# Introduction to Time Series Modelling

David Barber

# Table of Contents

# Table of Contents

# Probability

## Why Probability?

- Probability is a logical calculus of uncertainty.
- Natural framework to use in models of physical systems, such as the Ising Model (1920) and in AI applications, such as the HMM (Baum 1966, Stratonovich 1960).

---

## The need for structure

- We often want to make a probabilistic description of many objects (electron spins, neurons, customers, *etc.* ).
- Typically the representational and computational cost of probabilistic models grows exponentially with the number of objects represented.
- Without introducing strong structural limitations about how these objects can interact, probability is a non-starter.
- For this reason, computationally 'simpler' alternatives (such as fuzzy logic) were introduced to try to avoid some of these difficulties – however, these are typically frowed on by purists.

# Graphical Models

- We can use graphs to represent how objects can probabilistically interact with each other.
- Graphical Models and then a marriage between Graph and Probability theory.
- Many of the quantities that we would like to compute in a probability distribution can then be related to operations on the graph.
- The computational complexity of operations can often be related to the structure of the graph.
- Graphical Models are now used as a standard framework in Engineering, Statistics and Computer Science.

# Rules of probability

$p(x = \mathsf{x})$ : the probability of variable $x$ being in state x.

$$p(x = \mathsf{x}) = \begin{cases} 1 & \text{we are certain } x \text{ is in state x} \\ 0 & \text{we are certain } x \text{ is not in state x} \end{cases}$$

Values between 0 and 1 represent the degree of certainty of state occupancy.

---

## domain

$\mathrm{dom}(x)$ denotes the states $x$ can take. For example, $\mathrm{dom}(c) = \{\mathsf{heads}, \mathsf{tails}\}$. When summing over a variable $\sum_x f(x)$, the interpretation is that all states of $x$ are included, *i.e.* $\sum_x f(x) \equiv \sum_{\mathsf{s} \in \mathrm{dom}(x)} f(x = \mathsf{s})$.

---

## distribution

Given a variable, $x$, its domain $\mathrm{dom}(x)$ and a full specification of the probability values for each of the variable states, $p(x)$, we have a distribution for $x$.

---

## normalisation

The summation of the probability over all the states is 1:

$$\sum_{x \in \mathrm{dom}(x)} p(x = \mathsf{x}) = 1$$

We will usually more conveniently write $\sum_x p(x) = 1$.

# Operations

## AND

Use the shorthand $p(x, y) \equiv p(x \cap y)$ for $p(x \text{ and } y)$. Note that $p(y, x) = p(x, y)$.

---

## marginalisation

Given a joint distr. $p(x, y)$ the marginal distr. of $x$ is defined by

$$p(x) = \sum_y p(x, y)$$

More generally,

$$p(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = \sum_{x_i} p(x_1, \ldots, x_n)$$

# Conditional Probability and Bayes' Rule

The probability of event $x$ conditioned on knowing event $y$ (or more shortly, the probability of $x$ given $y$) is defined as

$$p(x|y) \equiv \frac{p(x,y)}{p(y)} = \frac{p(y|x)p(x)}{p(y)} \quad \text{(Bayes' rule)}$$

---

## Throwing darts

$$p(\text{region } 5|\text{not region } 20) = \frac{p(\text{region } 5, \text{not region } 20)}{p(\text{not region } 20)}$$
$$= \frac{p(\text{region } 5)}{p(\text{not region } 20)} = \frac{1/20}{19/20} = \frac{1}{19}$$

---

## Interpretation

$p(A = \text{a}|B = \text{b})$ should not be interpreted as 'Given the event $B = \text{b}$ has occurred, $p(A = \text{a}|B = \text{b})$ is the probability of the event $A = \text{a}$ occurring'. The correct interpretation should be '$p(A = \text{a}|B = \text{b})$ is the probability of $A$ being in state a under the constraint that $B$ is in state b'.

# Battleships

- Assume there are 2 ships, 1 vertical (ship 1) and 1 horizontal (ship 2), of 5 pixels each.
- Can be placed anywhere on the 10×10 grid, but cannot overlap.
- Let $s_1$ is the origin of ship 1 and $s_2$ the origin of ship 2
- Data $\mathcal{D}$ is a collection of query 'hit' or 'miss' responses.

$$p(s_1, s_2|\mathcal{D}) = \frac{p(\mathcal{D}|s_1, s_2)p(s_1, s_2)}{p(\mathcal{D})}$$

Let $X$ be the matrix of pixel occupancy

$$p(X|\mathcal{D}) = \sum_{s_1,s_2} p(X, s_1, s_2|\mathcal{D}) = \sum_{s_1,s_2} p(X|s_1, s_2)p(s_1, s_2|\mathcal{D})$$

`demoBattleships.m`

# Probability tables

The a priori probability that a randomly selected Great British person would live in England, Scotland or Wales, is $0.88$, $0.08$ and $0.04$ respectively.

We can write this as a vector (or probability table) :

$$\begin{pmatrix} p(Cnt = \mathsf{E}) \\ p(Cnt = \mathsf{S}) \\ p(Cnt = \mathsf{W}) \end{pmatrix} = \begin{pmatrix} 0.88 \\ 0.08 \\ 0.04 \end{pmatrix}$$

whose component values sum to 1.

The ordering of the components in this vector is arbitrary, as long as it is consistently applied.

# Probability tables

We assume that only three Mother Tongue languages exist : English (Eng), Scottish (Scot) and Welsh (Wel), with conditional probabilities given the country of residence, England (E), Scotland (S) and Wales (W). Using the state ordering:

$$MT = [\text{Eng}, \text{Scot}, \text{Wel}]; \qquad Cnt = [\text{E}, \text{S}, \text{W}]$$

we write a (fictitious) conditional probability table

$$p(MT|Cnt) = \begin{pmatrix} 0.95 & 0.7 & 0.6 \\ 0.04 & 0.3 & 0.0 \\ 0.01 & 0.0 & 0.4 \end{pmatrix}$$

# Probability tables

The distribution $p(Cnt, MT) = p(MT|Cnt)p(Cnt)$ can be written as a $3 \times 3$ matrix with (say) rows indexed by country and columns indexed by Mother Tongue:

$$\begin{pmatrix} 0.95 \times 0.88 & 0.7 \times 0.08 & 0.6 \times 0.04 \\ 0.04 \times 0.88 & 0.3 \times 0.08 & 0.0 \times 0.04 \\ 0.01 \times 0.88 & 0.0 \times 0.08 & 0.4 \times 0.04 \end{pmatrix} = \begin{pmatrix} 0.836 & 0.056 & 0.024 \\ 0.0352 & 0.024 & 0 \\ 0.0088 & 0 & 0.016 \end{pmatrix}$$

By summing a column, we have the marginal

$$p(Cnt) = \begin{pmatrix} 0.88 \\ 0.08 \\ 0.04 \end{pmatrix}$$

Summing the rows gives the marginal

$$p(MT) = \begin{pmatrix} 0.916 \\ 0.0592 \\ 0.0248 \end{pmatrix}$$

# Independence

Variables $x$ and $y$ are independent if knowing one event gives no extra information about the other event. Mathematically, this is expressed by

$$p(x, y) = p(x)p(y)$$

Independence of $x$ and $y$ is equivalent to

$$p(x|y) = p(x) \Leftrightarrow p(y|x) = p(y)$$

If $p(x|y) = p(x)$ for all states of $x$ and $y$, then the variables $x$ and $y$ are said to be independent.

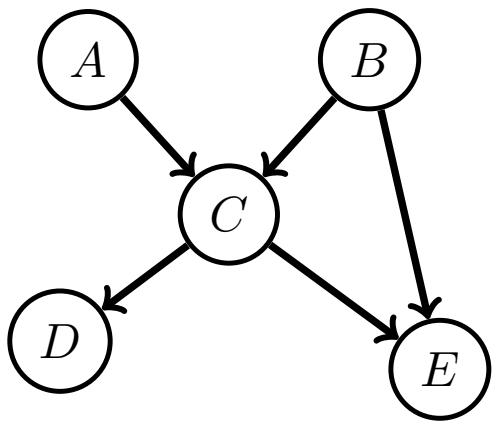
# Table of Contents

# Belief Networks (Bayesian Networks)

A belief network is a directed acyclic graph in which each node has associated the conditional probability of the node given its parents.

