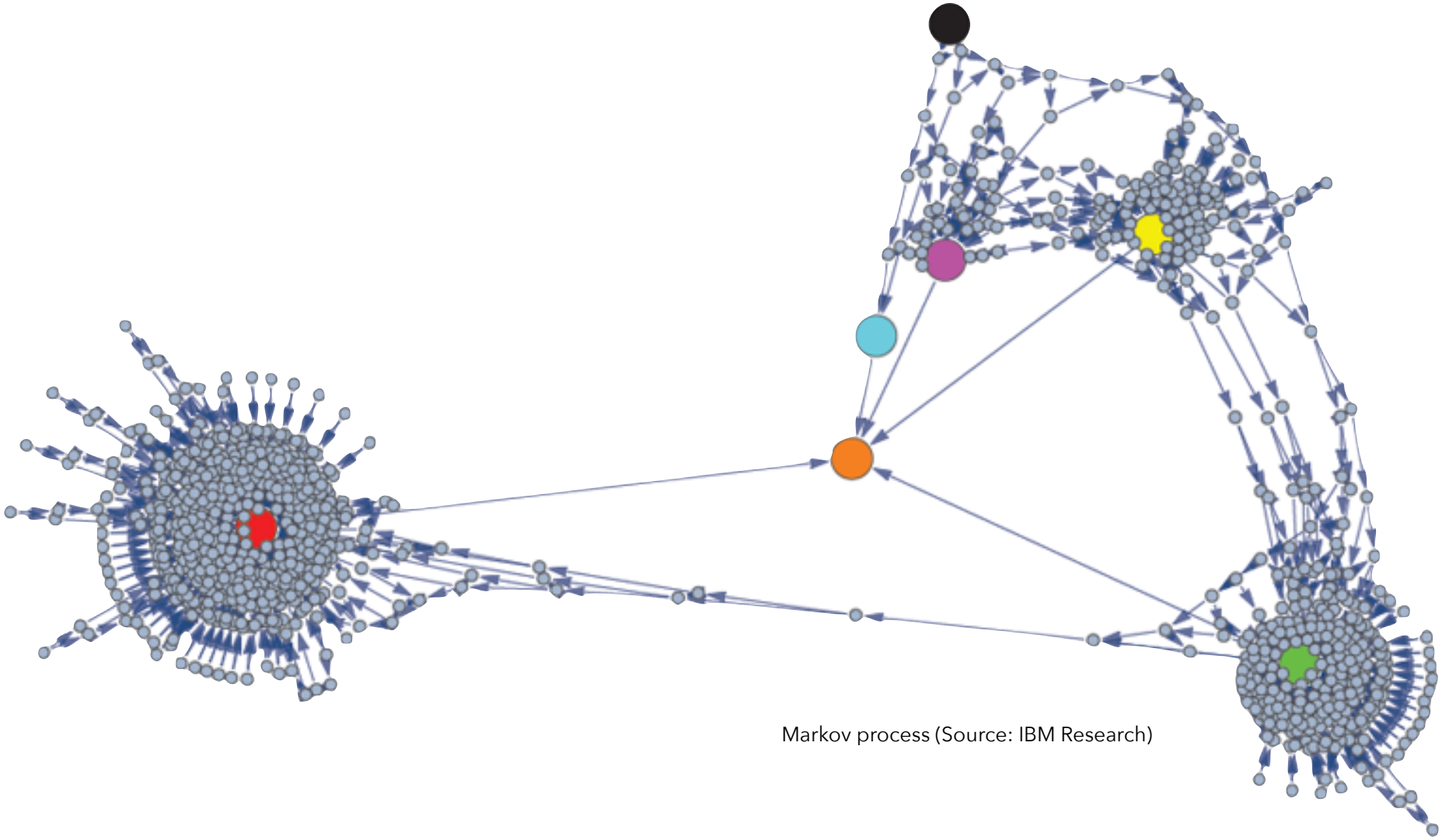
Section 2

Probabilistic Modeling



Markov process (Source: IBM Research)

Axioms of Probability

The axioms of probability are mathematical rules that probability must satisfy.

Let ***A*** and ***B*** be events. Let $P(A)$ denote the probability of the event A. The axioms of probability are these three conditions on the ***function P***:

- For every event A, $P(A) \geq 0$. (There is no such thing as a negative probability.)

- The probability of the entire outcome space S is 100%. (The outcome space contains every possible outcome.)

- If two events A and B are disjoint, $P(A \cup B) = P(A) + P(B)$, the probability that either of the events happens is the sum of the probabilities that each happens.

  If two events A and B are **not** disjoint, $P(A \cup B) = P(A) + P(B) - P(AB)$.

  Note: $P(A) = P(AB^C \cup AB) = P(AB^C) + P(AB)$,
  $P(B) = P(A^CB \cup AB) = P(A^CB) + P(AB)$

Conditional Probability

William Feller. An Introduction to Probability Theory and Its Applications (Third Edition). John Wiley & Sons,1968.

Conditional probability is a measure of the probability of an event occurring given that another event has occurred. If the event of interest is *A* and the event *B* is known or assumed to have occurred, the conditional probability of *A* given *B*—the probability of *A* under the condition *B*—is usually written as $P(A \mid B)$, or sometimes $P_B(A)$.

For example, the probability that any given person has a cough on any given day may be only 5%. But if we know or assume that the person has a cold, then they are much more likely to be coughing. The conditional probability of coughing given that you have a cold might thus be a much higher than 5%.

For $P(B) \geq 0$, we have

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

the probability that both events *A* and *B* occur

Bayes' Theorem

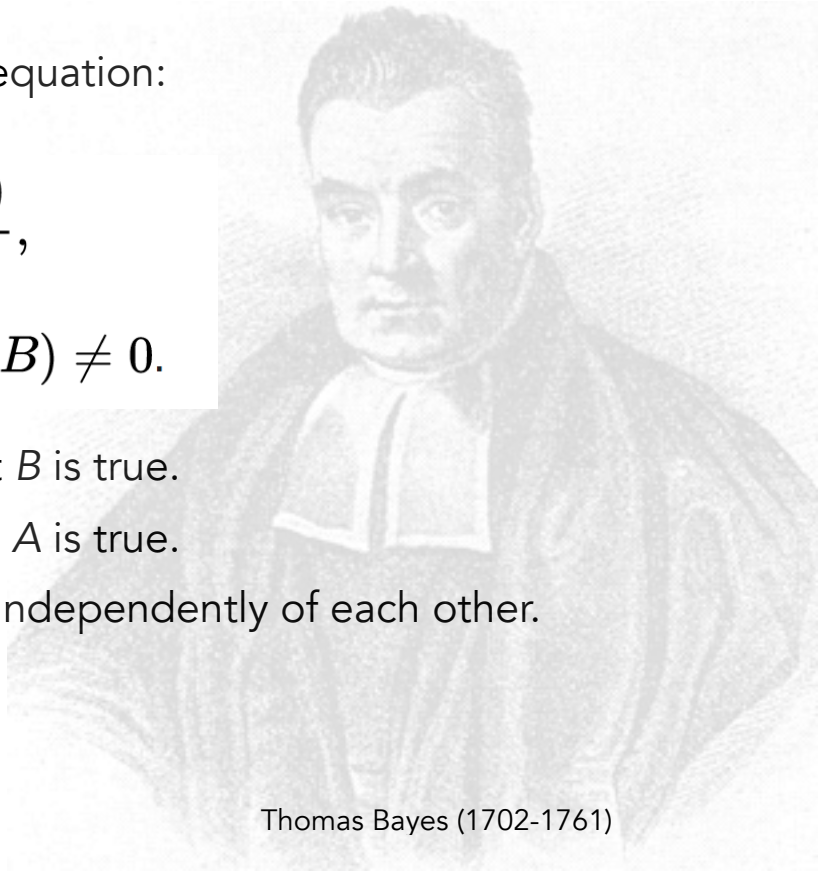
In probability theory and statistics, Bayes' theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event (a.k.a. *prior*). For example, if the chance of having a disease is related to age, then, using Bayes' theorem, a person's age can be used to more accurately assess the probability that they have the disease, compared to the assessment of the probability of the disease made without knowledge of the person's age.

Bayes' theorem is stated mathematically as the following equation:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)},$$

where A and B are events and $P(B) \neq 0$.

- $P(A \mid B)$ is the likelihood of event A occurring given that B is true.
- $P(B \mid A)$ is the likelihood of event B occurring given that A is true.
- $P(A)$ and $P(B)$ are the probabilities of observing A and B independently of each other.



Thomas Bayes (1702-1761)

The Bayesian Trap

https://www.youtube.com/watch?v=R13BD8qKeTg

Bayesian interpretation

The interpretation of Bayes' theorem depends on "the meaning of probabilities" as associated with the terms $P(x \mid y)$, $P(x)$, $P(y)$.

In the *Bayesian interpretation*, probability measures a "degree of belief." Bayes' theorem then links the degree of belief in a proposition **before** and **after** accounting for evidence.

For example, suppose it is believed with 50% certainty that a coin is twice as likely to land heads than tails. If the coin is flipped a number of times and the outcomes observed, that degree of belief may rise, fall or remain the same depending on the results.

For proposition A and evidence B,

- $P(A)$, the *prior*, is the initial degree of belief in A.
- $P(A \mid B)$, the *posterior*, is the degree of belief having accounted for B.
- $\frac{P(B \mid A)}{P(B)}$ represents the *support* B provides for A.

