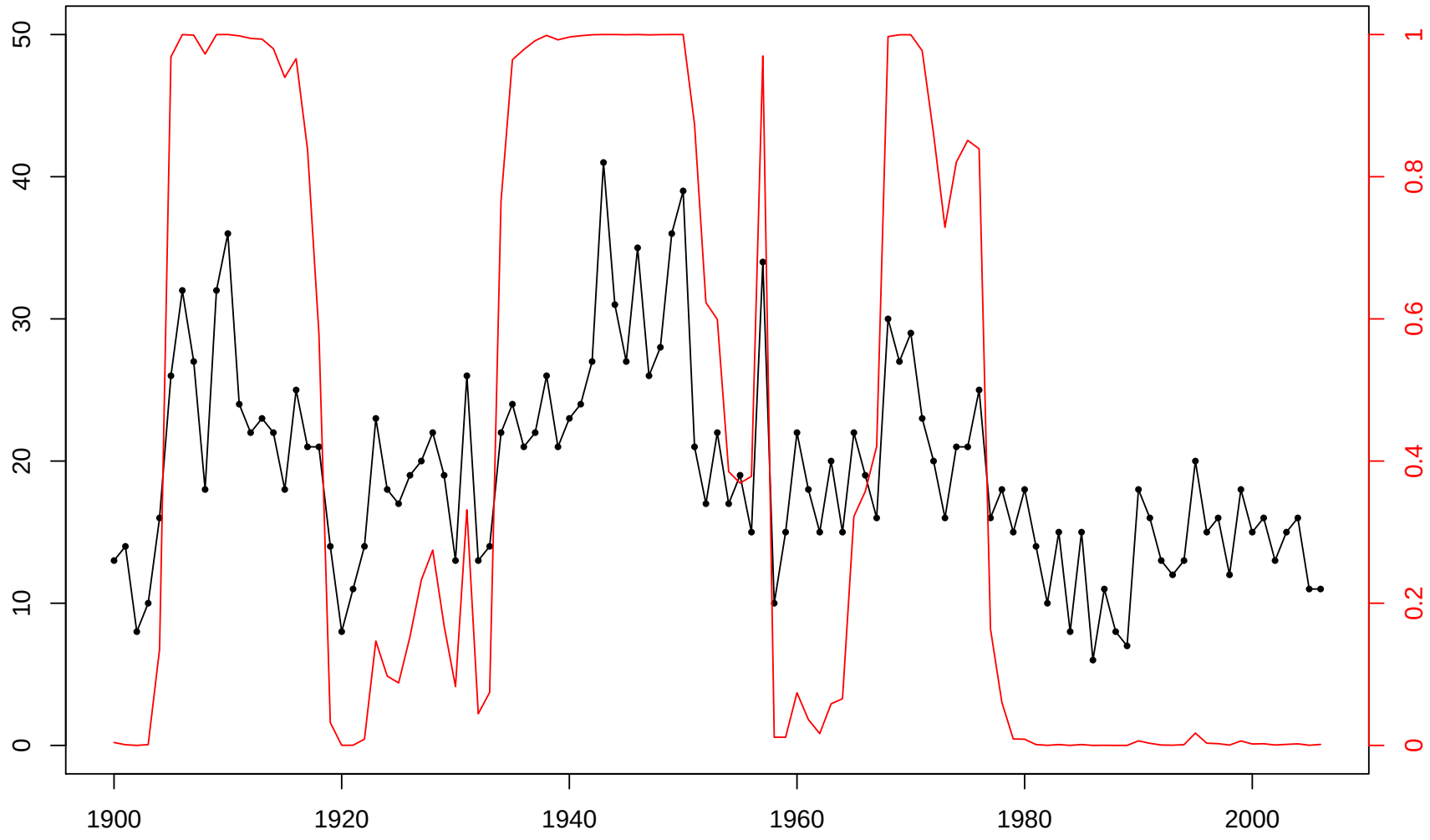
$$p_1(x) = e^{-15}15^x/x! \text{ and } p_2(x) = e^{-25}25^x/x!$$

