



WIKIPEDIA
The Free Encyclopedia

[Main page](#)

[Contents](#)

[Featured content](#)

[Current events](#)

[Random article](#)

[Donate to Wikipedia](#)

[Wikipedia store](#)

Interaction

[Help](#)

[About Wikipedia](#)

[Community portal](#)

[Recent changes](#)

[Contact page](#)

Tools

[What links here](#)

[Related changes](#)

[Upload file](#)

[Special pages](#)

[Permanent link](#)

[Page information](#)

[Wikidata item](#)

[Cite this page](#)

Print/export

[Create a book](#)

[Download as PDF](#)

[Printable version](#)

Languages

[Afrikaans](#)

[Български](#)

[Català](#)

[Čeština](#)

[Deutsch](#)

[Español](#)

[فارسی](#)

[Français](#)

[한국어](#)

[Bahasa Indonesia](#)

[Italiano](#)

[עברית](#)

[Nederlands](#)

[日本語](#)

[Português](#)

[Русский](#)

[Slovenščina](#)

[Српски / srpski](#)

[Srpskohrvatski / српскохрватски](#)

[Suomi](#)

[Svenska](#)

[ไทย](#)

[Українська](#)

[Tiếng Việt](#)

[中文](#)

[Edit links](#)

Article [Talk](#)

[Read](#) [Edit](#) [View history](#)

Search

Hidden Markov model

From Wikipedia, the free encyclopedia

A **hidden Markov model** (**HMM**) is a [statistical Markov model](#) in which the system being modeled is assumed to be a [Markov process](#) with unobserved (*hidden*) states. A HMM can be presented as the simplest [dynamic Bayesian network](#). The mathematics behind the HMM was developed by [L. E. Baum](#) and coworkers.^{[1][2][3][4][5]} It is closely related to an earlier work on the optimal nonlinear [filtering problem](#) by [Ruslan L. Stratonovich](#),^[6] who was the first to describe the [forward-backward procedure](#).

In simpler [Markov models](#) (like a [Markov chain](#)), the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a *hidden* Markov model, the state is not directly visible, but output, dependent on the state, is visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. Note that the adjective 'hidden' refers to the state sequence through which the model passes, not to the parameters of the model; the model is still referred to as a 'hidden' Markov model even if these parameters are known exactly.

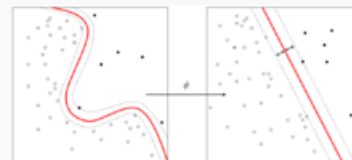
Hidden Markov models are especially known for their application in [temporal](#) pattern recognition such as [speech](#), [handwriting](#), [gesture recognition](#),^[7] [part-of-speech tagging](#), musical score following,^[8] [partial discharges](#)^[9] and [bioinformatics](#).

A hidden Markov model can be considered a generalization of a [mixture model](#) where the hidden variables (or [latent variables](#)), which control the mixture component to be selected for each observation, are related through a Markov process rather than independent of each other. Recently, hidden Markov models have been generalized to pairwise Markov models and triplet Markov models which allow consideration of more complex data structures ^{[10][11]} and the modelling of nonstationary data.^{[12][13]}

Contents

- Description in terms of urns
- Architecture
- Inference
 - Probability of an observed sequence
 - Probability of the latent variables
 - Filtering
 - Smoothing
 - Most likely explanation
 - Statistical significance
- A concrete example
- Learning
- Mathematical description
 - General description
 - Compared with a simple mixture model
 - Examples
 - A two-level Bayesian HMM
- Applications
- History

Machine learning and data mining



Problems

[Classification](#) · [Clustering](#) · [Regression](#) · [Anomaly detection](#) · [Association rules](#) · [Reinforcement learning](#) · [Structured prediction](#) · [Feature learning](#) · [Online learning](#) · [Semi-supervised learning](#) · [Unsupervised learning](#) · [Learning to rank](#) · [Grammar induction](#)

Supervised learning

([classification](#) · [regression](#))

[Decision trees](#) · [Ensembles](#) (Bagging, Boosting, Random forest) · [k-NN](#) · [Linear regression](#) · [Naive Bayes](#) · [Neural networks](#) · [Logistic regression](#) · [Perceptron](#) · [Support vector machine \(SVM\)](#) · [Relevance vector machine \(RVM\)](#)

Clustering

[BIRCH](#) · [Hierarchical](#) · [k-means](#) · [Expectation-maximization \(EM\)](#) · [DBSCAN](#) · [OPTICS](#) · [Mean-shift](#)

Dimensionality reduction

[Factor analysis](#) · [CCA](#) · [ICA](#) · [LDA](#) · [NMF](#) · [PCA](#) · [t-SNE](#)

Structured prediction

[Graphical models](#) (Bayes net, CRF, **HMM**)

Anomaly detection

[k-NN](#) · [Local outlier factor](#)

Neural nets

[Autoencoder](#) · [Deep learning](#) · [Multilayer perceptron](#) · [RNN](#) · [Restricted Boltzmann machine](#) · [SOM](#) · [Convolutional neural network](#)

Theory

[Bias-variance dilemma](#) · [Computational learning theory](#) · [Empirical risk minimization](#) · [PAC learning](#) · [Statistical learning](#) · [VC theory](#)



[Machine learning portal](#)



[Computer science portal](#)