Randomness

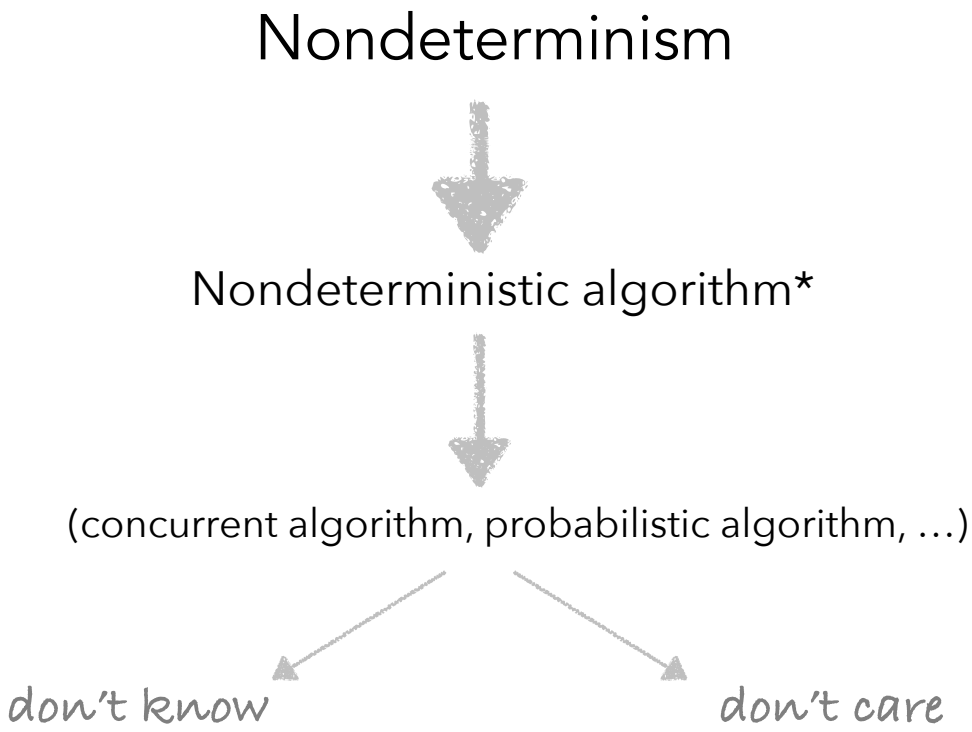
Abstraction principles and best practices

Nondeterminism

Nondeterministic algorithm*

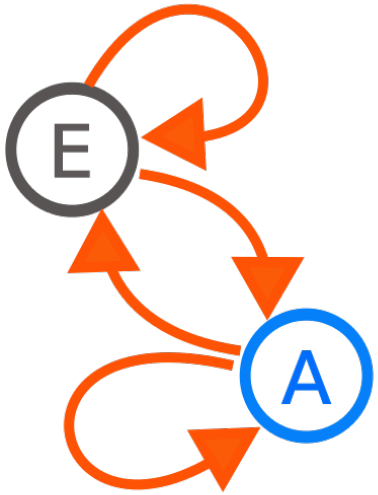
Randomness

Abstraction principles and best practices



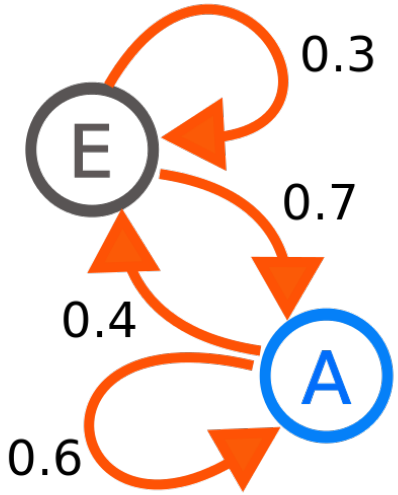
*In computer science, a **nondeterministic algorithm** is an algorithm that, even for the same input, can exhibit different behaviors on different runs, as opposed to a deterministic algorithm.

A simple example



Assume some computational process with two states, A and E, and state transitions as illustrated here. What is the observable behavior of the process?

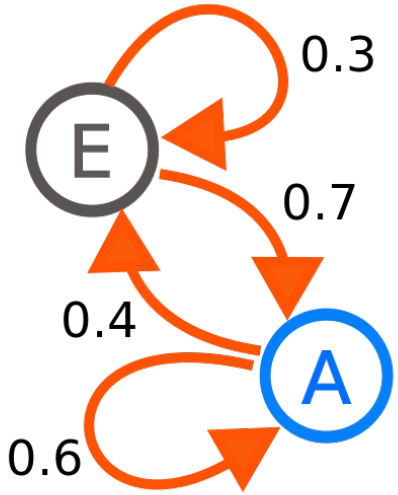
A simple example



Consistent labelling!

Each number represents the probability of the process changing from one state to another state, with the direction indicated by an arrow. For example, if the process is in state A, then the probability it changes to state E is 0.4, while the probability it remains in state A is 0.6.

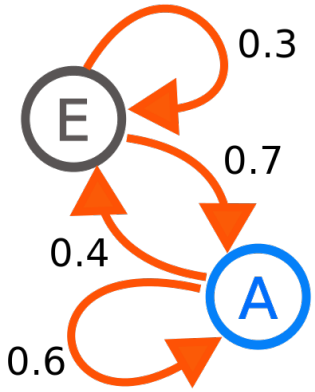
A simple example



Assume in any given run the choice of the next state does only depend on the current state (and the state transition probabilities) but neither on any previous state(s) nor on the history of computation.

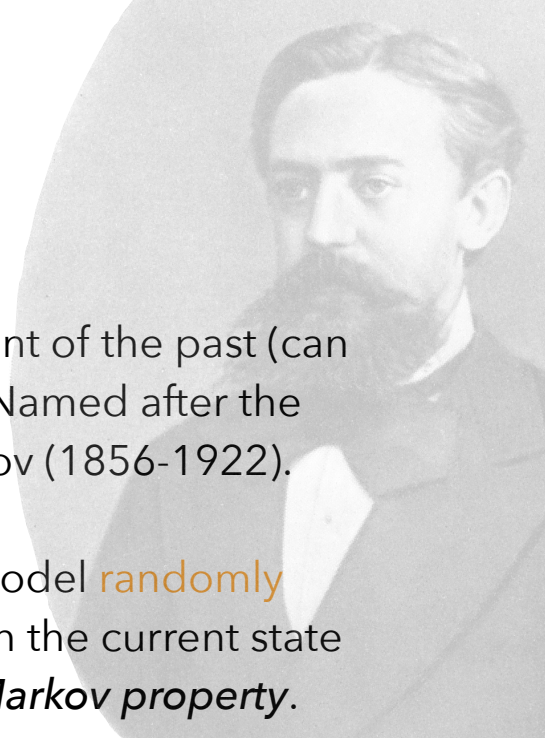
A simple example

With this assumption, the below diagram can be associated with the state transition behavior of a **two-state Markov process**, with the states labelled E and A.



What is missing here ...?

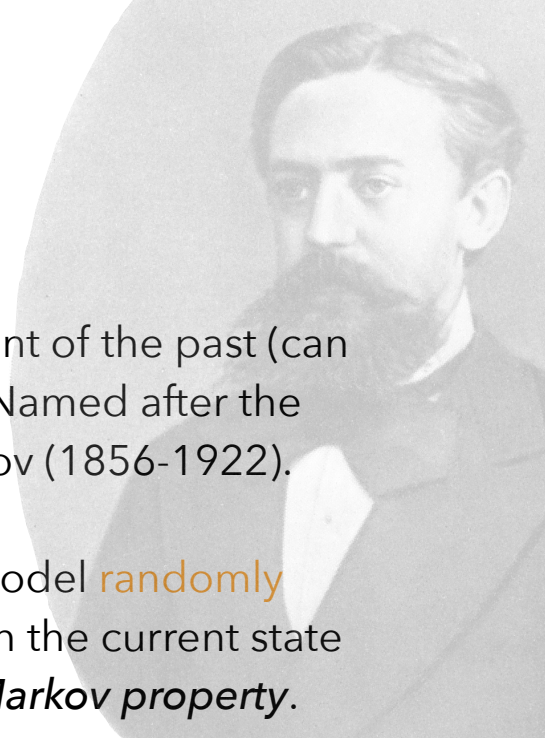
Markov Models



A Markov process is a *random process* in which the future is independent of the past (can not be accurately predicted from its past behavior), given the present. Named after the inventor of Markov analysis, the Russian mathematician Andrei A. Markov (1856-1922).

In probability theory, a Markov model is a ***stochastic model**** used to model randomly changing systems where it is assumed that future states depend only on the current state not on any events that occurred in a prior state, that is, it assumes the ***Markov property***.

**Stochastic* means being or having a random variable. A stochastic model is a tool for estimating probability distributions of potential outcomes by allowing for random variation in one or more inputs over time. The random variation is usually based on fluctuations observed in historical data for a selected period using standard time-series techniques.



Markov Models

A Markov process is a *random process* in which the future is independent of the past (can not be accurately predicted from its past behavior), given the present. Named after the inventor of Markov analysis, the Russian mathematician Andrei A. Markov (1856-1922).

In probability theory, a Markov model is a ***stochastic model**** used to model randomly changing systems where it is assumed that future states depend only on the current state not on any events that occurred in a prior state, that is, it assumes the ***Markov property***.

The term Markov property refers to the *memoryless property* of a stochastic process, that is the conditional probability distribution of future states of the process depends only upon the present state, not on the sequence of events that preceded it.

- Generally, this assumption enables reasoning and computation with Markov models that would otherwise be intractable. For this reason, in the fields of predictive modeling and probabilistic forecasting—a.k.a. **Predictive Analytics**—, it is desirable for a model to exhibit the Markov property.

**Stochastic* means being or having a random variable. A stochastic model is a tool for estimating probability distributions of potential outcomes by allowing for random variation in one or more inputs over time. The random variation is usually based on fluctuations observed in historical data for a selected period using standard time-series techniques.