The joint distribution is obtained by taking the product of the conditional probabilities:

$$p(A, B, C, D, E) = p(A)p(B)p(C|A, B)p(D|C)p(E|B, C)$$




$p(E|B, C)$

# Example – Part I

Sally's burglar **A**larm is sounding. Has she been **B**urgled, or was the alarm triggered by an **E**arthquake? She turns the car **R**adio on for news of earthquakes.

---

## Choosing an ordering

Without loss of generality, we can write

$$\begin{aligned} p(A,R,E,B) &= p(A|R,E,B)p(R,E,B) \\ &= p(A|R,E,B)p(R|E,B)p(E,B) \\ &= p(A|R,E,B)p(R|E,B)p(E|B)p(B) \end{aligned}$$

---

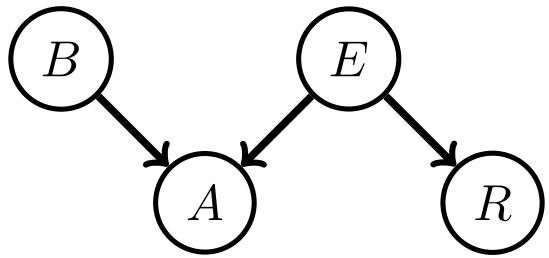
## Assumptions:

- The alarm is not directly influenced by any report on the radio, $p(A|R,E,B) = p(A|E,B)$
- The radio broadcast is not directly influenced by the burglar variable, $p(R|E,B) = p(R|E)$
- Burglaries don't directly 'cause' earthquakes, $p(E|B) = p(E)$

Therefore

$$p(A,R,E,B) = p(A|E,B)p(R|E)p(E)p(B)$$

# Example – Part II: Specifying the Tables




$p(A|B,E)$

| Alarm = 1 | Burglar | Earthquake |
|---|---|---|
| 0.9999 | 1 | 1 |
| 0.99 | 1 | 0 |
| 0.99 | 0 | 1 |
| 0.0001 | 0 | 0 |

$p(R|E)$

| Radio = 1 | Earthquake |
|---|---|
| 1 | 1 |
| 0 | 0 |

The remaining tables are $p(B = 1) = 0.01$ and $p(E = 1) = 0.000001$. The tables and graphical structure fully specify the distribution.

# Example Part III: Inference

Initial Evidence: The alarm is sounding

$$p(B=1|A=1) = \frac{\sum_{E,R} p(B=1,E,A=1,R)}{\sum_{B,E,R} p(B,E,A=1,R)}$$
$$= \frac{\sum_{E,R} p(A=1|B=1,E)p(B=1)p(E)p(R|E)}{\sum_{B,E,R} p(A=1|B,E)p(B)p(E)p(R|E)} \approx 0.99$$

---

Additional Evidence: The radio broadcasts an earthquake warning:

- A similar calculation gives $p(B=1|A=1,R=1) \approx 0.01$.
- Initially, because the alarm sounds, Sally thinks that she's been burgled. However, this probability drops dramatically when she hears that there has been an earthquake.

# Learning: maximum likelihood

- In modelling, we have a data model with some unknown parameters $\theta$.

  $$p(\text{data}|\theta)$$

- We can then learn $\theta$ by finding the process that would most likely have generated the observed data

  $$\theta_{optimal} = \arg\max_{\theta} p(\text{data}|\theta)$$

- This is an optimisation problem.
- For example, a coin has probability $\theta$ of coming up heads (H) and $1-\theta$ of coming up tails (T). Given data H,T,H:

  $$p(\text{H,T,H}|\theta) = \theta\,(1-\theta)\,\theta = \theta^2(1-\theta)$$

  If maximise this we find $\theta_{optimal} = 2/3$.
- For models in which not all variables are directly observable, a common algorithm is the 'EM' algorithm.

# Table of Contents

# Time-Series

A time-series is an ordered sequence:

$$x_{a:b} = \{x_a, x_{a+1}, \ldots, x_b\}$$

So that one can consider the 'past' and 'future' in the sequence. The $x$ can be either discrete or continuous.

---

## Biology

Gene sequences. Emphasis is on understanding sequences, filling in missing values, clustering sequences, detecting patterns. Hidden Markov Models are one of the key tools in this area.

---

## Finance

Price movement prediction.

---

## Planning

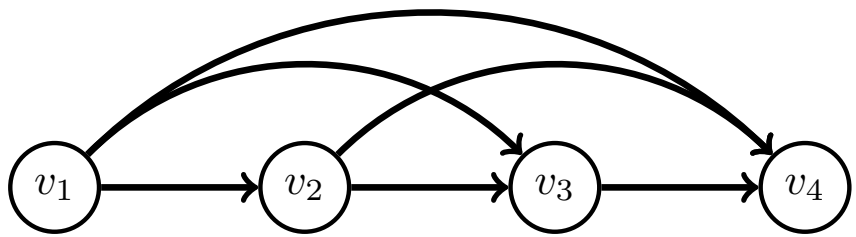
Forecasting – eg how many newspaper to deliver to retailers.

# Markov Models

For timeseries data $v_1, \ldots, v_T$, we need a model $p(v_{1:T})$. For causal consistency, it is meaningful to consider the decomposition

$$p(v_{1:T}) = \prod_{t=1}^{T} p(v_t | v_{1:t-1})$$

with the convention $p(v_t|v_{1:t-1}) = p(v_1)$ for $t = 1$.




---

## Independence assumptions

It is often natural to assume that the influence of the immediate past is more relevant than the remote past and in Markov models only a limited number of previous observations are required to predict the future.

# Markov Chain

Only the recent past is relevant:

$$p(v_t|v_1, \ldots, v_{t-1}) = p(v_t|v_{t-L}, \ldots, v_{t-1})$$

where $L \geq 1$ is the order of the Markov chain

$$p(v_{1:T}) = p(v_1)p(v_2|v_1)p(v_3|v_2) \ldots p(v_T|v_{T-1})$$

For a stationary Markov chain the transitions $p(v_t = s'|v_{t-1} = s) = f(s', s)$ are time-independent ('homogeneous').

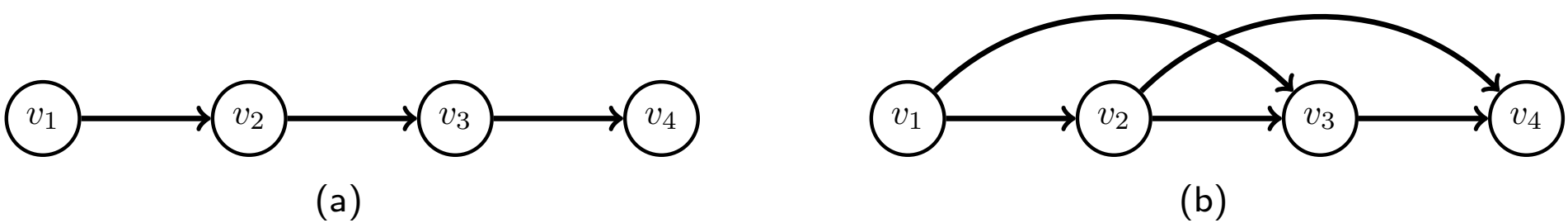



Figure: **(a)**: First order Markov chain. **(b)**: Second order Markov chain.

# Fitting Markov models (discrete variables)

## Single series

- Fitting a first-order stationary Markov chain by Maximum Likelihood corresponds to setting the transitions by counting the number of observed transitions in the sequence:

$$p(v_\tau = i | v_{\tau-1} = j) \propto \sum_{t=2}^{T} \mathbb{I}\left[v_t = i, v_{t-1} = j\right]$$

- The Maximum Likelihood setting for the initial first timestep distribution is $p(v_1 = i) \propto \sum_n \mathbb{I}\left[v_1^n = i\right]$.

---

## Multiple series

For a set of timeseries, $v_{1:T_n}^n, n = 1, \ldots, N$, the transition is given by counting all transitions across time and datapoints.

# Rock Paper Scissors

- Two people game: each player plays either Rock, Paper or Scissors.
- Paper beats Rock, Scissors beats Paper, Rock beats Scissors.
- Let's use the encoding $Rock = 1$, $Scissors = 2$, $Paper = 3$.