[Statistics portal](#)

v · t · e

- 9 Types
- 10 Extensions
- 11 See also
- 12 References
- 13 External links
 - 13.1 Concepts
 - 13.2 Software

Description in terms of urns [\[edit\]](#)

In its discrete form, a hidden Markov process can be visualized as a generalization of the [Urn problem](#) with replacement (where each item from the urn is returned to the original urn before the next step).^[14] Consider this example: in a room that is not visible to an observer there is a genie. The room contains urns X_1, X_2, X_3, \dots each of which contains a known mix of balls, each ball labeled y_1, y_2, y_3, \dots . The genie chooses an urn in that room and randomly draws a ball from that urn. It then puts the ball onto a conveyor belt, where the observer can observe the sequence of the balls but not the sequence of urns from which they were drawn. The genie has some procedure to choose urns; the choice of the urn for the n -th ball depends only upon a random number and the choice of the urn for the $(n - 1)$ -th ball. The choice of urn does not directly depend on the urns chosen before this single previous urn; therefore, this is called a [Markov process](#). It can be described by the upper part of Figure 1.

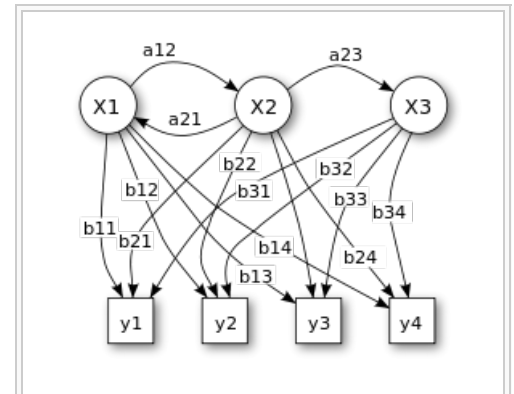


Figure 1. Probabilistic parameters of a hidden Markov model (example)
 X — states
 y — possible observations
 a — state transition probabilities
 b — output probabilities

The Markov process itself cannot be observed, and only the sequence of labeled balls can be observed, thus this arrangement is called a "hidden Markov process". This is illustrated by the lower part of the diagram shown in Figure 1, where one can see that balls y_1, y_2, y_3, y_4 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. y_1, y_2 and y_3 on the conveyor belt, the observer still cannot be *sure* which urn (*i.e.*, at which state) the genie has drawn the third ball from. However, the observer can work out other information, such as the likelihood that the third ball came from each of the urns.

Architecture [\[edit\]](#)

The diagram below shows the general architecture of an instantiated HMM. Each oval shape 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 (with the model from the above diagram, $x(t) \in \{x_1, x_2, x_3\}$). The random variable $y(t)$ is the observation at time t (with $y(t) \in \{y_1, y_2, y_3, y_4\}$). The arrows in the diagram (often called a [trellis diagram](#)) denote conditional dependencies.

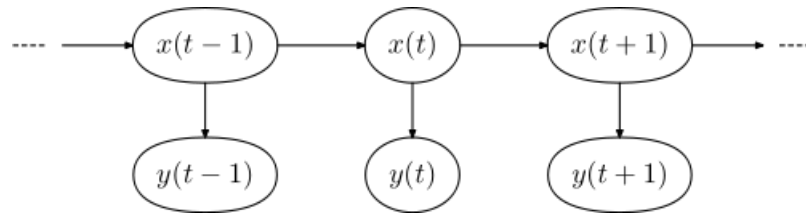
From the diagram, it is clear that the [conditional probability distribution](#) of the hidden variable $x(t)$ at time t , given the values of the hidden variable x at all times, depends *only* on the value of the hidden variable $x(t - 1)$: the values at time $t - 2$ and before have no influence. This is called the [Markov property](#). Similarly, the value of the observed variable $y(t)$ only depends on the value of the hidden variable $x(t)$ (both at time t).

In the standard type of hidden Markov model considered here, the state space of the hidden variables is discrete, while the observations themselves can either be discrete (typically generated from a [categorical distribution](#)) or continuous (typically from a [Gaussian distribution](#)). The parameters of a hidden Markov model are of two types, *transition probabilities* and *emission probabilities* (also known as *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. (See the section below on extensions for other possibilities.) 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 $N \times N$ matrix of transition probabilities is a [Markov matrix](#). Because any one transition probability can be determined

once the others are known, there are a total of $N(N - 1)$ transition parameters.

In addition, for each of the N possible states, there is a set of emission 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. For example, if the observed variable is discrete with M possible values, governed by a [categorical distribution](#), there will be $M - 1$ separate parameters, for a total of $N(M - 1)$ emission parameters over all hidden states. On the other hand, if the observed variable is an M -dimensional vector distributed according to an arbitrary [multivariate Gaussian distribution](#), there will be M parameters controlling the [means](#) and $M(M + 1)/2$ parameters controlling the [covariance matrix](#), for a total of $N(M + \frac{M(M + 1)}{2}) = NM(M + 3)/2 = O(NM^2)$ emission parameters. (In such a case, unless the value of M is small, it may be more practical to restrict the nature of the covariances between individual elements of the observation vector, e.g. by assuming that the elements are independent of each other, or less restrictively, are independent of all but a fixed number of adjacent elements.)



Inference [\[edit\]](#)

Several [inference](#) problems are associated with hidden Markov models, as outlined below.