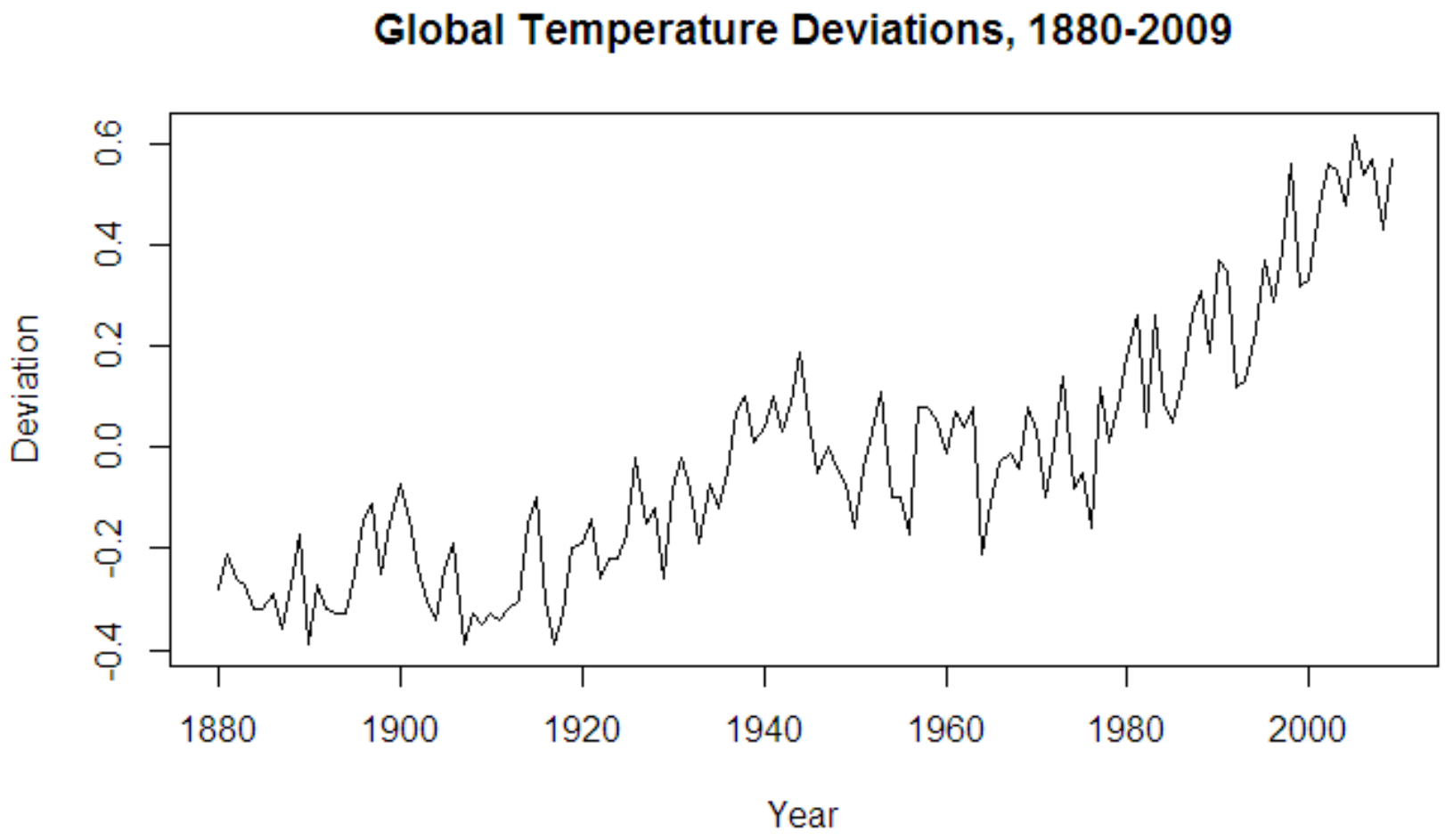
Markov chain

A *discrete-time stochastic process* (in contrast to continuous-time stochastic process) satisfying the Markov property is known as a **Markov chain**.

The term "Markov chain" refers to the sequence of random variables such a process moves through, with the Markov property defining **serial dependence** only between adjacent periods (as in a “chain").

Markov chains can thus be used for describing systems that follow a **chain of linked events**, where what happens next depends only on the current state of the system.

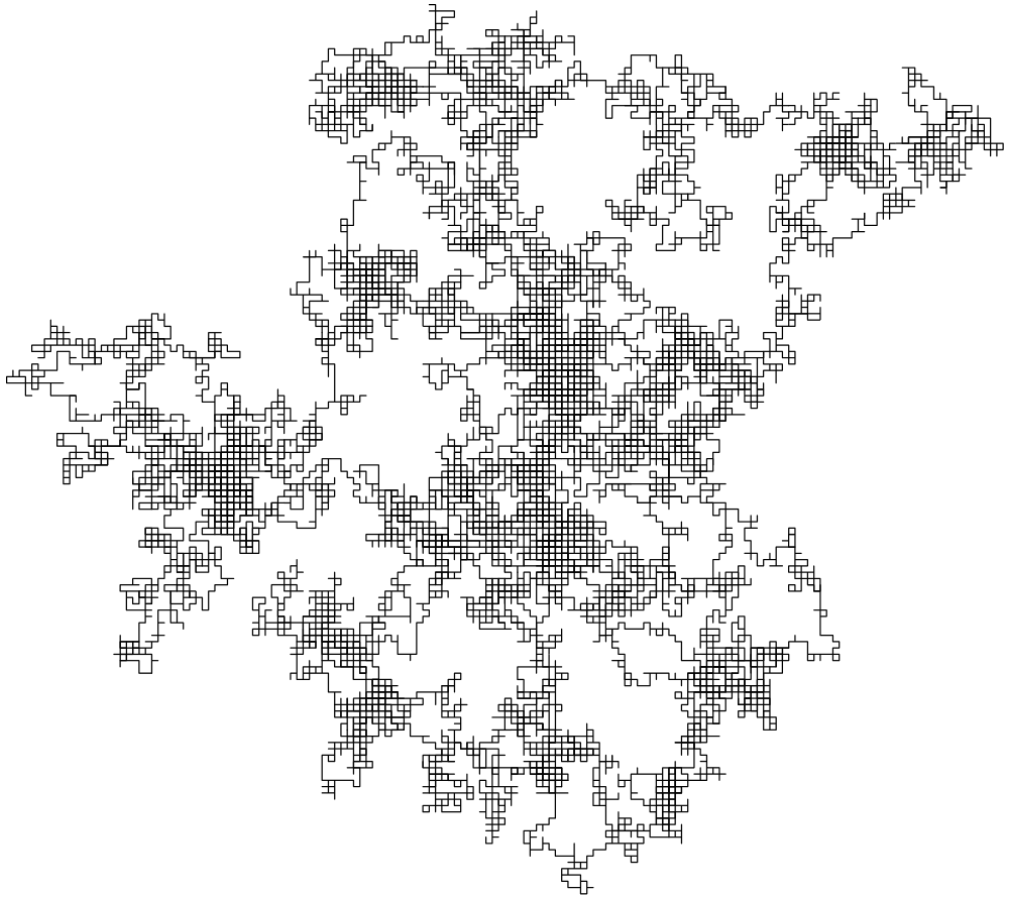
Time Series



What about the Markov property ...?

Example 1

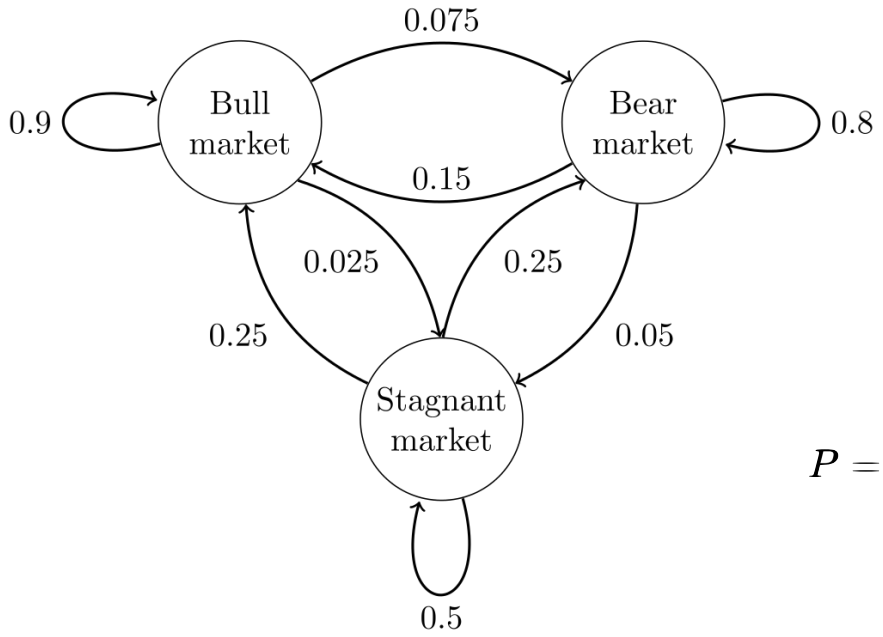
Random walk on a grid where the decision how to continue from a given grid point is independent of any grid points visited prior to the current one.



Random walk in two dimensions with 25 thousand steps. (Source: Wikipedia)

Example 2

A state diagram for a simple example is using a directed graph to picture the state transitions. The states represent whether a **hypothetical stock market** is exhibiting a bull market, bear market, or stagnant market trend during a given week.



$$P = \begin{bmatrix} 0.9 & 0.075 & 0.025 \\ 0.15 & 0.8 & 0.05 \\ 0.25 & 0.25 & 0.5 \end{bmatrix}$$

Explanation: A bull week is followed by another bull week 90% of the time, a bear week 7.5% of the time, and a stagnant week the other 2.5% of the time. Labelling the **state space** {1 = bull, 2 = bear, 3 = stagnant} the **transition matrix** for this example is shown separately.