---

## First Order Markov Model

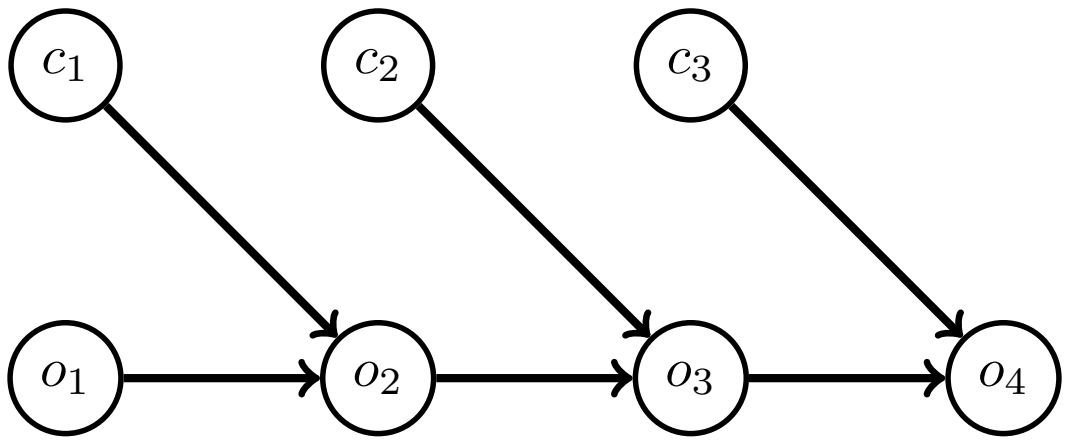
- $o_t \in \{1, 2, 3\}$ : human opponent play at time $t$.
- $c_t \in \{1, 2, 3\}$ : human opponent play at time $t$.
- The computer assumes the human moves based on what the human did on the last move.


`demoRockPaperScissorsMarkovHuman.m`

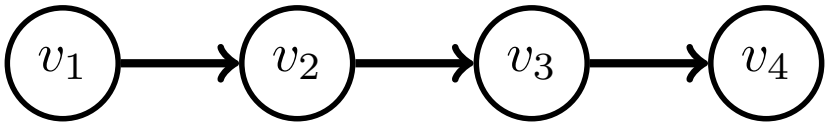
# Rock Paper Scissors

## First Order Markov Model with Computer past move

- The computer assumes the human moves based on what the human did on the last move and also on what the computer did on the last move.
- It is an exercise in the afternoon to program this.



# Markov Chains

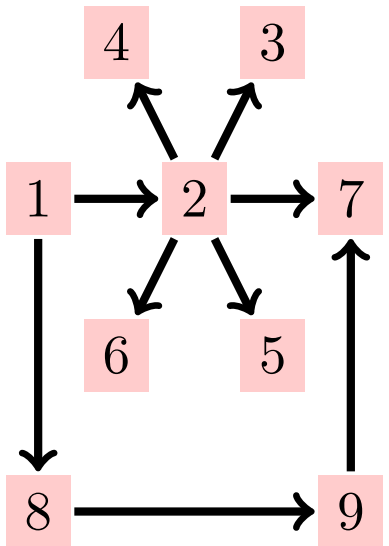



$$p(v_1, \ldots, v_T) = \underbrace{p(v_1)}_{\text{initial}} \prod_{t=2}^{T} \underbrace{p(v_t|v_{t-1})}_{\text{Transition}}$$

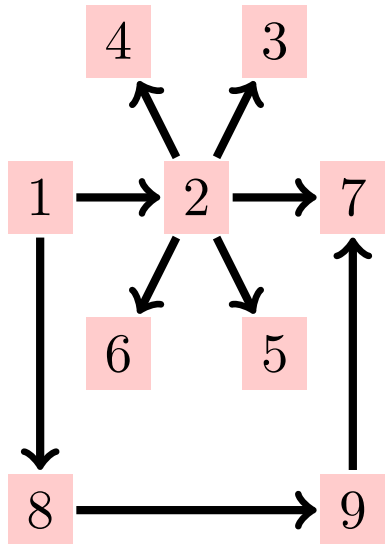
---

## State transition diagram

Nodes represent states of the variable $v$ and arcs non-zero elements of the transition $p(v_t|v_{t-1})$



# Most probable and shortest paths




- The shortest (unweighted) path from state 1 to state 7 is $1-2-7$.
- The most probable path from state 1 to state 7 is $1-8-9-7$ (assuming uniform transition probabilities). The latter path is longer but more probable since for the path $1-2-7$, the probability of exiting state 2 into state 7 is $1/5$.

# Equilibrium distribution

- It is interesting to know how the marginal $p(x_t)$ evolves through time:

$$p(x_t = i) = \sum_j \underbrace{p(x_t = i | x_{t-1} = j)}_{M_{ij}} p(x_{t-1} = j)$$

- $p(x_t = i)$ is the frequency that we visit state $i$ at time $t$, given we started from $p(x_1)$ and randomly drew samples from the transition $p(x_\tau | x_{\tau-1})$.
- As we repeatedly sample a new state from the chain, the distribution at time $t$, for an initial distribution $\mathbf{p}_1(i)$ is

$$\mathbf{p}_t = \mathbf{M}^{t-1} \mathbf{p}_1$$

If, for $t \to \infty$, $\mathbf{p}_\infty$ is independent of the initial distribution $\mathbf{p}_1$, then $\mathbf{p}_\infty$ is called the equilibrium distribution of the chain:

$$\mathbf{p}_\infty = \mathbf{M} \mathbf{p}_\infty$$

- The equil. distribution is proportional to the eigenvector with unit eigenvalue of the transition matrix.

# PageRank

Define the matrix

$$A_{ij} = \begin{cases} 1 & \text{if website } j \text{ has a hyperlink to website } i \\ 0 & \text{otherwise} \end{cases}$$

From this we can define a Markov transition matrix with elements

$$M_{ij} = \frac{A_{ij}}{\sum_{i'} A_{i'j}}$$

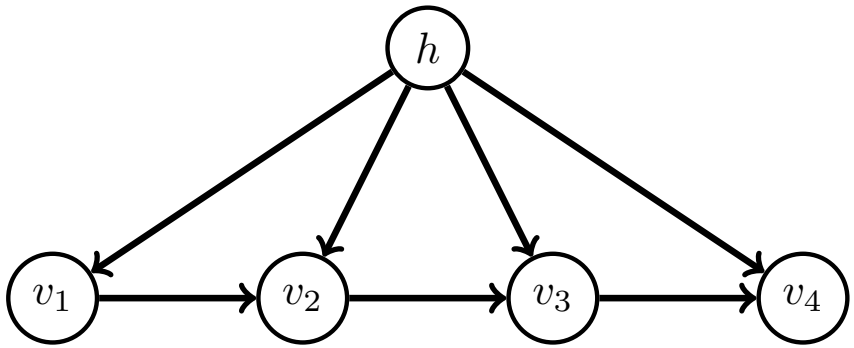
- If we jump from website to website, the equilibrium distribution component $p_\infty(i)$ is the relative number of times we will visit website $i$. This has a natural interpretation as the 'importance' of website $i$.
- For each website $i$ a list of words associated with that website is collected. After doing this for all websites, one can make an 'inverse' list of which websites contain word $w$. When a user searches for word $w$, the list of websites that contain word is then returned, ranked according to the importance of the site.

# Gene Clustering

- Consider the 20 fictitious gene sequences below presented in an arbitrarily chosen order.
- Each sequence consists of 20 symbols from the set $\{A, C, G, T\}$.
- The task is to try to cluster these sequences into two groups, based on the (perhaps biologically unrealistic) assumption that gene sequences in the same cluster follow a stationary Markov chain.

| | | | |
|---|---|---|---|
| CATAGGCATTCTATGTGCTG | CCAGTTACGGACGCCGAAAG | TGGAACCTTAAAAAAAAAAA | GTCTCCTGCCCTCTCTGAAC |
| GTGCCTGGACCTGAAAAGCC | CGGCCGCGCCTCCGGGAACG | AAAGTGCTCTGAAAACTCAC | ACATGAACTACATAGTATAA |
| GTTGGTCAGCACACGGACTG | CCTCCCCTCCCCTTTCCTGC | CACTACGGCTACCTGGGCAA | CGGTCCGTCCGAGGCACTC |
| TAAGTGTCCTCTGCTCCTAA | CACCATCACCCTTGCTAAGG | AAAGAACTCCCCTCCCTGCC | CAAATGCCTCACGCGTCTCA |
| GCCAAGCAGGGTCTCAACTT | CATGGACTGCTCCACAAAGG | AAAAAAACGAAAAACCTAAG | GCGTAAAAAAAGTCCTGGGT |

# Mixture of Markov models




- The discrete hidden variable $\mathrm{dom}(h) = \{1, \ldots, H\}$ indexes the Markov chain

$$\prod_t p(v_t | v_{t-1}, h)$$

- Such models can be useful as simple sequence clustering tools.