Earthquake example

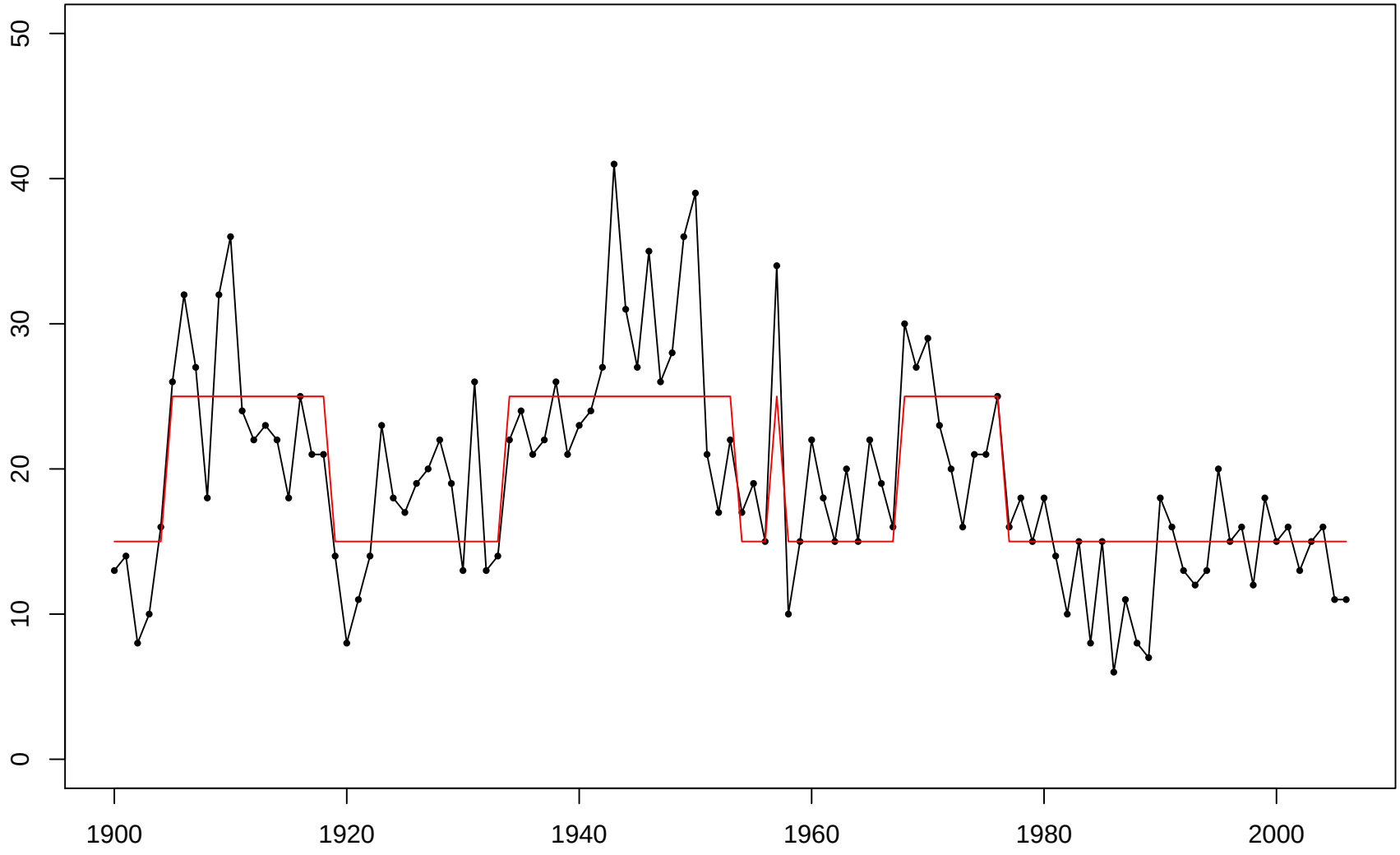
```
library(depmixS4)
m=depmix(quakes~1,nstates=2,
             family=poisson(),ntimes=length(quakes))
m=setpars(m,c(0.5,0.5,0.9,0.1,0.1,0.9,log(15),log(25)))
a=forwardbackward(m)$gamma #Local decoding
b=viterbi(m)$state         #Global decoding
summary(m)
```

```
## Initial state probabilties model
## pr1 pr2
## 0.5 0.5
##
## Transition matrix
##        toS1 toS2
## fromS1  0.9  0.1
## fromS2  0.1  0.9
##
## Response parameters
## Resp 1 : poisson
##     Re1.(Intercept)
## St1           2.708
## St2           3.219
```