The **distribution over states** can be written as a stochastic *row vector x* with the relation

$$x^{(n+1)} = x^{(n)}P$$

So if at time n the system is in state $x(n)$, then three time periods later, at time $n + 3$ the distribution is

$$\begin{aligned} x^{(n+3)} &= x^{(n+2)}P = \left(x^{(n+1)}P\right)P \\ &= x^{(n+1)}P^2 = \left(x^{(n)}P\right)P^2 \\ &= x^{(n)}P^3 \end{aligned}$$

Example 3

Weather forecast using Markov chains?

https://www.youtube.com/watch?v=4XqWadvEj2k

Modeling Markov Processes

Source: Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Consider real-world processes producing **observable outputs** that can be abstractly characterized as signals. Distinct characteristic properties result in various types of signals:

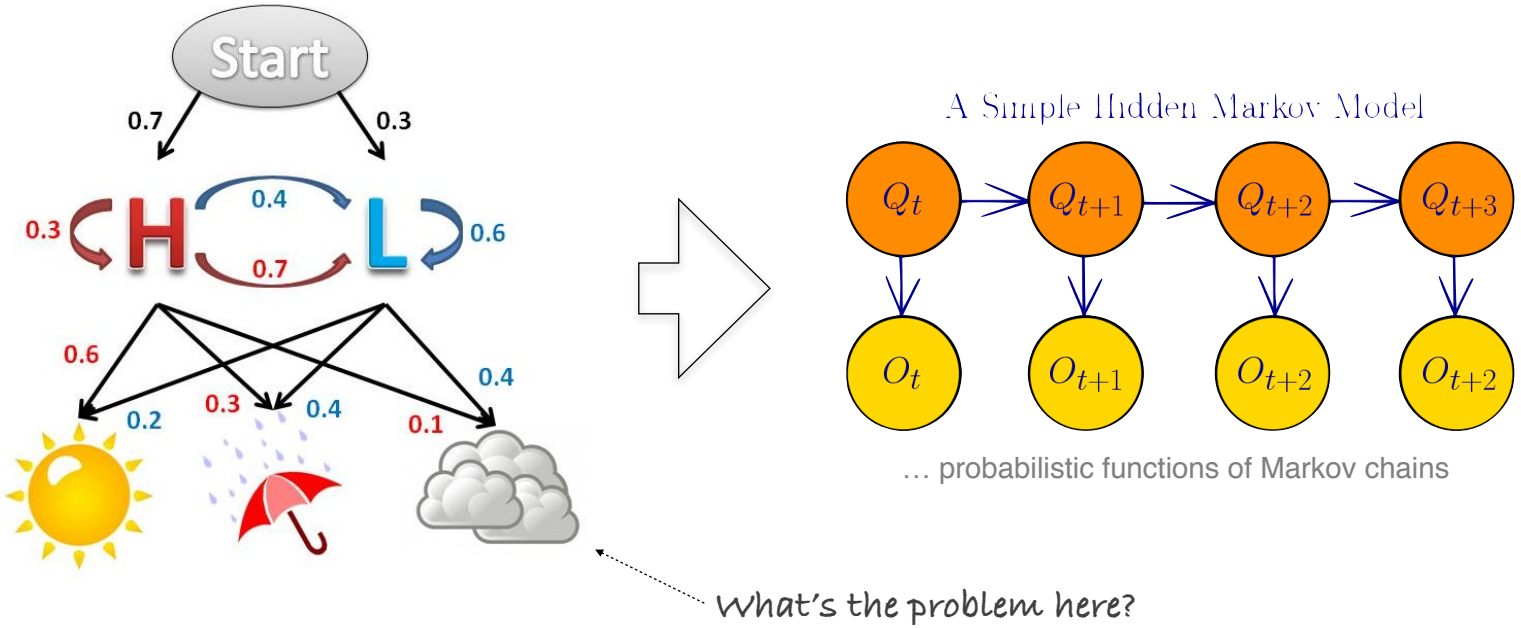
- discrete or continuous (symbol alphabet — temperature measurements)
- stationary or non-stationary signal source (statistical properties vary with time)
- pure or noisy (one source only — additional sources causing distortion, noise etc.)

Modeling Markov Processes

Source: Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Consider real-world processes producing **observable outputs** that can be abstractly characterized as signals. Distinct characteristic properties result in various types of signals:

- discrete or continuous (symbol alphabet — temperature measurements)
- stationary or non-stationary signal source (statistical properties vary with time)
- pure or noisy (one source only — additional sources causing distortion, noise etc.)



Modeling Markov Processes

Source: Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Consider real-world processes producing **observable outputs** that can be abstractly characterized as signals. Distinct characteristic properties result in various types of signals:

- discrete or continuous (symbol alphabet — temperature measurements)
- stationary or non-stationary signal source (statistical properties vary with time)
- pure or noisy (one source only — additional sources causing distortion, noise etc.)

Challenging problem: How can one characterize real signals in terms of signal models?

Rational for using signal models

- provide a basis for a theoretical description of signal processing systems
- enhance understanding of the signal source even if the source is unavailable (through simulation of the real-world process)
- enable building prediction systems, recognition systems, identification systems etc.

Approaches to signal modeling

1. Deterministic models: use known specific properties of the signal (amplitude, frequency etc.)
2. **Statistical models: characterize only the statistical signal properties** (Gaussian, Poissons etc.)

Statistical modeling is based on the assumption that the signal can be well characterized as a *parametric random process*—like a Gaussian process, Poissons process, Markov process etc.—and that the parameters of this stochastic process can be determined (estimated) in a precise, well-defined manner.

⚐ In the following we will focus exclusively on statistical approaches to signal modeling, and restrict the scope to real-world phenomena that can be viewed as Markov processes. Specifically, we use Hidden Markov models as a compact representation of process behavior for the purpose of predicting "normal" behavior in terms of sequences of observations.

Three fundamental problems in Hidden Markov model (HMM) design and analysis:

- **evaluation** of the probability (or likelihood) of a sequence of observations generated by a given HMM
- **determination** of a "best" sequence of model states
- **adjustment** of model parameters so as to best account for the observed signal

Discrete Markov Processes

Consider a **system** with N distinct states, S_1, S_2, …, S_n. At regularly spaced discrete times t, where $t = 1, 2, \ldots$, the system performs a state transition from a given state q_t to the next state q_{t+1} (possibly back to the same state) according to a set of **transition probabilities** associated with each state.

A complete probabilistic description of such a system generally requires specification of the current state at time t, as well as the sequence of predecessor states. For a discrete (first order) **Markov chain**, the general description can be truncated as it requires only the current and the predecessor state:

$$P[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, \cdots] = P[q_t = S_j | q_{t-1} = S_i].$$

For the Markov chain, the right-hand side of the above equation does not depend on time, leading to a set of *state transition probabilities* a_{ij} of the form

$$a_{ij} = P[q_t = S_j | q_{t-1} = S_i], \quad 1 \leq i, j \leq N$$

with the additional **properties** on state transition coefficients: $a_{ij} \geq 0$

$$\sum_{j=1}^{N} a_{ij} = 1$$

Assume the state space of a stochastic process as illustrated below is **observable** and each state identifies the occurrence of a physical event. Then the output generate by the process is a sequence of observable events (states) ordered in time.

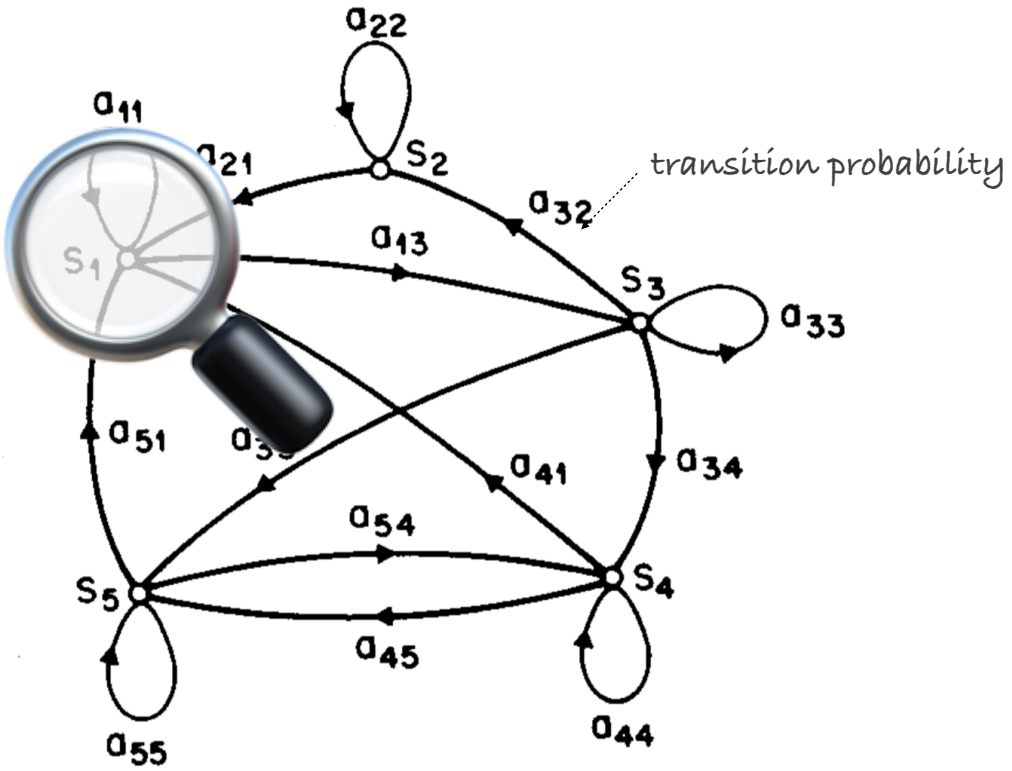


Fig. 1. A Markov chain with 5 states (labeled S_1 to S_5) with selected state transitions.

Consider a 3-state Markov chain of … *whatever* with the state transition probability matrix *A and initial state probabilities* $\pi_i = P[q_1 = S_i]$, $1 \le i \le N$, with $P[q_1 = S_3] = 1$.

$$A = \{a_{ij}\} = \begin{bmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$$

For a given observation sequence O, with $O = \{ S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3\}$, we calculate the probability that O is generated by the model in 8 consecutive steps $t = 1, 2, \ldots, 7, 8$.