# Mixture of Markov models

Given a set of sequences $\mathcal{V} = \{v_{1:T}^n, n = 1, \ldots, N\}$, how might we cluster them?

- We can define a mixture model for a single sequence $v_{1:T}$.
- Here we assume each component model is first order Markov

$$p(v_{1:T}) = \sum_{h=1}^{H} p(h)p(v_{1:T}|h) = \sum_{h=1}^{H} p(h) \prod_{t=1}^{T} p(v_t|v_{t-1}, h)$$

- Clustering can then be achieved by finding the maximum likelihood parameters $p(h)$, $p(v_t|v_{t-1}, h)$ and subsequently assigning the clusters according to $p(h|v_{1:T}^n)$.

# Clustering Genes

- After running the EM maximum likelihood algorithm, we can then assign each of the sequences by examining $p(h = 1|v^n_{1:T})$.
- If this posterior probability is greater than 0.5, we assign it to cluster 1, otherwise to cluster 2.
- Using this procedure, we find the following clusters:

| | |
|---|---|
| CATAGGCATTCTATGTGCTG | TGGAACCTTAAAAAAAAAAA |
| CCAGTTACGGACGCCGAAAG | GTCTCCTGCCCTCTCTGAAC |
| CGGCCGCGCCTCCGGGAACG | GTGCCTGGACCTGAAAAGCC |
| ACATGAACTACATAGTATAA | AAAGTGCTCTGAAAACTCAC |
| GTTGGTCAGCACACGGACTG | CCTCCCCTCCCCTTTCCTGC |
| CACTACGGCTACCTGGGCAA | TAAGTGTCCTCTGCTCCTAA |
| CGGTCCGTCCGAGGCACTCG | AAAGAACTCCCCTCCCTGCC |
| CACCATCACCCTTGCTAAGG | AAAAAAACGAAAAACCTAAG |
| CAAATGCCTCACGCGTCTCA | GCGTAAAAAAAGTCCTGGGT |
| GCCAAGCAGGGTCTCAACTT | |
| CATGGACTGCTCCACAAAGG | |

where sequences in the first column are assigned to cluster 1, and sequences in the second column to cluster 2.

demoMixMarkov.m

# Table of Contents

# Hidden Markov Models

The HMM defines a Markov chain on hidden variables $h_{1:T}$. The observed variables depend on the hidden variables through an emission $p(v_t|h_t)$. This defines a joint distribution

$$p(h_{1:T}, v_{1:T}) = p(v_1|h_1)p(h_1)\prod_{t=2}^{T} p(v_t|h_t)p(h_t|h_{t-1})$$

$p(h_t|h_{t-1})$ and $p(v_t|h_t)$ are constant through time.

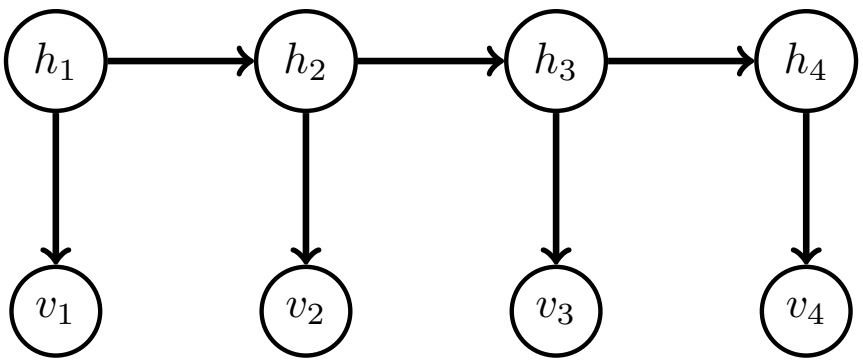



Figure: A first order hidden Markov model with 'hidden' variables $\mathrm{dom}(h_t) = \{1, \ldots, H\}$, $t = 1 : T$. The 'visible' variables $v_t$ can be either discrete or continuous.

*Probably the most common timeseries model in all of engineering/biology/physical science.*

# HMM parameters

## Transition Distribution

For a stationary HMM the transition distribution $p(h_{t+1}|h_t)$ is defined by the $H \times H$ transition matrix

$$A_{i',i} = p(h_{t+1} = i'|h_t = i)$$

and an initial distribution

$$a_i = p(h_1 = i).$$

---

## Emission Distribution

For a stationary HMM and emission distribution $p(v_t|h_t)$ with discrete states $v_t \in \{1, \ldots, V\}$, we define a $V \times H$ emission matrix

$$B_{i,j} = p(v_t = i|h_t = j)$$

For continuous outputs, $h_t$ selects one of $H$ possible output distributions $p(v_t|h_t)$, $h_t \in \{1, \ldots, H\}$.

# The classical inference problems

| | | | |
|---|---|---|---|
| **Filtering** | (Inferring the present) | $p(h_t \mid v_{1:t})$ | |
| **Prediction** | (Inferring the future) | $p(h_t \mid v_{1:s})$ | $t > s$ |
| **Smoothing** | (Inferring the past) | $p(h_t \mid v_{1:u})$ | $t < u$ |
| **Likelihood** | | $p(v_{1:T})$ | |
| **Most likely path** | (Viterbi alignment) | $\operatorname{argmax}_{h_{1:T}} p(h_{1:T} \mid v_{1:T})$ | |

For prediction, one is also often interested in $p(v_t \mid v_{1:s})$ for $t > s$.

---

## Uses of the HMM

- Biology: gene sequence analysis
- Computer Vision: tracking of people in videos
- Signal Processing: cleaning up noise corrupted music signals
- Speech Recognition (dominant approach until recently)
- Engineering: the famous 'Kalman Filter' is a special case of a HMM (with continuous variables)
- Weather Forecasting
- Financial prediction, product purchase prediction, modelling the economy
- Military: tracking ballistic objects
- ... and many more ...

# Filtering $p(h_t|v_{1:t})$

$$p(h_t, v_{1:t}) = \sum_{h_{t-1}} p(h_t, h_{t-1}, v_{1:t-1}, v_t) \tag{1}$$

$$= \sum_{h_{t-1}} p(v_t|\cancel{v_{1:t-1}}, h_t, \cancel{h_{t-1}}) p(h_t|\cancel{v_{1:t-1}}, h_{t-1}) p(v_{1:t-1}, h_{t-1}) \tag{2}$$

$$= \sum_{h_{t-1}} p(v_t|h_t) p(h_t|h_{t-1}) p(h_{t-1}, v_{1:t-1}) \tag{3}$$

Hence if we define $\alpha(h_t) \equiv p(h_t, v_{1:t})$ (3) above gives the $\alpha$-recursion

$$\alpha(h_t) = \underbrace{p(v_t|h_t)}_{\text{corrector}} \underbrace{\sum_{h_{t-1}} p(h_t|h_{t-1}) \alpha(h_{t-1})}_{\text{predictor}}, \qquad t > 1$$

$$p(h_t|v_{1:t}) = \frac{\alpha(h_t)}{\sum_{h_t} \alpha(h_t)}$$

Similar recursions do smoothing and Viterbi in $O(T)$ time.

# Prediction

## Predicting the future hidden variable

$$p(h_{t+1}|v_{1:t}) = \sum_{h_t} p(h_{t+1}|h_t) \underbrace{p(h_t|v_{1:t})}_{filtering}$$

---

## Predicting the future observation

The one-step ahead predictive distribution is given by

$$p(v_{t+1}|v_{1:t}) = \sum_{h_t, h_{t+1}} p(v_{t+1}|h_{t+1}) p(h_{t+1}|h_t) p(h_t|v_{1:t})$$

# Localisation example – Part I

Problem: You're asleep upstairs in your house and awoken by a burglar on the ground floor. You want to figure out where the burglar might be based on a sequence of noise information.