Probability of an observed sequence [\[edit\]](#)

The task is to compute in a best way, given the parameters of the model, the probability of a particular output sequence. This requires summation over all possible state sequences:

The probability of observing a sequence

$$Y = y(0), y(1), \dots, y(L - 1)$$

of length L is given by

$$P(Y) = \sum_X P(Y | X)P(X),$$

where the sum runs over all possible hidden-node sequences

$$X = x(0), x(1), \dots, x(L - 1).$$

Applying the principle of [dynamic programming](#), this problem, too, can be handled efficiently using the [forward algorithm](#).

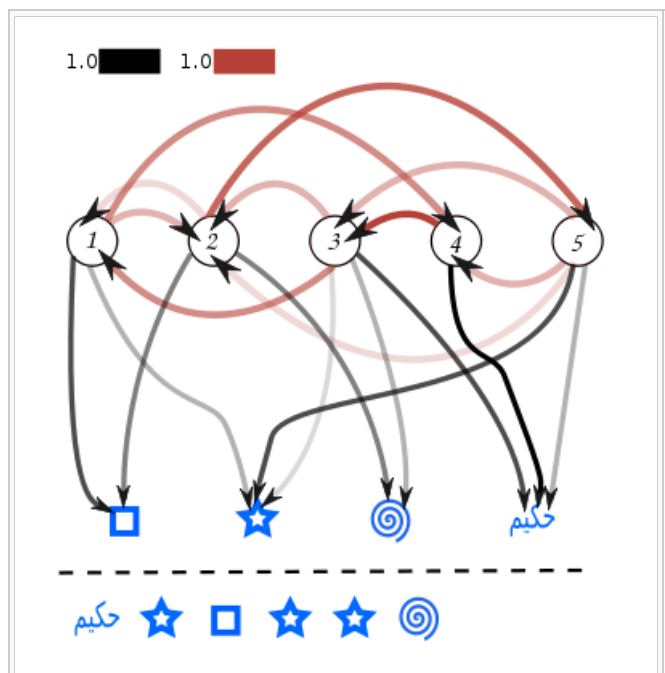
Probability of the latent variables [\[edit\]](#)

A number of related tasks ask about the probability of one or more of the latent variables, given the model's parameters and a sequence of observations

$$y(1), \dots, y(t).$$

Filtering [\[edit\]](#)

The task is to compute, given the model's parameters and a sequence of observations, the distribution over



The state transition and output probabilities of an HMM are indicated by the line opacity in the upper part of the diagram. Given that we have observed the 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

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). In general, this type of problem (i.e. finding the most likely explanation for an observation sequence) can be solved efficiently using the [Viterbi algorithm](#).

hidden states of the last latent variable at the end of the sequence, i.e. to compute $P(x(t) \mid y(1), \dots, y(t))$. This task is normally used when the sequence of latent variables is thought of as the underlying states that a process moves through at a sequence of points of time, with corresponding observations at each point in time. Then, it is natural to ask about the state of the process at the end.

This problem can be handled efficiently using the [forward algorithm](#).

Smoothing [\[edit\]](#)

This is similar to filtering but asks about the distribution of a latent variable somewhere in the middle of a sequence, i.e. to compute $P(x(k) \mid y(1), \dots, y(t))$ for some $k < t$. From the perspective described above, this can be thought of as the probability distribution over hidden states for a point in time k in the past, relative to time t .

The [forward-backward algorithm](#) is an efficient method for computing the smoothed values for all hidden state variables.

Most likely explanation [\[edit\]](#)

The task, unlike the previous two, asks about the [joint probability](#) of the *entire* sequence of hidden states that generated a particular sequence of observations (see illustration on the right). This task is generally applicable when HMM's are applied to different sorts of problems from those for which the tasks of filtering and smoothing are applicable. An example is [part-of-speech tagging](#), where the hidden states represent the underlying [parts of speech](#) corresponding to an observed sequence of words. In this case, what is of interest is the entire sequence of parts of speech, rather than simply the part of speech for a single word, as filtering or smoothing would compute.

This task requires finding a maximum over all possible state sequences, and can be solved efficiently by the [Viterbi algorithm](#).

Statistical significance [\[edit\]](#)

For some of the above problems, it may also be interesting to ask about [statistical significance](#). What is the probability that a sequence drawn from some [null distribution](#) will have an HMM probability (in the case of the forward algorithm) or a maximum state sequence probability (in the case of the Viterbi algorithm) at least as large as that of a particular output sequence?^[15] When an HMM is used to evaluate the relevance of a hypothesis for a particular output sequence, the statistical significance indicates the [false positive rate](#) associated with failing to reject the hypothesis for the output sequence.

A concrete example [\[edit\]](#)

Consider two friends, Alice and Bob, who live far apart from each other and who talk together daily over the telephone about what they did that day. Bob is only interested in three activities: walking in the park, shopping, and cleaning his apartment. The choice of what to do is determined exclusively by the weather on a given day. Alice has no definite information about the weather where Bob lives, but she knows general trends. Based on what Bob tells her he did each day, Alice tries to guess what the weather must have been like.