3-state Markov chain

$$\begin{aligned} P(O|\text{Model}) &= P[S_3, S_3, S_3, S_1, S_1, S_3, S_2, S_3|\text{Model}] \\ &= P[S_3] \cdot P[S_3|S_3] \cdot P[S_3|S_3] \cdot P[S_1|S_3] \\ &\quad \cdot P[S_1|S_1] \cdot P[S_3|S_1] \cdot P[S_2|S_3] \cdot P[S_3|S_2] \\ &= \pi_3 \cdot a_{33} \cdot a_{33} \cdot a_{31} \cdot a_{11} \cdot a_{13} \cdot a_{32} \cdot a_{23} \\ &= 1 \cdot (0.8)(0.8)(0.1)(0.4)(0.3)(0.1)(0.2) \\ &= 1.536 \times 10^{-4} \end{aligned}$$

Given the model is in a known state, what is the probability that it stays in that state for exactly d steps? For the observation sequence

$$O = \{\underset{1}{S_i}, \underset{2}{S_i}, \underset{3}{S_i}, \cdots, \underset{d}{S_i}, \underset{d+1}{S_j} \neq S_i\},$$

the probability that this sequence occurs can be evaluated as

$$P(O|\text{Model}, q_1 = S_i) = (a_{ii})^{d-1}(1 - a_{ii}) = p_i(d).$$

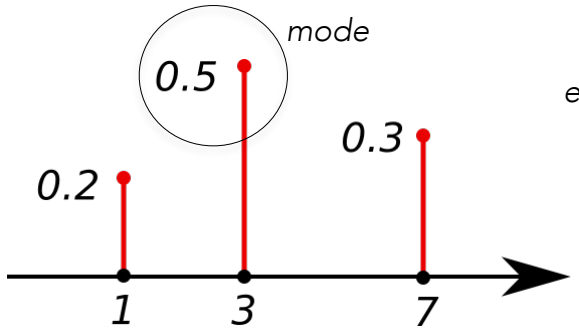
The quantity $p_i(d)$ is the discrete probability density function—a.k.a. ***probability mass function***—of duration d in state i. This exponential duration density is characteristic of the state duration in a Markov chain. Based on $p_i(d)$, we calculate the **expected durations** in a state, conditioned on starting in that state as

$$\begin{aligned}\bar{d}_i &= \sum_{d=1}^{\infty} d p_i(d) \\ &= \sum_{d=1}^{\infty} d(a_{ii})^{d-1}(1 - a_{ii}) = \frac{1}{1 - a_{ii}}.\end{aligned}$$

Probability mass function

- A probability mass function states the probability that a **discrete random variable** is exactly equal to some value.
- The value of the random variable having the largest probability mass is called the ***mode***; it is the value at which its probability mass function takes its maximum value (i.e. the value that is most likely to be sampled).

Graph of a probability mass function for a random variable with sample space {1,3,7}



expected value = 3.8

All the values of this function must be non-negative and sum up to 1.

Source: Wikipedia

Expected value

The expected value of a discrete random variable *X* is the probability-weighted average of all possible values *X* can take on.

Countably infinite case

Let *X* be a random variable with a countable set of finite outcomes x_1, x_2, … occurring with probabilities p_1, p_2, … respectively, such that the infinite sum $\sum_{i=1}^{\infty} |x_i| \, p_i$ converges.

E[*X*], the ***expectation*** (expected value) of *X*, is defined as the series

$$\mathrm{E}[X] = \sum_{i=1}^{\infty} x_i \, p_i .$$

Given the model is in a known state, what is the probability that it stays in that state for exactly d steps? For the observation sequence

$$O = \{\underset{1}{S_i}, \underset{2}{S_i}, \underset{3}{S_i}, \cdots, \underset{d}{S_i}, \underset{d+1}{S_j} \neq S_i\},$$

the probability that this sequence occurs can be evaluated as

$$P(O|\text{Model}, q_1 = S_i) = (a_{ii})^{d-1}(1 - a_{ii}) = p_i(d).$$

The quantity $p_i(d)$ is the discrete probability density function—a.k.a. ***probability mass function***—of duration d in state i. This exponential duration density is characteristic of the state duration in a Markov chain. Based on $p_i(d)$, we calculate the **expected durations** in a state, conditioned on starting in that state as

$$\bar{d}_i = \sum_{d=1}^{\infty} d p_i(d)$$

$$= \sum_{d=1}^{\infty} d(a_{ii})^{d-1}(1 - a_{ii}) = \frac{1}{1 - a_{ii}}.$$

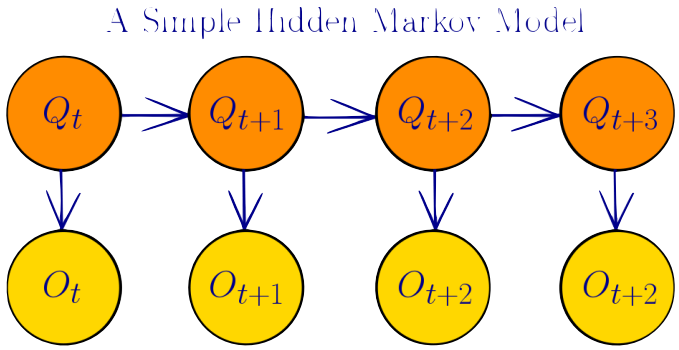
The expected number of consecutive states for S3 is 1 / (1 - 0.8) = 5, for S2 is 2.5, and for S1 is 1.67.

Hidden Markov model

A hidden Markov model (HMM) is a Markov model in which the system being modeled is assumed to be a Markov process with **unobserved** (hidden) states. (Source: Rabiner, 1989)

In simpler Markov models (like a Markov chain) states are *directly observable*, whereas in an HMM the state sequence through which the model passes remains *hidden*.

In an HMM, the observation is a probabilistic function of the state. The resulting model is a **doubly embedded stochastic process** that is not directly observable, but can be observed indirectly through another set of stochastic processes that produce a sequence of observations.



Each state has a probability distribution over the possible *output tokens*. Thus, the sequence of observations (tokens generated by a HMM) gives some information about the underlying sequence of states associated with the observed token sequence.

Hidden Markov models are especially known for applications in **temporal pattern recognition** in signal processing such as

- speech
- handwriting
- gesture recognition
- part-of-speech tagging
- musical score following
- partial discharges
- bioinformatics
- …

the process of automatically listening to a live music performance and tracking the position in the score; an active area of research at the intersection of artificial intelligence, pattern recognition, signal processing, and musicology.

Thus, it is not surprising that HMMs can also be used for detecting anomalous patterns in time series data as will be explained.

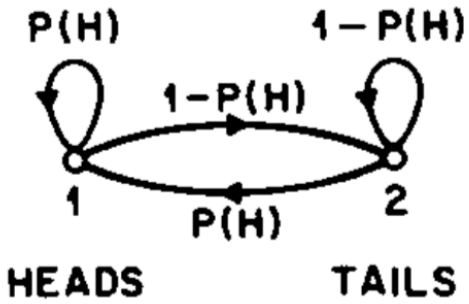
A Tibetan musical score from the 19th century.

Coin Tossing Scenarios

Someone performs a series of coin tosses using one or more coins. We cannot observe the coins but only learn about the outcomes in terms of an observation sequence of heads or tails.

How can we build an HMM that **models** the observed sequence of heads or tails?

- There is more than one way of doing this. Choices we make depend on what the states of the model represent and how many states we intend to use.
- Let's start with a **simple 2-state model** where one state represents head and the other tail. That is, we assume only a single (biased) coin is being tossed. So we only need to decide on the bias, say the probability of heads.

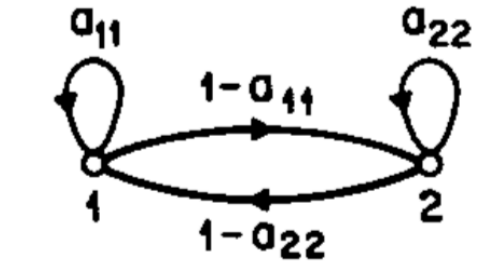


Not really a ***Hidden*** Markov model

O = H H T T H T H H T T H ...
S = 1 1 2 2 1 2 1 1 2 2 1 ...

- Alternatively one could in principle also use a **1-state model** and encode the bias in the probabilistic observability function of the state (resulting in a degenerate model).

- A **2-state model** for representing two different coins with different biases, where each state is characterized by its probability distribution of heads and tails as well as its state transition probabilities:



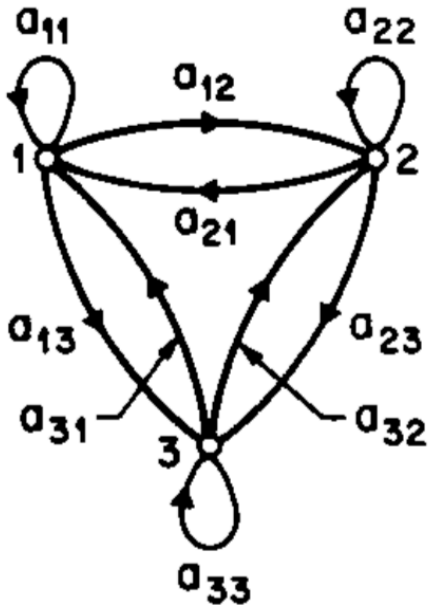
$P(H) = P_1$ $P(H) = P_2$

$P(T) = 1-P_1$ $P(T) = 1-P_2$

O = H H T T H T H H T T H ...

S = 2 1 1 2 2 2 1 2 2 1 2 ...

- Finally, a model for representing three different biased coins:



O = H H T T H T H H T T H ...
S = 3 1 2 3 3 1 1 2 3 1 3 ...

	STATE		
	1	2	3
P(H)	P_1	P_2	P_3
P(T)	$1-P_1$	$1-P_2$	$1-P_3$

Given the choices, which of the three models (excluding the degenerate one) matches best the actual observations of *heads* and *tails*?