You mentally partition the ground floor into a $5 \times 5$ grid. For each grid position

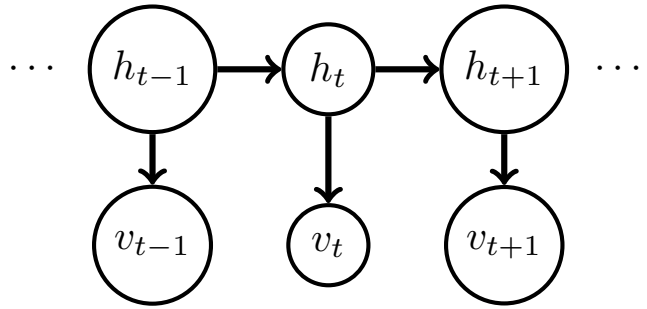
- you know the probability that if someone is in that position the floorboard will creak
- you know the probability that if someone is in that position he will bump into something in the dark
- you assume that the burglar can move only into a neighbor grid square with uniform probability

Prob. of creak

Prob. of bump

# Localisation example – Part II

We can represent the scenario using a HMM where




- The hidden variable $h_t$ represents the position of the burglar in the grid at time $t$

  $h_t \in \{1, \ldots, 25\}$

- The visible variable $v_t$ represents creak/bump at time $t$

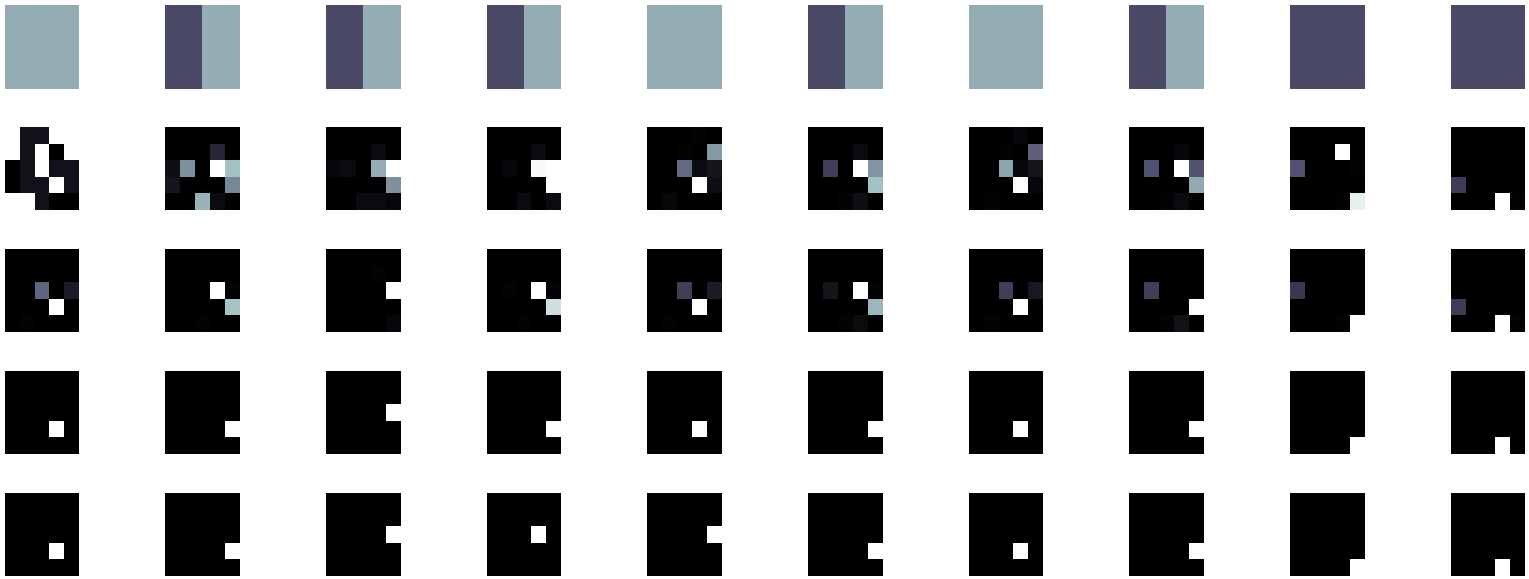
  v=1: no creak, no bump

  v=2: creak, no bump

  v=3: no creak, bump

  v=4: creak, bump

# Localisation example – Part III



(top) Observed creaks and bumps for 10 time-steps

(below top) Filtering $p(h_t|v_{1:t})$

(middle) Smoothing $p(h_t|v_{1:10})$

(above bottom) Most likely sequence $\underset{h_{1:T}}{\operatorname{argmax}}\, p(h_{1:T}|v_{1:T})$

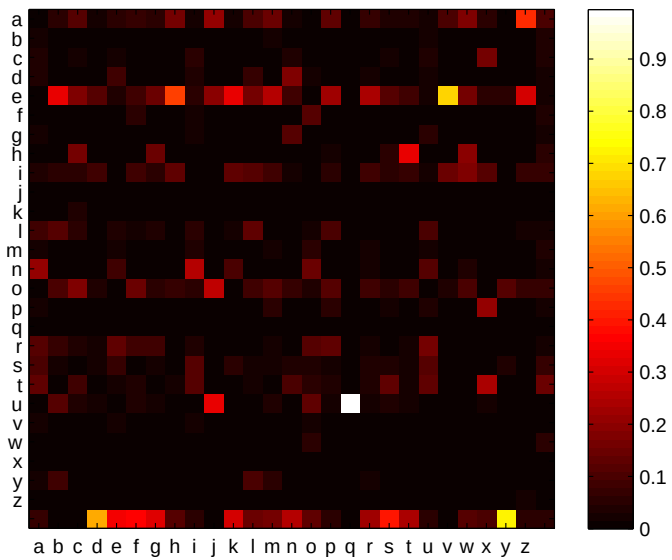
(bottom) True Burglar position

# Natural Language Model Example – Part I

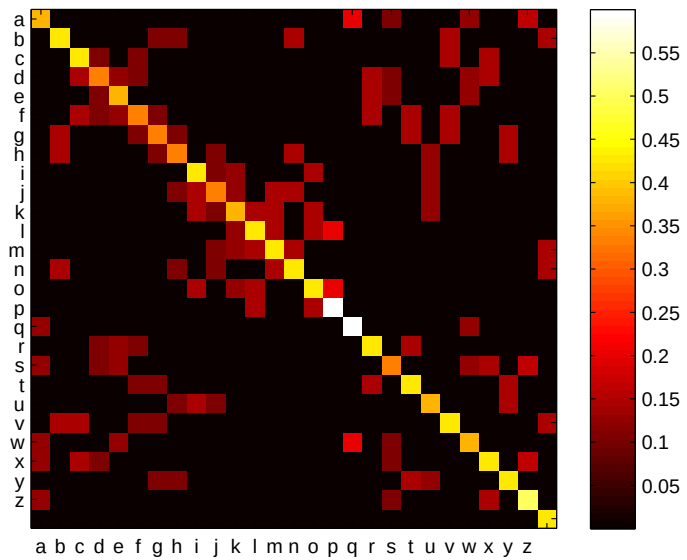
Problem: A 'stubby finger' typist has the tendency to hit either the correct key or a neighbouring key. Given a typed sequence you want to infer what is the most likely word that this corresponds to.

- The hidden variable $h_t$ represents the intended letter at time $t$
- The visible variable $v_t$ represents the letter that was actually typed at time $t$

We assume that there are 27 keys: lower case a to lower case z and the space bar.