Alice believes that the weather operates as a discrete [Markov chain](#). There are two states, "Rainy" and "Sunny", but she cannot observe them directly, that is, they are *hidden* from her. On each day, there is a certain chance that Bob will perform one of the following activities, depending on the weather: "walk", "shop", or "clean". Since Bob tells Alice about his activities, those are the *observations*. The entire system is that of a hidden Markov model (HMM).

Alice knows the general weather trends in the area, and what Bob likes to do on average. In other words, the parameters of the HMM are known. They can be represented as follows in the [Python](#):

```
states = ('Rainy', 'Sunny')

observations = ('walk', 'shop', 'clean')

start_probability = {'Rainy': 0.6, 'Sunny': 0.4}

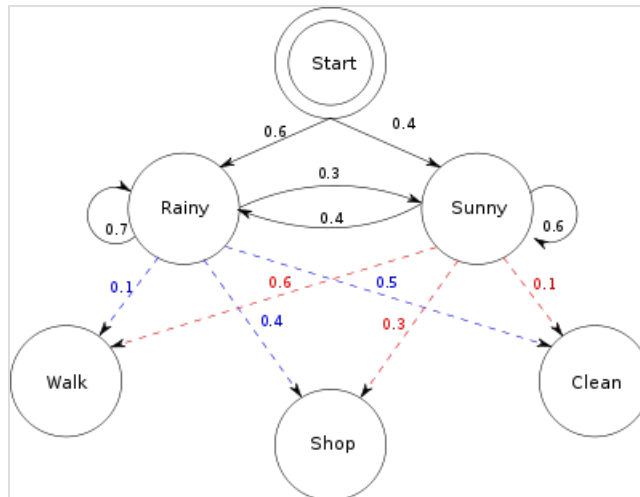
transition_probability = {
    'Rainy' : {'Rainy': 0.7, 'Sunny': 0.3},
    'Sunny' : {'Rainy': 0.4, 'Sunny': 0.6},
}
```

```

emission_probability = {
    'Rainy' : {'walk': 0.1, 'shop': 0.4, 'clean': 0.5},
    'Sunny' : {'walk': 0.6, 'shop': 0.3, 'clean': 0.1},
}

```

In this piece of code, `start_probability` represents Alice's belief about which state the HMM is in when Bob first calls her (all she knows is that it tends to be rainy on average). The particular probability distribution used here is not the equilibrium one, which is (given the transition probabilities) approximately `{'Rainy': 0.57, 'Sunny': 0.43}`. The `transition_probability` represents the change of the weather in the underlying Markov chain. In this example, there is only a 30% chance that tomorrow will be sunny if today is rainy. The `emission_probability` represents how likely Bob is to perform a certain activity on each day. If it is rainy, there is a 50% chance that he is cleaning his apartment; if it is sunny, there is a 60% chance that he is outside for a walk.



A similar example is further elaborated in the [Viterbi algorithm](#) page.

Learning [\[edit\]](#)

The parameter learning task in HMMs is to find, given an output sequence or a set of such sequences, the best set of state transition and emission probabilities. The task is usually to derive the [maximum likelihood](#) estimate of the parameters of the HMM given the set of output sequences. No tractable algorithm is known for solving this problem exactly, but a local maximum likelihood can be derived efficiently using the [Baum–Welch algorithm](#) or the Baldi–Chauvin algorithm. The [Baum–Welch algorithm](#) is a special case of the [expectation-maximization algorithm](#).

Mathematical description [\[edit\]](#)

General description [\[edit\]](#)

A basic, non-Bayesian hidden Markov model can be described as follows:

N	=	number of states
T	=	number of observations
$\theta_{i=1\dots N}$	=	emission parameter for an observation associated with state i
$\phi_{i=1\dots N, j=1\dots N}$	=	probability of transition from state i to state j
$\phi_{i=1\dots N}$	=	N -dimensional vector, composed of $\phi_{i,1\dots N}$; must sum to 1
$x_{t=1\dots T}$	=	state of observation at time t
$y_{t=1\dots T}$	=	observation at time t
$F(y \theta)$	=	probability distribution of an observation, parametrized on θ
$x_{t=2\dots T}$	\sim	$\text{Categorical}(\phi_{x_{t-1}})$
$y_{t=1\dots T}$	\sim	$F(\theta_{x_t})$

Note that, in the above model (and also the one below), the prior distribution of the initial state x_1 is not specified. Typical learning models correspond to assuming a discrete uniform distribution over possible states (i.e. no particular prior distribution is assumed).

In a Bayesian setting, all parameters are associated with random variables, as follows:

N, T	= as above
$\theta_{i=1\dots N}, \phi_{i=1\dots N, j=1\dots N}, \phi_{i=1\dots N}$	= as above
$x_{t=1\dots T}, y_{t=1\dots T}, F(y \theta)$	= as above
α	= shared hyperparameter for emission parameters
β	= shared hyperparameter for transition parameters
$H(\theta \alpha)$	= prior probability distribution of emission parameters, parametrized on α
$\theta_{i=1\dots N}$	$\sim H(\alpha)$
$\phi_{i=1\dots N}$	$\sim \text{Symmetric-Dirichlet}_N(\beta)$
$x_{t=2\dots T}$	$\sim \text{Categorical}(\phi_{x_{t-1}})$
$y_{t=1\dots T}$	$\sim F(\theta_{x_t})$

These characterizations use F and H to describe arbitrary distributions over observations and parameters, respectively. Typically H will be the [conjugate prior](#) of F . The two most common choices of F are [Gaussian](#) and [categorical](#); see below.