Model complexity:

- The 1-coin model has only **one** unknown parameter.
- The 2-coin model has **four** unknown parameters.
- The 3-coin model has **nine** unknown parameters.

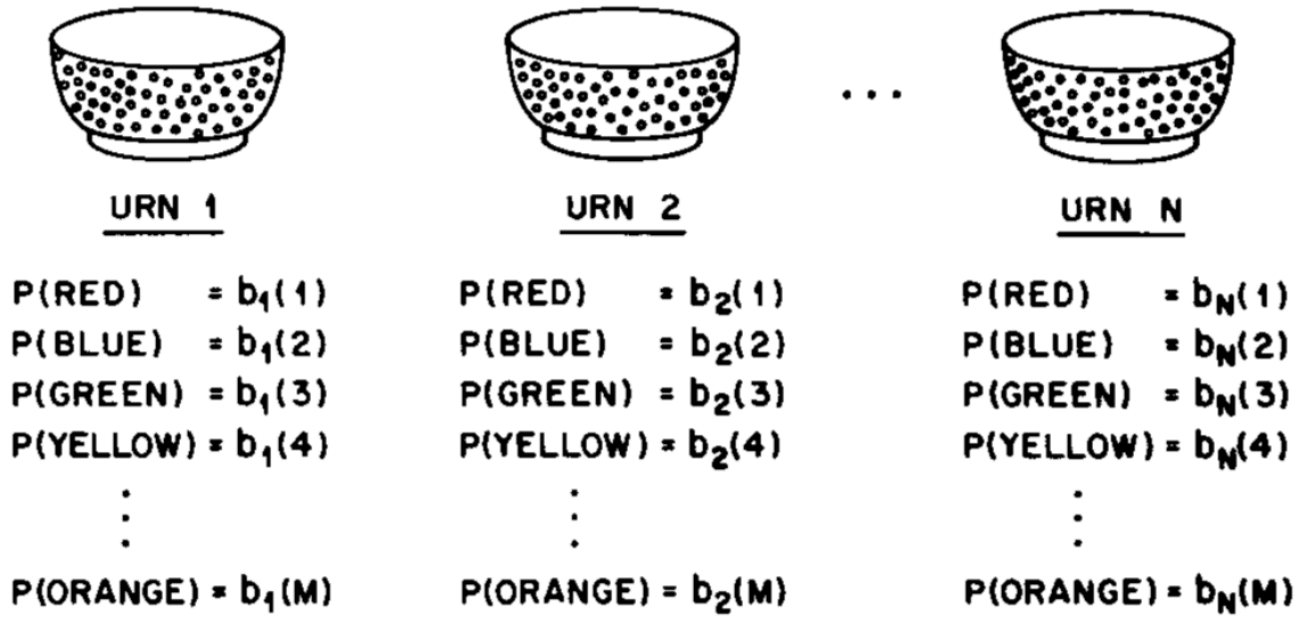
With the extra degrees of freedom, the larger HMM may somehow seem more capable of modeling a series of coin tossing experiments; however, there are additional considerations to be taken into account, which impose limitations on the **size of the model** for practical reasons.

Specifically, if the actual experiments use only a single coin, the 3-state model would be highly inappropriate as the model would not correspond to the actual physical event.

⚐ A **good solution** is a model with the least complexity necessary to reflect the observable behaviour of the Markov process with sufficient accuracy in a direct and intuitive way. Generally, there may be (and usually is) more than one such model.

The Urn and Ball Process

In its discrete form, a hidden Markov process can be visualized as a generalization of the urn problem with replacement (where each item taken from the urn is returned to the original urn before the next step).



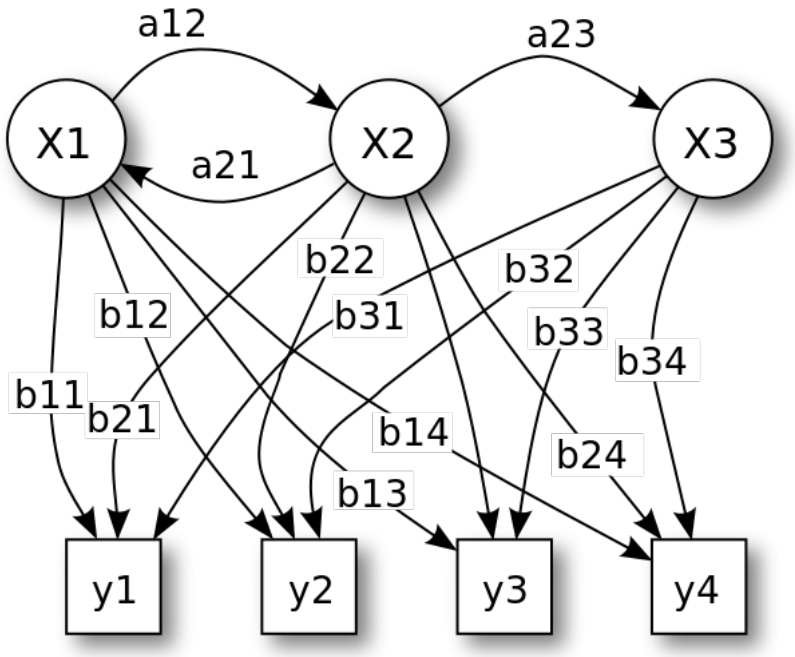
N-state urn and ball model to illustrate the general case of a discrete symbol HMM

Problem definition

- In a room, **not visible to an observer**, there is a *"genie"*.
- Connected to the room is a (moving) *conveyor belt* visible to the observer.
- The room contains urns X1, X2, X3, …, each of which contains (a large number of) balls labeled y1, y2, y3, …, with a **known mix of balls**.
- The genie chooses an urn, using some random selection process (1), and randomly draws a ball from that urn (2).
- It then puts the identical label onto a conveyer belt, where the observer can **observe the sequence of ball labels** but not the sequence of urns from which they were drawn.

Problem definition (cont.)

- The procedure to choose the urn for the *n*-th ball depends only upon the choice of the urn for the (*n* − 1)-*th* ball but does not directly depend on the urns chosen before this single previous urn; therefore, this is called a (first order) Markov process.



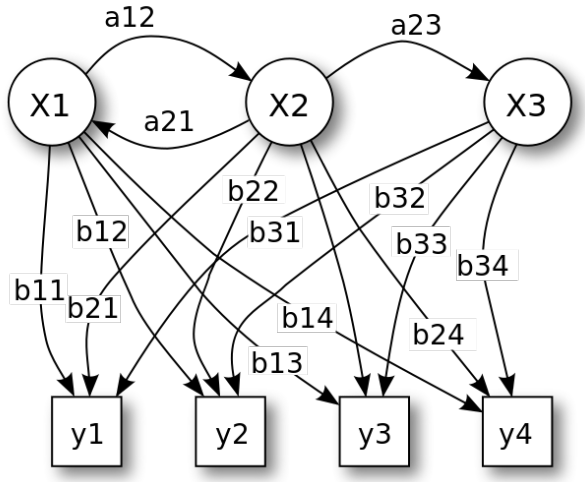
Scenario with 3 urns, 4 different types of balls, where the probability that a certain ball type is drawn from a given urn is illustrated by the labels **bij**.

The Markov process itself cannot be observed, **only the sequence of labeled balls**, thus this arrangement is called a "hidden Markov process".

One can see that balls y1, y2, y3, y4 can be drawn at each state. Even if the observer knows the composition of the urns and has just observed a sequence of three balls, e.g. y1, y2 and y3 on the conveyor belt, the observer still cannot be certain which urn the genie has drawn the third ball from. One can reason though that the starting state was either X1 or X2 ~~and that the third ball was drawn in either X1 or X2~~.

Still, the observer can work out information, such as the **likelihood** that the third ball came from each of the urns.

Probabilistic parameters of a hidden Markov model:
X – states
y – possible observations
a – state transition probabilities
b – output probabilities
…