Transition $p(h_t = j|h_{t-1} = i)$




Emission $p(v_t = j|h_t = i)$

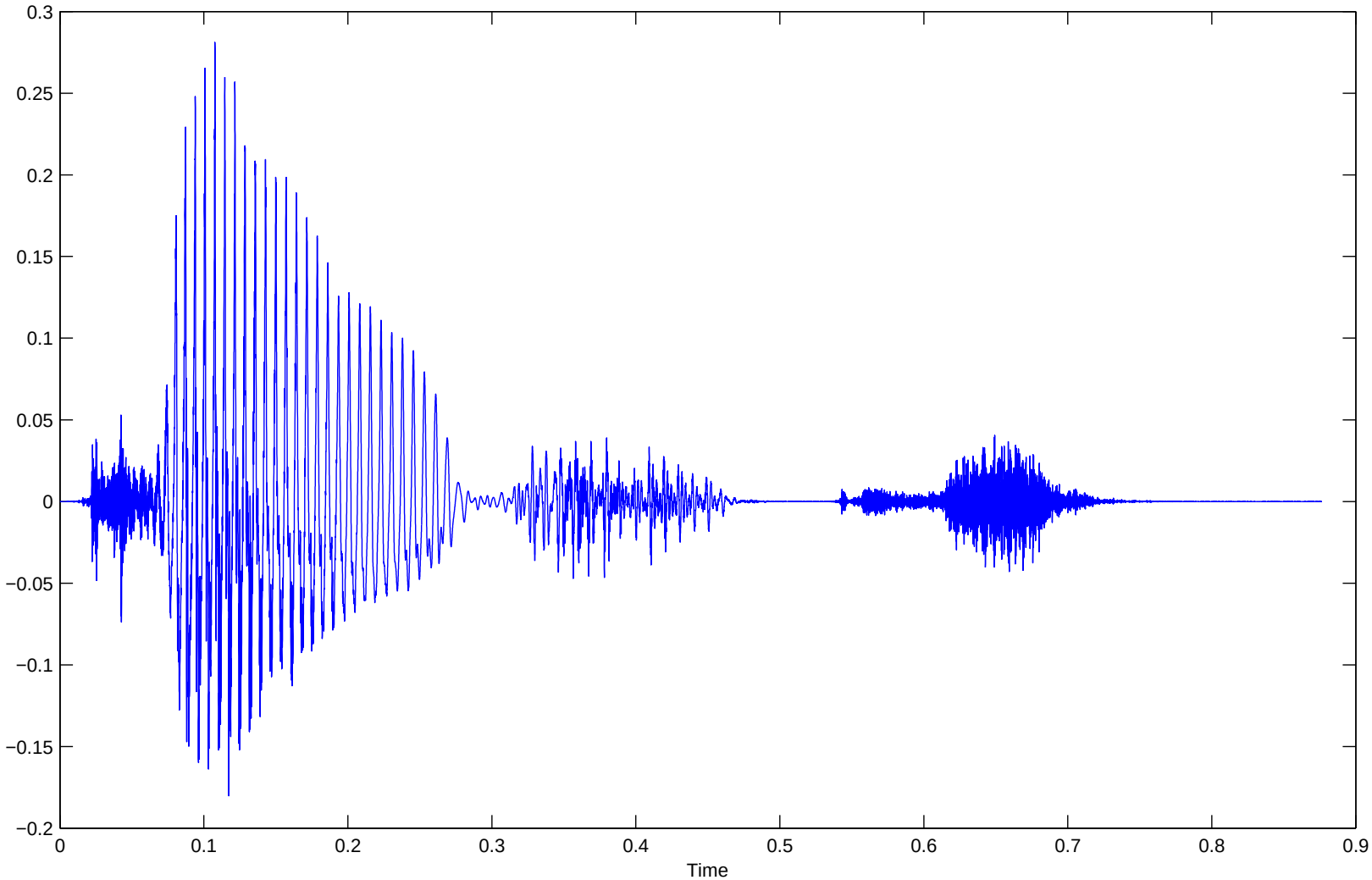
# Natural Language Model Example – Part II

Given the typed sequence `kezrninh` what is the most likely word that this corresponds to?

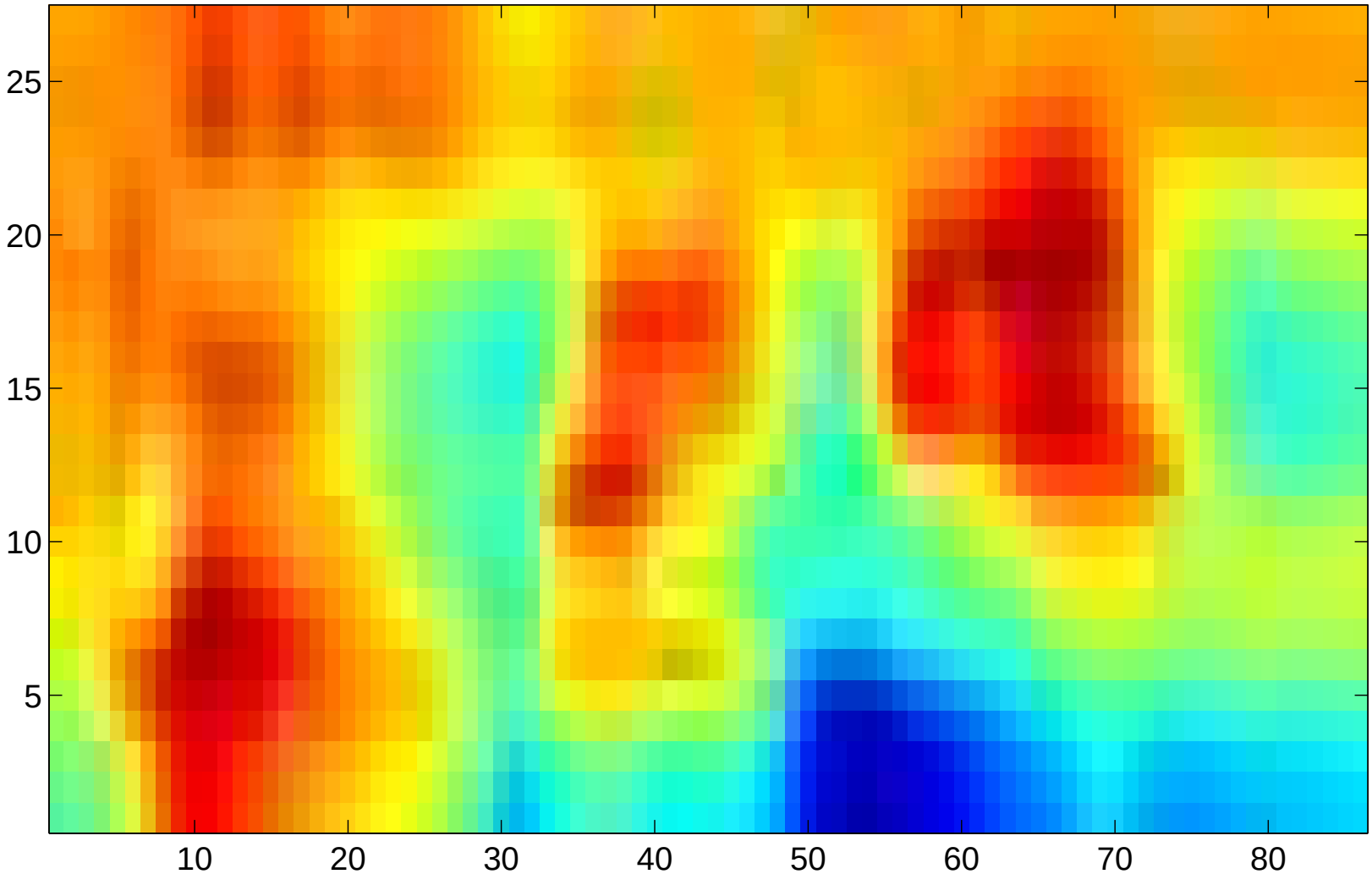
- Listing the 200 most likely hidden sequences (using a form of Viterbi)
- Discard those that are not in a standard English dictionary
- Take the most likely proper English word as the intended typed word

. . . and the answer is . . .