Compared with a simple mixture model [\[edit\]](#)

As mentioned above, the distribution of each observation in a hidden Markov model is a [mixture density](#), with the states of the corresponding to mixture components. It is useful to compare the above characterizations for an HMM with the corresponding characterizations, of a [mixture model](#), using the same notation.

A non-Bayesian mixture model:

N	= number of mixture components
T	= number of observations
$\theta_{i=1\dots N}$	= parameter of distribution of observation associated with component i
$\phi_{i=1\dots N}$	= mixture weight, i.e. prior probability of component i
ϕ	= N -dimensional vector, composed of $\phi_{1\dots N}$; must sum to $\mathbf{1}$
$x_{t=1\dots T}$	= component of observation t
$y_{t=1\dots T}$	= observation t
$F(y \theta)$	= probability distribution of an observation, parametrized on θ
$x_{t=1\dots T} \sim \text{Categorical}(\phi)$	
$y_{t=1\dots T} \sim F(\theta_{x_t})$	

A Bayesian mixture model:

N, T	= as above
$\theta_{i=1\dots N}, \phi_{i=1\dots N}, \phi$	= as above
$x_{t=1\dots T}, y_{t=1\dots T}, F(y \theta)$	= as above
α	= shared hyperparameter for component parameters
β	= shared hyperparameter for mixture weights
$H(\theta \alpha)$	= prior probability distribution of component parameters, parametrized on α
$\theta_{i=1\dots N}$	$\sim H(\alpha)$
ϕ	$\sim \text{Symmetric-Dirichlet}_N(\beta)$
$x_{t=1\dots T}$	$\sim \text{Categorical}(\phi)$
$y_{t=1\dots T}$	$\sim F(\theta_{x_t})$

Examples [\[edit\]](#)

The following mathematical descriptions are fully written out and explained, for ease of implementation.

A typical non-Bayesian HMM with Gaussian observations looks like this:

N	= number of states
-----	--------------------

T	=	number of observations
$\phi_{i=1\dots N, j=1\dots N}$	=	probability of transition from state i to state j
$\phi_{i=1\dots N}$	=	N -dimensional vector, composed of $\phi_{i,1\dots N}$; must sum to $\mathbf{1}$
$\mu_{i=1\dots N}$	=	mean of observations associated with state i
$\sigma_{i=1\dots N}^2$	=	variance of observations associated with state i
$x_{t=1\dots T}$	=	state of observation at time t
$y_{t=1\dots T}$	=	observation at time t
$x_{t=2\dots T}$	\sim	$\text{Categorical}(\phi_{x_{t-1}})$
$y_{t=1\dots T}$	\sim	$\mathcal{N}(\mu_{x_t}, \sigma_{x_t}^2)$

A typical Bayesian HMM with Gaussian observations looks like this:

N	=	number of states
T	=	number of observations
$\phi_{i=1\dots N, j=1\dots N}$	=	probability of transition from state i to state j
$\phi_{i=1\dots N}$	=	N -dimensional vector, composed of $\phi_{i,1\dots N}$; must sum to $\mathbf{1}$
$\mu_{i=1\dots N}$	=	mean of observations associated with state i
$\sigma_{i=1\dots N}^2$	=	variance of observations associated with state i
$x_{t=1\dots T}$	=	state of observation at time t
$y_{t=1\dots T}$	=	observation at time t
β	=	concentration hyperparameter controlling the density of the transition matrix
μ_0, λ	=	shared hyperparameters of the means for each state
ν, σ_0^2	=	shared hyperparameters of the variances for each state
$\phi_{i=1\dots N}$	\sim	$\text{Symmetric-Dirichlet}_N(\beta)$
$x_{t=2\dots T}$	\sim	$\text{Categorical}(\phi_{x_{t-1}})$
$\mu_{i=1\dots N}$	\sim	$\mathcal{N}(\mu_0, \lambda \sigma_i^2)$
$\sigma_{i=1\dots N}^2$	\sim	$\text{Inverse-Gamma}(\nu, \sigma_0^2)$
$y_{t=1\dots T}$	\sim	$\mathcal{N}(\mu_{x_t}, \sigma_{x_t}^2)$

A typical non-Bayesian HMM with categorical observations looks like this:

N	=	number of states
T	=	number of observations
$\phi_{i=1\dots N, j=1\dots N}$	=	probability of transition from state i to state j
$\phi_{i=1\dots N}$	=	N -dimensional vector, composed of $\phi_{i,1\dots N}$; must sum to $\mathbf{1}$
V	=	dimension of categorical observations, e.g. size of word vocabulary
$\theta_{i=1\dots N, j=1\dots V}$	=	probability for state i of observing the j th item
$\theta_{i=1\dots N}$	=	V -dimensional vector, composed of $\theta_{i,1\dots V}$; must sum to $\mathbf{1}$
$x_{t=1\dots T}$	=	state of observation at time t
$y_{t=1\dots T}$	=	observation at time t
$x_{t=2\dots T}$	\sim	$\text{Categorical}(\phi_{x_{t-1}})$
$y_{t=1\dots T}$	\sim	$\text{Categorical}(\theta_{x_t})$

A typical Bayesian HMM with categorical observations looks like this:

N	=	number of states
T	=	number of observations
$\phi_{i=1\dots N, j=1\dots N}$	=	probability of transition from state i to state j
$\phi_{i=1\dots N}$	=	N -dimensional vector, composed of $\phi_{i,1\dots N}$; must sum to $\mathbf{1}$
V	=	dimension of categorical observations, e.g. size of word vocabulary
$\theta_{i=1\dots N, j=1\dots V}$	=	probability for state i of observing the j th item
$\theta_{i=1\dots N}$	=	V -dimensional vector, composed of $\theta_{i,1\dots V}$; must sum to $\mathbf{1}$

$x_{t=1...T}$	= state of observation at time t
$y_{t=1...T}$	= observation at time t
α	= shared concentration hyperparameter of θ for each state
β	= concentration hyperparameter controlling the density of the transition matrix
$\phi_{i=1...N}$	$\sim \text{Symmetric-Dirichlet}_N(\beta)$
$\theta_{1...V}$	$\sim \text{Symmetric-Dirichlet}_V(\alpha)$
$x_{t=2...T}$	$\sim \text{Categorical}(\phi_{x_{t-1}})$
$y_{t=1...T}$	$\sim \text{Categorical}(\theta_{x_t})$

Note that in the above Bayesian characterizations, β (a [concentration parameter](#)) controls the density of the transition matrix. That is, with a high value of β (significantly above 1), the probabilities controlling the transition out of a particular state will all be similar, meaning there will be a significant probability of transitioning to any of the other states. In other words, the path followed by the Markov chain of hidden states will be highly random. With a low value of β (significantly below 1), only a small number of the possible transitions out of a given state will have significant probability, meaning that the path followed by the hidden states will be somewhat predictable.

A two-level Bayesian HMM [\[edit\]](#)

An alternative for the above two Bayesian examples would be to add another level of prior parameters for the transition matrix. That is, replace the lines

$$\begin{aligned}\beta &= \text{concentration hyperparameter controlling the density of the transition matrix} \\ \phi_{i=1...N} &\sim \text{Symmetric-Dirichlet}_N(\beta)\end{aligned}$$

with the following:

$$\begin{aligned}\gamma &= \text{concentration hyperparameter controlling how many states are intrinsically likely} \\ \beta &= \text{concentration hyperparameter controlling the density of the transition matrix} \\ \eta &= N\text{-dimensional vector of probabilities, specifying the intrinsic probability of a given state} \\ \eta &\sim \text{Symmetric-Dirichlet}_N(\gamma) \\ \phi_{i=1...N} &\sim \text{Dirichlet}_N(\beta N \eta)\end{aligned}$$

What this means is the following:

1. η is a [probability distribution](#) over states, specifying which states are inherently likely. The greater the probability of a given state in this vector, the more likely is a transition to that state (regardless of the starting state).
2. γ controls the density of η . Values significantly above 1 cause a dense vector where all states will have similar [prior probabilities](#). Values significantly below 1 cause a sparse vector where only a few states are inherently likely (have prior probabilities significantly above 0).
3. β controls the density of the transition matrix, or more specifically, the density of the N different probability vectors $\phi_{i=1...N}$ specifying the probability of transitions out of state i to any other state.

Imagine that the value of β is significantly above 1. Then the different ϕ vectors will be dense, i.e. the probability mass will be spread out fairly evenly over all states. However, to the extent that this mass is unevenly spread, η controls which states are likely to get more mass than others.

Now, imagine instead that β is significantly below 1. This will make the ϕ vectors sparse, i.e. almost all the probability mass is distributed over a small number of states, and for the rest, a transition to that state will be very unlikely. Notice that there are different ϕ vectors for each starting state, and so even if all the vectors are sparse, different vectors may distribute the mass to different ending states. However, for all of the vectors, η controls which ending states are likely to get mass assigned to them. For example, if β is 0.1, then each ϕ will be sparse and, for any given starting state i , the set of states \mathbf{J}_i to which transitions are likely to occur will be very small, typically having only one or two members. Now, if the probabilities in η are all the same (or equivalently, one of the above models without η is used), then for different i , there will be different states in the corresponding \mathbf{J}_i , so that all states are equally likely to occur in any given \mathbf{J}_i . On the other hand, if the values in η are unbalanced, so that one state has a much higher probability than others, almost all \mathbf{J}_i will contain this state; hence, regardless of the starting state, transitions will nearly always occur to this given state.

Hence, a two-level model such as just described allows independent control over (1) the overall density of the

transition matrix, and (2) the density of states to which transitions are likely (i.e. the density of the prior distribution of states in any particular hidden variable \mathbf{x}_i). In both cases this is done while still assuming ignorance over which particular states are more likely than others. If it is desired to inject this information into the model, the probability vector $\boldsymbol{\eta}$ can be directly specified; or, if there is less certainty about these relative probabilities, a non-symmetric [Dirichlet distribution](#) can be used as the prior distribution over $\boldsymbol{\eta}$. That is, instead of using a symmetric Dirichlet distribution with the single parameter γ (or equivalently, a general Dirichlet with a vector all of whose values are equal to γ), use a general Dirichlet with values that are variously greater or less than γ , according to which state is more or less preferred.