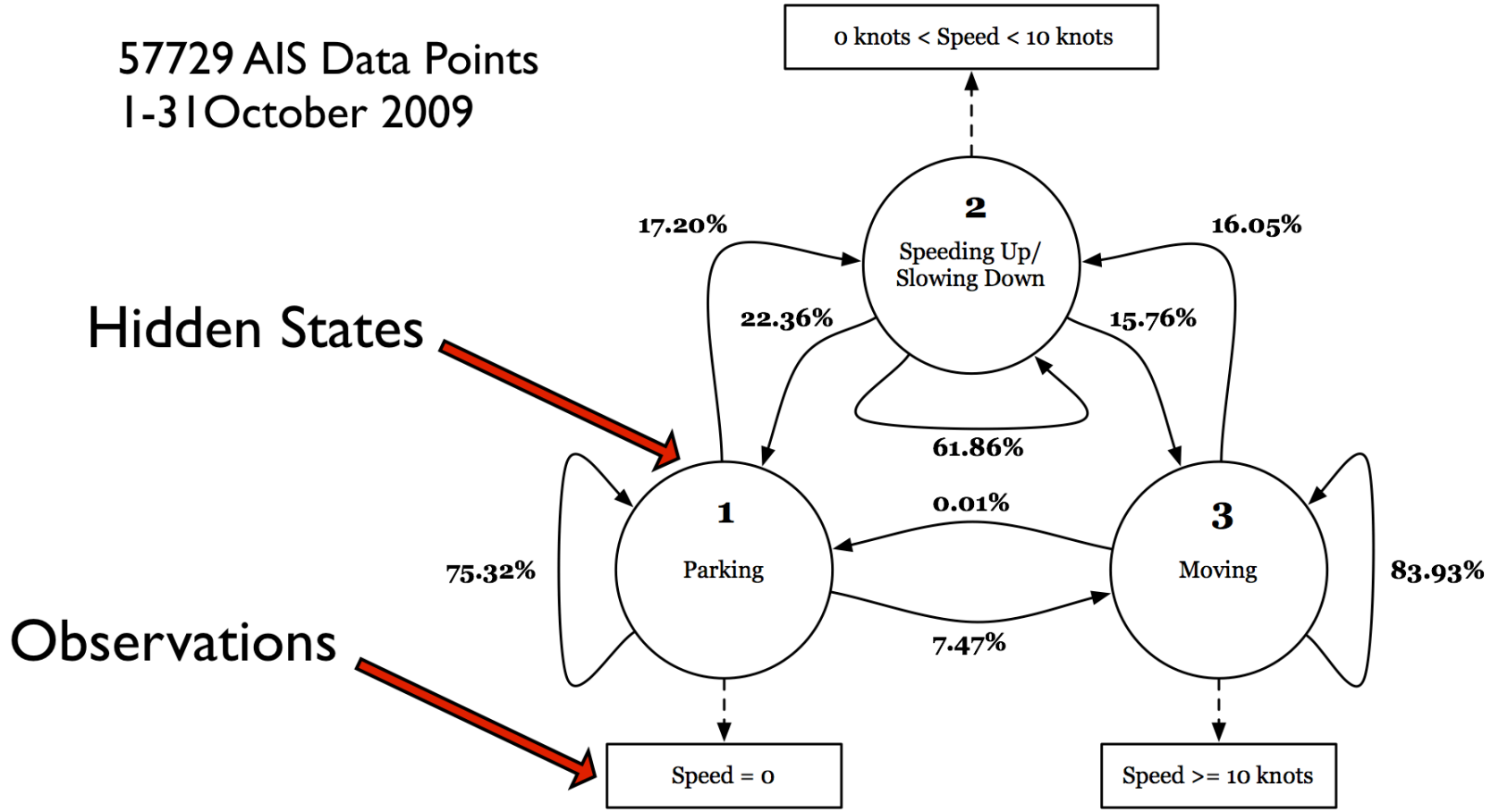


The **simplest HMM** that corresponds to the *urn and ball process* is one in which

- each state corresponds to a specific urn, and for which
- a (ball) label probability is defined for each state;
- the choice of urns is dictated by the state transition matrix of the HMM.

Real-life example

Seabus movement pattern

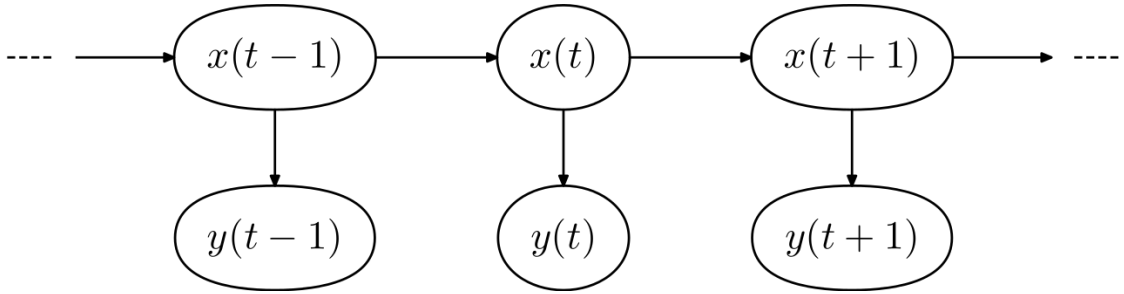


HMM Architecture

The diagram below shows the general architecture of an instantiated HMM:

- each oval represents a random variable that can adopt any of a number of values;
- the random variable *x*(*t*) is the **hidden state at time *t;***
- the random variable *y*(*t*) is the **observation at time t**.

Arrows in the diagram denote ***conditional dependencies***.



HMM Abstract Characterization

In the standard type of hidden Markov model considered here, the **state space** of the hidden variables is discrete, while the **observation values** can either be discrete or continuous. The parameters of a hidden Markov model are of two types: *i) transition probabilities* and *initial state probabilities*, and *ii) output probabilities*. The transition probabilities control the way the hidden state at time t is chosen given the hidden state at time $t - 1$.

- The hidden state space is assumed to consist of one of N **possible values**, modeled as a categorical distribution (a discrete distribution characterized by a probability mass function). This means that for each of the N possible states that a hidden variable at time t can be in, there is a transition probability from this state to each of the N possible states of the hidden variable at time $t + 1$, for a total of N^2 **transition probabilities**.

- Note that the set of transition probabilities for transitions from any given state **must sum to 1**. Thus, the NxN-matrix of transition probabilities is a Markov matrix.

- In addition, for each of the N possible states, there is a set of output probabilities governing the **distribution of the observed variable** at a particular time given the state of the hidden variable at that time. The size of this set depends on the **nature of the observed variable**.

Hidden Markov Model - Definition

An HMM is characterized by the following five aspects (Rabiner, 1989):

1. ***N***, the number of **states** in the model; ***S***, the individual states $S = \{S_1, S_2, \ldots, S_N\}$, and q_t, the state at time t.

2. ***M***, the number of distinct **observation symbols** per state, the size of the discrete alphabet, and the individual symbols as $V = \{v_1, v_2, \ldots, v_M\}$.

   Observation symbols correspond to the physical output of the system being modeled.

3. The state **transition probability distribution** $A = \{ a_{ij} \}$, where

   $$a_{ij} = P\,[q_{t+1} = S_j \mid q_t = S_i]\,,\ 1 \le i,j \le N.$$

   If each state can reach every other state in a single step, we have $aij > 0$ for all i,j.

4. The **observation symbol probability distribution** in state j, $B = \{ b_j(k) \}$, where

   $$b_j(k) = P\,[v_k \text{ at } t \mid q_t = S_j], \quad \text{for } 1 \le j \le N \text{ and } 1 \le k \le M.$$

5. The **initial state distribution** $\pi = \{\pi_i\}$, where $\pi_i = P\,[q_1 = S_i]$, for $1 \le i \le N$.

Given appropriate values for *N, M, A, B*, and π, the HMM can be used as a **generator** for an observation sequence $O = O_1, O_2, \ldots, O_T$, where each observation O_t is a symbol from V and T is the number of observations in the sequence as follows:

1. **Choose** an initial state $q_1 = S_i$

   according to the initial state distribution π;

2. **Set** $t = 1$;

3. **Choose** $O_t = v_k$

   according to the observation symbol probability distribution in state S_i as

   determined by $b_j(k)$;

4. **Transition** to state $q_{t+1} = S_j$

   according to the state transition probability distribution for state S_i as

   determined by $\{ a_{ij} \}$;

5. **Set** $t = t + 1$;

6. **Return** to Step 3, if $t < T$; otherwise, **Terminate**.

Given appropriate values for N, M, A, B, and π, the HMM can be used as a **generator** for an observation sequence $O = O_1, O_2, \ldots, O_T$, where each observation O_t is a symbol from V and T is the number of observations in the sequence as follows:

1. **Choose** an initial state $q_1 = S_i$ according to the initial state distribution π;
2. **Set** $t = 1$;
3. **Choose** $O_t = v_k$ according to the observation symbol probability distribution in state S_i as determined by $b_j(k)$;
4. **Transition** to state $q_{t+1} = S_j$ according to the state transition probability distribution for state S_i as determined by $\{ a_{ij} \}$;
5. **Set** $t = t + 1$;
6. **Return** to Step 3, if $t < T$; otherwise, **Terminate**.

Note: The above procedure can be used as both a **generator of observations**, and as a **model to explain** how a given observation sequence was generated by an HMM.

applies to the course project!

Model Parameters

In short, a complete specification of an HMM requires specification of

- two model parameters (***N***, ***M***),
- specification of observation symbols, and
- the specification of three probability measures ***A***, ***B***, and π.

For convenience, the compact notation λ = (***A***, ***B***, π) is used to indicate the **complete parameter set** of a Hidden Markov model.

Three basic problems

Given the above form of HMM, there are three basic problems of interest that must be solved for a model to be used in real-world applications (Rabiner, 1989):

Problem 1: Given the observation sequence $O = O_1, O_2, \ldots, O_T$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence given the model?

This problem can also be viewed as one of *scoring* how well a given model matches a given observation sequence.

Three basic problems

Given the above form of HMM, there are three basic problems of interest that must be solved for a model to be used in real-world applications (Rabiner, 1989):

Problem 1: Given the observation sequence $O = O_1, O_2, \ldots, O_T$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence given the model?

This problem can also be viewed as one of *scoring* how well a given model matches a given observation sequence.

Problem 2: Given the observation sequence $O = O_1, O_2, \ldots, O_T$, and the model $\lambda = (A, B, \pi)$, how do we choose a corresponding state sequence $Q = q_1\, q_2\, \ldots\, q_T$ that is optimal in some meaningful sense (i.e., best explains the observations)?

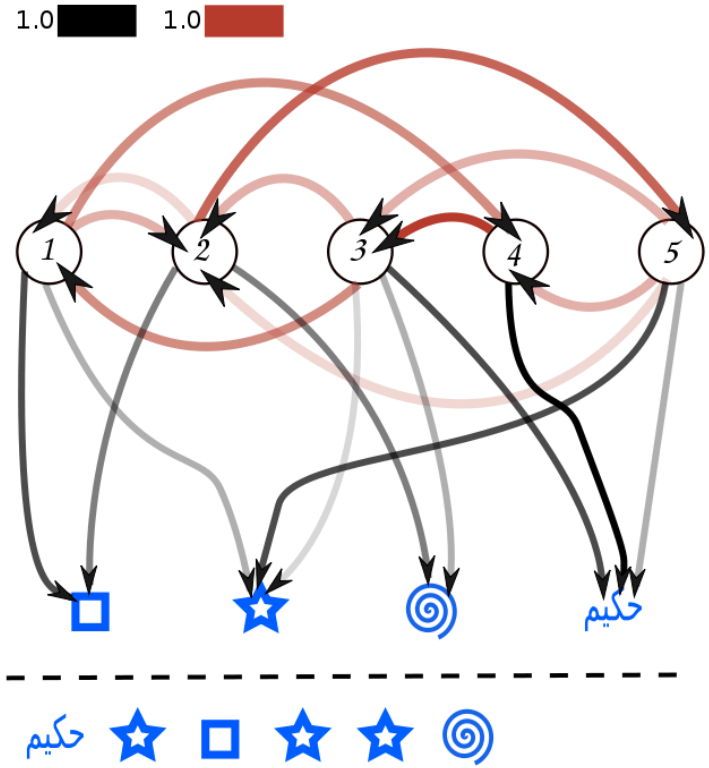
This problem can be viewed as attempting to uncover the hidden part of a model.