# Speech Recognition: raw signal

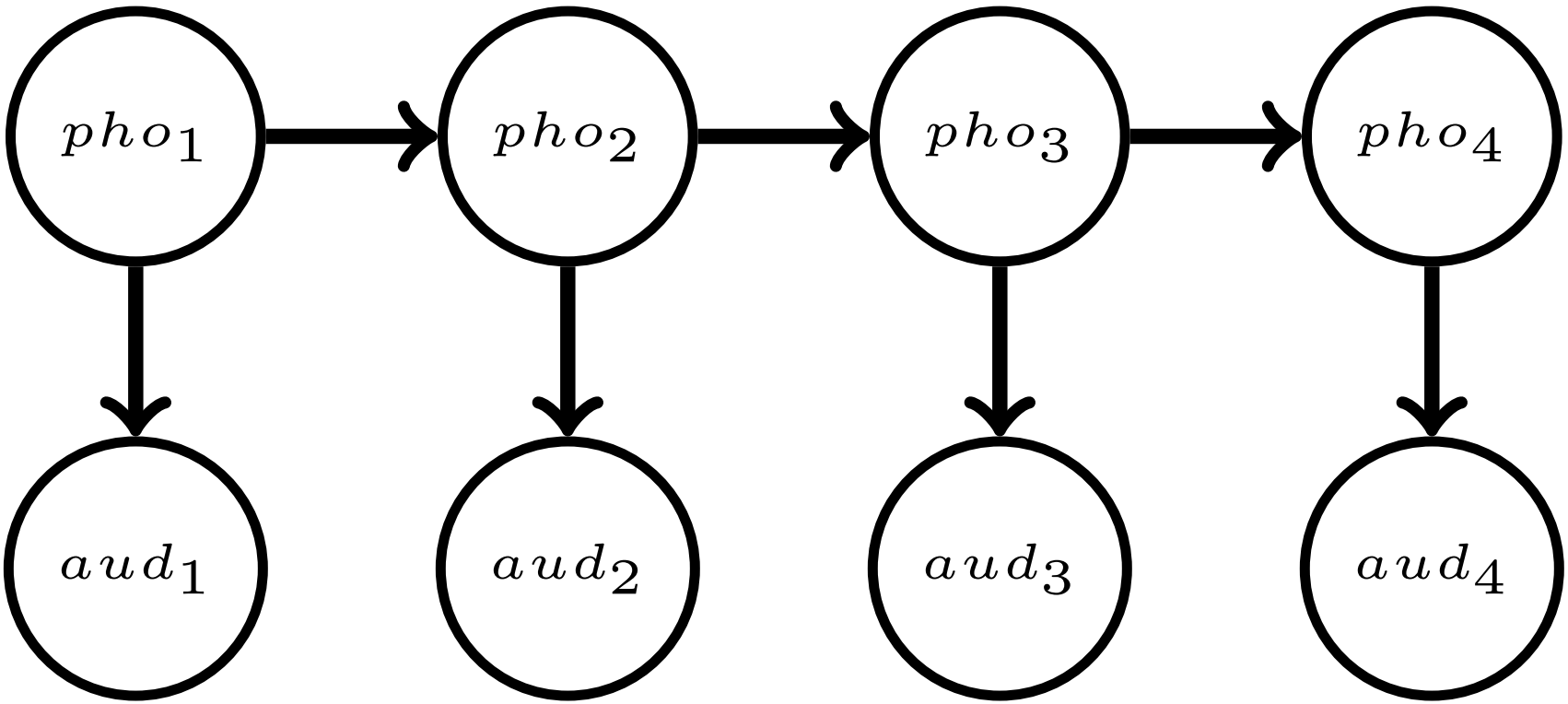


# ‘Spectogram’representation




Horizontal axis is time. Vertical axis is frequency.

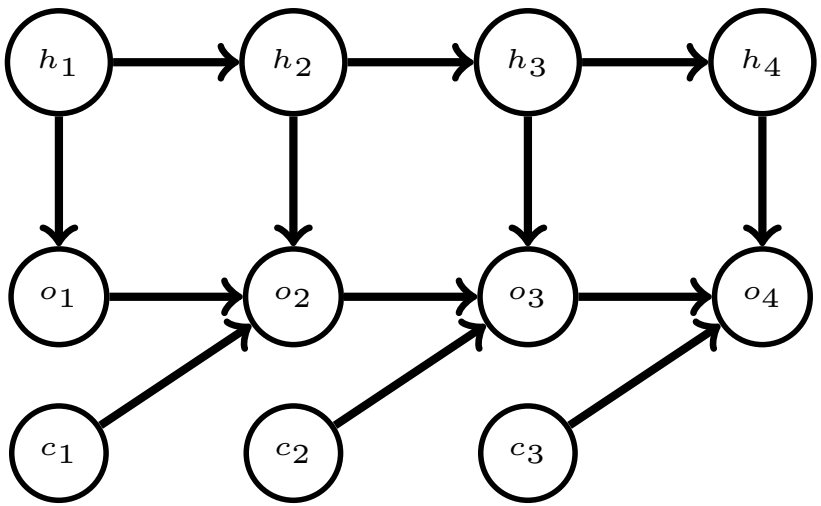
# Speech Recognition



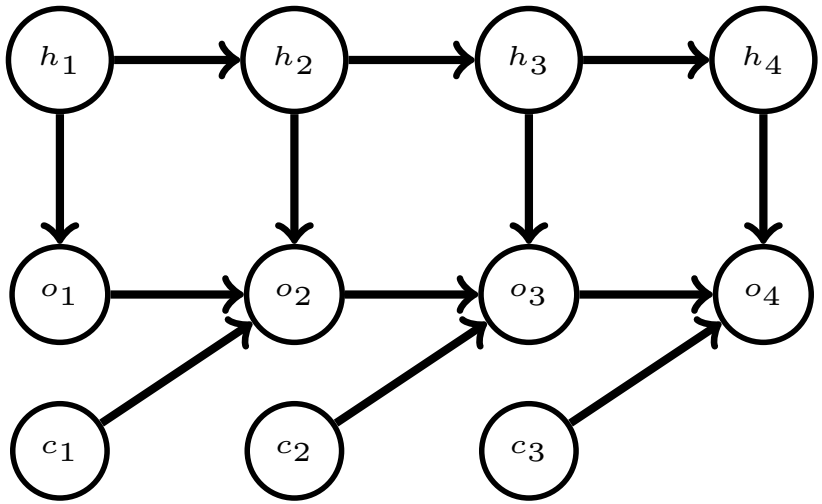

$pho$: phoneme (letter)
$aud$: audio signal (neural representation)

# Rock Paper Scissors, again!

- Let's assume there are four kinds of play
  1. Play randomly
  2. Do what you did last time
  3. Do what the computer did last time
  4. Play a different move to either your own or the computer's last time.
- We can use an index $h_t \in \{1, 2, 3, 4\}$ to denote the strategy
- Let's assume that the strategy doesn't change very often, so the strategy $h_t$ is most likely to to be the same as $h_{t-1}$.



# Rock Paper Scissors, again!




demoHMMRockPaperScissors.m

- As we gather information about the plays the human and computer makes, we can calculate the filtered distribution $p(h_t|o_{1:t-1}, c_{1:t-1})$ of the likely strategy that the human is currently playing.
- We can use this to then predict what move the human is likely to make at the next timestep.
- An exercise this afternoon is to extend the demo of this to include another strategies.

# Table of Contents

# Auto-Regressive Models

- The timeseries value $v_t$ is modelled by a weighted sum of previous values:

$$v_t \approx \sum_{l=1}^{L} a_l v_{t-l}$$

  where the $a_l$ are called the 'AR coefficients'.
- We can view this then as a form of regression, and find the AR cofficients by minimising the squared loss

$$\sum_t \left( v_t - \sum_{l=1}^{L} a_l v_{t-l} \right)^2$$

- This is a simple quadratic function of the AR coefficients and easy to optimise.

# Auto-Regressive Models: The Belief Network

$$v_t = \sum_{l=1}^{L} a_l v_{t-l} + \eta_t, \qquad \eta_t \sim \mathcal{N}\left(\eta_t | \mu, \sigma^2\right)$$

where $\mathbf{a} = (a_1, \ldots, a_L)^\mathsf{T}$ are called the AR coefficients and $\sigma^2$ is called the innovation noise. The model predicts the future based on a linear combination of the previous $L$ observations. This is an $L^{th}$ order Markov model:

$$p(v_{1:T}) = \prod_{t=1}^{T} p(v_t | v_{t-1}, \ldots, v_{t-L}), \qquad \text{with } v_i = \emptyset \text{ for } i \leq 0$$

with

$$p(v_t | v_{t-1}, \ldots, v_{t-L}) = \mathcal{N}\left(v_t \middle| \sum_{l=1}^{L} a_l v_{t-l}, \sigma^2\right)$$