Applications [\[edit\]](#)

HMMs can be applied in many fields where the goal is to recover a data sequence that is not immediately observable (but other data that depend on the sequence are). Applications include:

- [Single Molecule Kinetic analysis](#)^[16]
- [Cryptanalysis](#)
- [Speech recognition](#)
- [Speech synthesis](#)
- [Part-of-speech tagging](#)
- Document Separation in scanning solutions
- [Machine translation](#)
- [Partial discharge](#)
- [Gene prediction](#)
- [Alignment of bio-sequences](#)
- [Time Series Analysis](#)
- [Activity recognition](#)
- [Protein folding](#)^[17]
- [Metamorphic Virus Detection](#)^[18]
- [DNA Motif Discovery](#)^[19]

History [\[edit\]](#)

The forward and backward recursions used in HMM as well as computations of marginal smoothing probabilities were first described by [Ruslan L. Stratonovich](#) in 1960^[6] (pages 160—162) and in the late 1950s in his papers in Russian. The Hidden Markov Models were later described in a series of statistical papers by [Leonard E. Baum](#) and other authors in the second half of the 1960s. One of the first applications of HMMs was [speech recognition](#), starting in the mid-1970s.^{[20][21][22][23]}

In the second half of the 1980s, HMMs began to be applied to the analysis of biological sequences,^[24] in particular [DNA](#). Since then, they have become ubiquitous in the field of [bioinformatics](#).^[25]

Types [\[edit\]](#)

Hidden Markov models can model complex [Markov](#) processes where the states emit the observations according to some probability distribution. One such example is the [Gaussian](#) distribution, in such a Hidden Markov Model the states output are represented by a [Gaussian](#) distribution.

Moreover it could represent even more complex behavior when the output of the states is represented as mixture of two or more Gaussians, in which case the [probability](#) of generating an observation is the product of the probability of first selecting one of the Gaussians and the probability of generating that observation from that Gaussian.

Extensions [\[edit\]](#)

In the hidden Markov models considered above, the state space of the hidden variables is discrete, while the observations themselves can either be discrete (typically generated from a [categorical distribution](#)) or continuous (typically from a [Gaussian distribution](#)). Hidden Markov models can also be generalized to allow continuous state spaces. Examples of such models are those where the Markov process over hidden variables is a [linear dynamical system](#), with a linear relationship among related variables and where all hidden and observed variables follow a [Gaussian distribution](#). In simple cases, such as the linear dynamical system just mentioned, exact inference is tractable (in this case, using the [Kalman filter](#)); however, in general, exact inference in HMMs with continuous latent variables is infeasible, and approximate methods must be used, such

as the [extended Kalman filter](#) or the [particle filter](#).

Hidden Markov models are [generative models](#), in which the [joint distribution](#) of observations and hidden states, or equivalently both the [prior distribution](#) of hidden states (the *transition probabilities*) and [conditional distribution](#) of observations given states (the *emission probabilities*), is modeled. The above algorithms implicitly assume a [uniform](#) prior distribution over the transition probabilities. However, it is also possible to create hidden Markov models with other types of prior distributions. An obvious candidate, given the categorical distribution of the transition probabilities, is the [Dirichlet distribution](#), which is the [conjugate prior](#) distribution of the categorical distribution. Typically, a symmetric Dirichlet distribution is chosen, reflecting ignorance about which states are inherently more likely than others. The single parameter of this distribution (termed the *concentration parameter*) controls the relative density or sparseness of the resulting transition matrix. A choice of 1 yields a uniform distribution. Values greater than 1 produce a dense matrix, in which the transition probabilities between pairs of states are likely to be nearly equal. Values less than 1 result in a sparse matrix in which, for each given source state, only a small number of destination states have non-negligible transition probabilities. It is also possible to use a two-level prior Dirichlet distribution, in which one Dirichlet distribution (the upper distribution) governs the parameters of another Dirichlet distribution (the lower distribution), which in turn governs the transition probabilities. The upper distribution governs the overall distribution of states, determining how likely each state is to occur; its concentration parameter determines the density or sparseness of states. Such a two-level prior distribution, where both concentration parameters are set to produce sparse distributions, might be useful for example in [unsupervised part-of-speech tagging](#), where some parts of speech occur much more commonly than others; learning algorithms that assume a uniform prior distribution generally perform poorly on this task. The parameters of models of this sort, with non-uniform prior distributions, can be learned using [Gibbs sampling](#) or extended versions of the [expectation-maximization algorithm](#).

An extension of the previously described hidden Markov models with [Dirichlet](#) priors uses a [Dirichlet process](#) in place of a Dirichlet distribution. This type of model allows for an unknown and potentially infinite number of states. It is common to use a two-level Dirichlet process, similar to the previously described model with two levels of Dirichlet distributions. Such a model is called a *hierarchical Dirichlet process hidden Markov model*, or *HDP-HMM* for short. It was originally described under the name "Infinite Hidden Markov Model"^[2] and was further formalized in^[3].