Since there is no "correct" state sequence to be found, for practical situations, one usually uses an *optimality criterion* to solve the problem as best as possible. However, there can be several reasonable optimality criteria that can be imposed, depending on the intended use of the uncovered state sequence.

An illustrative example

The state transition and output probabilities of an HMM are indicated by the line opacity in the upper part of the diagram. Given the observed output sequence in the lower part of the diagram, we may be interested in the most likely sequence of states that could have produced it. Based on the arrows that are present in the diagram, the following state sequences are candidates:

5 3 2 5 3 2
4 3 2 5 3 2
3 1 2 5 3 2



Source: Wikipedia

We can find the most likely sequence by evaluating the joint probability of both the state sequence and the observations for each case (simply by multiplying the probability values, which here correspond to the opacities of the arrows involved).

Problem 3: Given the observation sequence $O = O_1, O_2, \ldots, O_T$, and a model $\lambda = (A, B, \pi)$, **how do we adjust the model parameters** A, B, and π to maximize $P(O|\lambda)$?

One attempts to optimize the model parameters so as to best describe how a given observation sequence comes about.

- The observation sequence used to adjust the model parameters is called **a training sequence** since it is used to "train" the HMM.
- The training problem is the crucial one for most applications of HMMs, since it allows us to fit (optimally adapt) the model to the observed training data, that is, creating the best model for a real phenomenon of interest.

Solutions to the Problems

Given a model $\lambda = (A, B, \pi)$ and an observation sequence $O = O_1, O_2, \ldots, O_T$, how can one **compute** the probability $P(O|\lambda)$ – the probability of O given the model.

The most straightforward (brute force) way of computing $P(O|\lambda)$ is by enumerating every possible state sequence of length T – this could take a while though.

Consider one such state sequence $Q = q_1, q_2, \ldots, q_T$, where q_1 is the initial state. The probability of O for the state sequence Q is

$$P(O|Q, \lambda) = \prod_{t=1}^{T} P(O_t|q_t, \lambda)$$

Solutions to the Problems

Given a model $\lambda = (A, B, \pi)$ and an observation sequence $O = O_1, O_2, \ldots, O_T$, how can one **compute** the probability $P(O|\lambda)$ – the probability of O given the model.

The most straightforward (brute force) way of computing $P(O|\lambda)$ is by enumerating every possible state sequence of length T – this could take a while though.

Consider one such state sequence $Q = q_1, q_2, \ldots, q_T$, where q_1 is the initial state. The probability of O for the state sequence Q is

$$P(O|Q, \lambda) = \prod_{t=1}^{T} P(O_t|q_t, \lambda)$$

Assuming statistical independence of observations, we get

$$P(O|Q, \lambda) = b_{q_1}(O_1) \cdot b_{q_2}(O_2) \cdots b_{q_T}(O_T).$$

We also know that the probability of the state sequence Q can be written as

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \cdots a_{q_{T-1} q_T}.$$

Now, the joint probability of O and Q (both events occurring simultaneously) is the product off the above two terms,

$$P(O, Q|\lambda) = P(O|Q, \lambda) \cdot P(Q| \lambda).$$

The probability of O (given the model) is obtained by summing up this joint probability over all possible state sequences Q, that is

$$\begin{aligned} P(O|\lambda) &= \sum_{\text{all } Q} P(O|Q, \lambda) \cdot P(Q|\lambda) \\ &= \sum_{q_1, q_2, \cdots, q_T} \pi_{q_1} b_{q_1}(O_1)\, a_{q_1 q_2} b_{q_2}(O_2) \\ &\quad \cdots a_{q_{T-1} q_T} b_{q_T}(Q_T). \end{aligned}$$

Computational Complexity

The calculation of $P(O|\lambda)$ involves on the order of $2T \cdot N^T$ calculations, since at every $t = 1,2, \ldots, T$, there are N possible states that can be reached (hence, there are N^T possible state sequences), and for each such state sequence about $2T$ calculations are required for each term in the sum of the above equation for $P(O|\lambda)$.

This complexity means that the calculation is computationally infeasible, even for small values of N and T; for instance, for $N = 5$ (states) and $T = 100$ (observations), there are on the order of $2 \cdot 100 \cdot 5^{100} \approx 10^{72}$ computations.

A more efficient procedure is required to solve Problem 1.

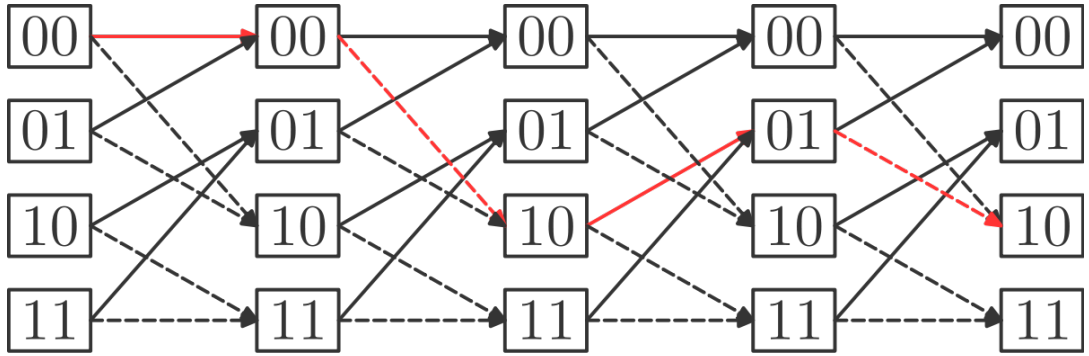
Applying the principle of *dynamic programming*, this problem and related problems can be handled efficiently using dedicated algorithms such as the Forward-Backward algorithm or the Viberti algorithm.

Efficient Solutions to the three HMM problems

Problem 1: Likelihood computation

The **Forward algorithm** is an algorithm with $O(N^2T)$ complexity that

- uses a table to store intermediate values as it builds up the probability of the observation sequence;
- computes the observation probability by summing over the probabilities of all possible hidden state paths that could generate the observation sequence, but it does so
- efficiently by implicitly **folding** each of these paths into a single forward trellis[1].



[1] A trellis is a graph whose nodes are ordered into vertical slices (time), and with each node at each time connected to at least one node at an earlier and at least one node at a later time. The earliest and latest times in the trellis have only one node.

Problem 2: Decoding

- For any model (such as an HMM) that contains hidden variables, the task of determining which sequence of variables is ''the underlying source'' of the sequence of observations is called the decoding task. The most common decoding algorithm for HMMs is the **Viterbi algorithm**.
- This algorithm finds the most likely sequence of hidden states—called the *Viterbi path*—that results in a sequence of observed events, especially in the context of Markov information sources and hidden Markov models.
- Like the Forward algorithm, Viterbi is a dynamic programming algorithms that uses dynamic programming trellis.

Problem 3: HMM Training (find the unknown parameters)

The **Forward-Backward** algorithm is an inference algorithm for hidden Markov models which computes the *posterior marginals* of all hidden state variables given a sequence of observations $O_{1:T} = O_1,\ldots,O_T$. That is, the algorithm computes, for all hidden state variables $X_t \in \{X_1, \ldots, X_T\}$, the distribution $P(X_t \mid O_{1:T})$.

The algorithm makes use of dynamic programming to efficiently compute the values that are required to obtain these distributions in two passes. The first pass goes forward in time while the second goes backward in time; hence the name forward–backward algorithm.

- In the **first pass**, the forward–backward algorithm computes a set of forward probabilities which provide, for all $t \in \{1, \ldots, T\}$, the probability of ending up in any particular state given the first t observations in the sequence, i.e. $P(X_t \mid O_{1:t})$.
- In the **second pass**, the algorithm computes a set of backward probabilities which provide the probability of observing the remaining observations given any starting point t, i.e. $P(O_{t+1:T} \mid X_t)$.
- These two sets of probability distributions can then be combined to obtain the distribution over states at any specific point in time given the entire observation sequence.

Overfitting

Example of Overfitting

Let's say we want to predict if a student will land a job interview based on her resume.

Now, assume we train a model from a dataset of 10,000 **resumes and their outcomes**.

Next, we try the model out on the **original dataset**, and it predicts outcomes with 99% accuracy… wow! Really?

…

Overfitting

Example of Overfitting

Let's say we want to predict if a student will land a job interview based on her resume.

Now, assume we train a model from a dataset of 10,000 resumes and their outcomes.

Next, we try the model out on the **original dataset**, and it predicts outcomes with 99% accuracy… wow! Really?

But now comes the bad news.

When we run the model on a new ("unseen") dataset of resumes, we only get about 50% accuracy… uh-oh!

Our model doesn't generalize well from our training data to unseen data.

This is known as **overfitting**, a common problem in machine learning and data science.

Hidden Semi-Markov model

A hidden semi-Markov model (HSMM) is a statistical model with the same structure as a hidden Markov model, except that the unobservable process is semi-Markov rather than Markov. This means that the probability of there being a change in the hidden state depends on the amount of time that has elapsed since entry into the current state.

This is in contrast to hidden Markov models where there is a constant probability of changing a state irrespective of the time spent in a state.

Statistical inference for hidden semi-Markov models is more difficult than in HMMs, since algorithms like the Baum-Welch algorithm are not directly applicable, and must be adapted requiring more resources.

Sources

Feller, 1968
William Feller. An Introduction to Probability Theory and Its Applications.
John Wiley & Sons, 1968.

Rabiner, 1989
Lawrence R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition.
Proceedings of the IEEE, 77(2):257–286, 1989.