# Fitting a trend

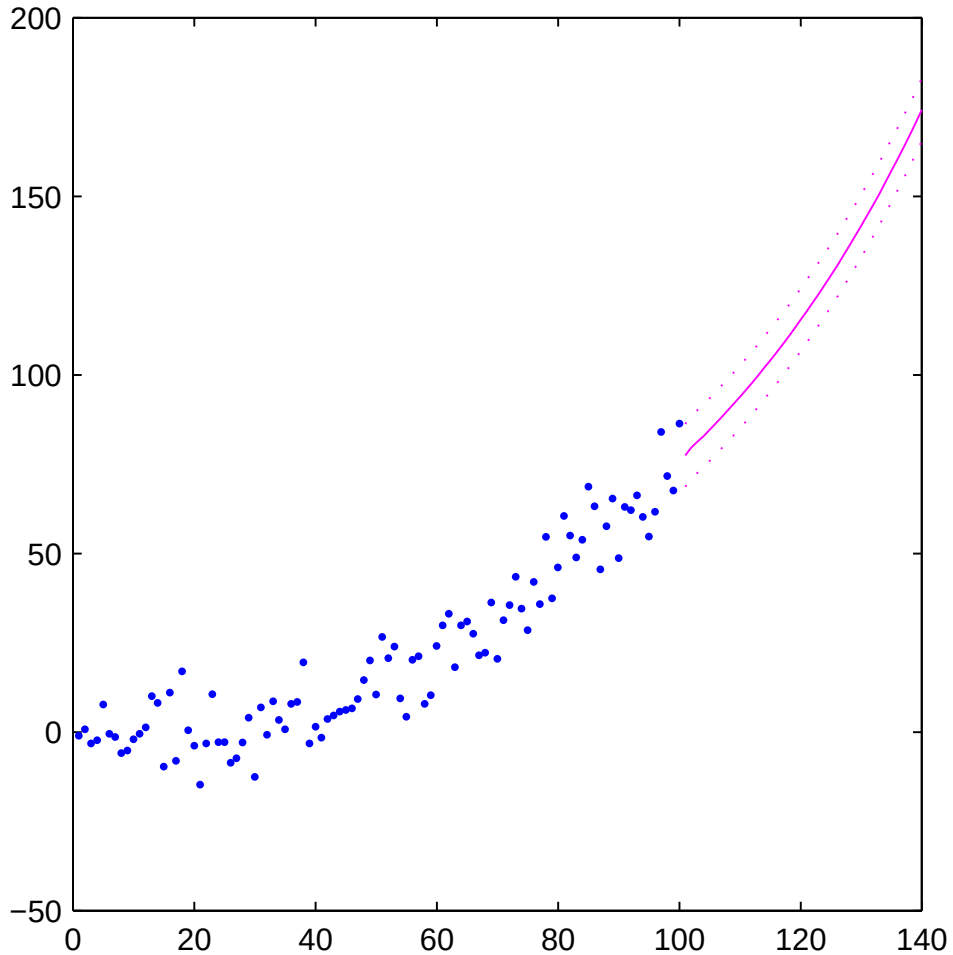



Figure: Fitting an order 3 AR model to the training points. The $x$ axis represents time, and the $y$ axis the value of the timeseries. The solid line is the mean prediction and the dashed lines $\pm$ one standard deviation around the mean predi.

# Uses of AR models

- AR models are heavily used in financial time-series prediction, being able to capture simple trends in the data.
- They are probably the most common continuous variable timeseries model.
- Another common application area is in speech processing whereby for a one-dimensional speech signal partitioned into windows of length $T$, the AR coefficients best able to describe the signal in each window are found.
- These AR coefficients then form a compressed representation of the signal and subsequently transmitted for each window, rather than the original signal itself.
- The signal can then be approximately reconstructed based on the AR coefficients.
- Such a representation is used for example in mobile phone and known as a linear predictive vocoder.

# Discrete Fourier Transform

For a sequence $x_{0:N-1}$ the DFT $f_{0:N-1}$ is defined as

$$f_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi \mathrm{i}}{N} kn}, \qquad k = 0, \ldots, N-1$$

$f_k$ is a (complex) representation as to how much frequency $k$ is present in the sequence $x_{0:N-1}$. The power of component $k$ is defined as the absolute length of the complex $f_k$.

---

## Spectrogram

- Given a timeseries $x_{1:T}$ the spectrogram at time $t$ is a representation of the frequencies present in a window localised around $t$.
- For each window one computes the Discrete Fourier Transform, from which we obtain a vector of log power in each frequency. The window is then moved (usually) one step forward and the DFT recomputed.

# Nightingale

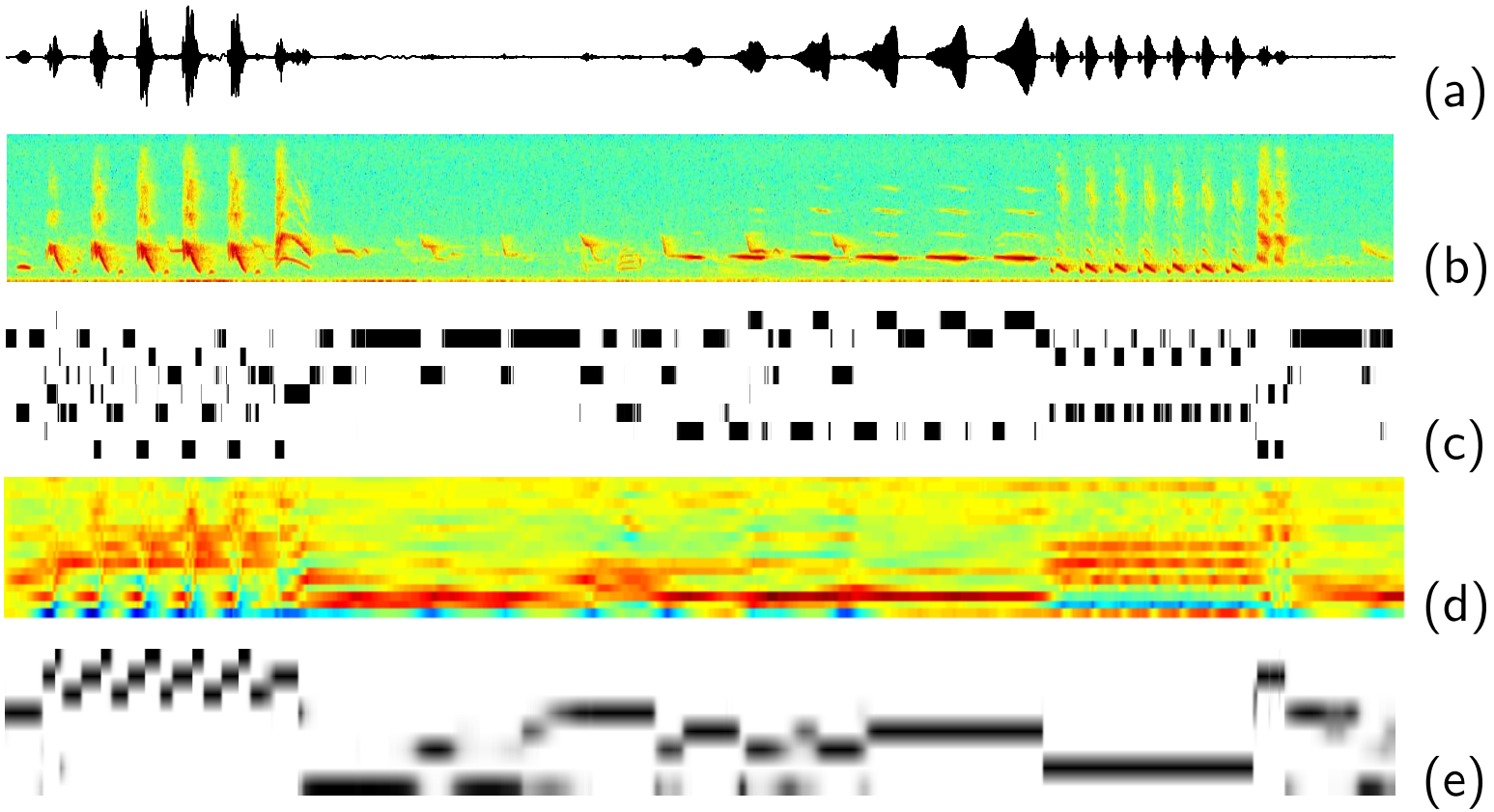



Figure: **(a)**: The raw recording of 5 seconds of a nightingale song (with additional background birdsong). **(b)**: Spectrogram of (a) up to 20,000 Hz. **(c)**: Clustering of the results in panel (b) using an 8 component Gaussian mixture model. The index (from 1 to 8) of the component most probably responsible for the observation is indicated vertically in black. **(d)**: The 20 AR coefficients learned using $\sigma_v^2 = 0.001$, $\sigma_h^2 = 0.001$. **(e)**: Clustering the results in panel (d) using a Gaussian mixture model with 8 components. The AR components group roughly according to the different song regimes.

# Resources

- You can download a free book on modelling and timeseries from `http://www.cs.ucl.ac.uk/staff/d.barber/brml`
- This includes also software (Matlab and Julia).