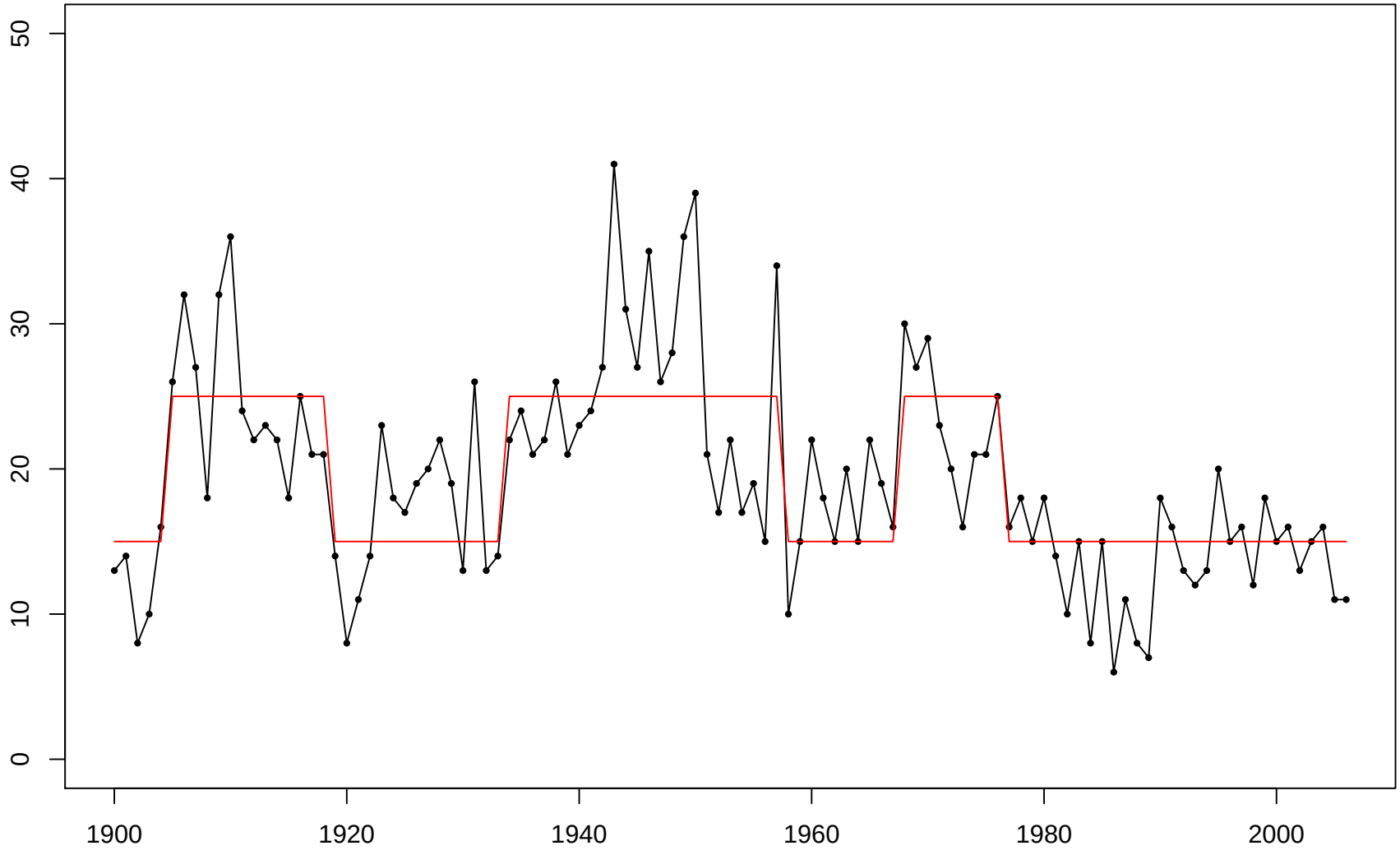
Earthquake example: local decoding



Earthquake example:local decoding



Earthquake example: global decoding



Conclusions

- Local decoding and global decoding are different
- Both problems can be solved very effectively in Tm^2 operations using dynamic programming: forward-backward algorithm for local decoding and Viterbi algorithm for global decoding
- Depending on the application, one or the other is preferred
- In practice, local and global decoding often have similar overall results
- In our next lecture we will turn to the problem of estimating the parameters of the model, ie the values of the transition and emission probabilities