A different type of extension uses a [discriminative model](#) in place of the [generative model](#) of standard HMMs. This type of model directly models the conditional distribution of the hidden states given the observations, rather than modeling the joint distribution. An example of this model is the so-called *maximum entropy Markov model* (MEMM), which models the conditional distribution of the states using [logistic regression](#) (also known as a "[maximum entropy](#) model"). The advantage of this type of model is that arbitrary features (i.e. functions) of the observations can be modeled, allowing domain-specific knowledge of the problem at hand to be injected into the model. Models of this sort are not limited to modeling direct dependencies between a hidden state and its associated observation; rather, features of nearby observations, of combinations of the associated observation and nearby observations, or in fact of arbitrary observations at any distance from a given hidden state can be included in the process used to determine the value of a hidden state. Furthermore, there is no need for these features to be [statistically independent](#) of each other, as would be the case if such features were used in a generative model. Finally, arbitrary features over pairs of adjacent hidden states can be used rather than simple transition probabilities. The disadvantages of such models are: (1) The types of prior distributions that can be placed on hidden states are severely limited; (2) It is not possible to predict the probability of seeing an arbitrary observation. This second limitation is often not an issue in practice, since many common usages of HMM's do not require such predictive probabilities.

A variant of the previously described discriminative model is the linear-chain [conditional random field](#). This uses an undirected graphical model (aka [Markov random field](#)) rather than the directed graphical models of MEMM's and similar models. The advantage of this type of model is that it does not suffer from the so-called *label bias* problem of MEMM's, and thus may make more accurate predictions. The disadvantage is that training can be slower than for MEMM's.

Yet another variant is the *factorial hidden Markov model*, which allows for a single observation to be conditioned on the corresponding hidden variables of a set of K independent Markov chains, rather than a single Markov chain. It is equivalent to a single HMM, with N^K states (assuming there are N states for each chain), and therefore, learning in such a model is difficult: for a sequence of length T , a straightforward Viterbi algorithm has complexity $O(N^{2K} T)$. To find an exact solution, a junction tree algorithm could be used, but it results in an $O(N^{K+1} K T)$ complexity. In practice, approximate techniques, such as variational approaches, could be used.^[26]

All of the above models can be extended to allow for more distant dependencies among hidden states, e.g.

allowing for a given state to be dependent on the previous two or three states rather than a single previous state; i.e. the transition probabilities are extended to encompass sets of three or four adjacent states (or in general K adjacent states). The disadvantage of such models is that dynamic-programming algorithms for training them have an $O(N^K T)$ running time, for K adjacent states and T total observations (i.e. a length- T Markov chain).

Another recent extension is the *triplet Markov model*,^[27] in which an auxiliary underlying process is added to model some data specificities. Many variants of this model have been proposed. One should also mention the interesting link that has been established between the *theory of evidence* and the *triplet Markov models*^[10] and which allows to fuse data in Markovian context^[11] and to model nonstationary data.^{[12][13]}

See also ^[edit]

- Andrey Markov
- Baum–Welch algorithm
- Bayesian inference
- Bayesian programming
- Conditional random field
- Estimation theory
- HHpred / HHsearch free server and software for protein sequence searching
- HMMER, a free hidden Markov model program for protein sequence analysis
- Hidden Bernoulli model
- Hidden semi-Markov model
- Hierarchical hidden Markov model
- Layered hidden Markov model
- Poisson hidden Markov model
- Sequential dynamical system
- Stochastic context-free grammar
- Time Series Analysis
- Variable-order Markov model
- Viterbi algorithm











References ^[edit]

- ↑ Baum, L. E.; Petrie, T. (1966). "Statistical Inference for Probabilistic Functions of Finite State Markov Chains" ^[↗]. *The Annals of Mathematical Statistics* **37** (6): 1554–1563. doi:10.1214/aoms/1177699147 ^[↗]. Retrieved 28 November 2011.
- ↑ Baum, L. E.; Eagon, J. A. (1967). "An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology" ^[↗]. *Bulletin of the American Mathematical Society* **73** (3): 360. doi:10.1090/S0002-9904-1967-11751-8 ^[↗]. Zbl 0157.11101 ^[↗].
- ↑ Baum, L. E.; Sell, G. R. (1968). "Growth transformations for functions on manifolds" ^[↗]. *Pacific Journal of Mathematics* **27** (2): 211–227. doi:10.2140/pjm.1968.27.211 ^[↗]. Retrieved 28 November 2011.
- ↑ Baum, L. E.; Petrie, T.; Soules, G.; Weiss, N. (1970). "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains" ^[↗]. *The Annals of Mathematical Statistics* **41**: 164. doi:10.1214/aoms/1177697196 ^[↗]. JSTOR 2239727 ^[↗]. MR MR287613 ^[↗]. Zbl 0188.49603 ^[↗].
- ↑ Baum, L.E. (1972). "An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of a Markov Process". *Inequalities* **3**: 1–8.
- ↑ ^a ^b Stratonovich, R.L. (1960). "Conditional Markov Processes". *Theory of Probability and its Applications* **5** (2): 156–178. doi:10.1137/1105015 ^[↗].
- ↑ Thad Stamer, Alex Pentland. *Real-Time American Sign Language Visual Recognition From Video Using Hidden Markov Models* ^[↗]. Master's Thesis, MIT, Feb 1995, Program in Media Arts
- ↑ B. Pardo and W. Birmingham. *Modeling Form for On-line Following of Musical Performances* ^[↗]. AAAI-05 Proc., July 2005.
- ↑ Satish L, Gururaj BI (April 2003). "Use of hidden Markov models for partial discharge pattern classification" ^[↗]. *IEEE Transactions on Dielectrics and Electrical Insulation*.
- ↑ ^a ^b Pr. Pieczynski ^[↗], W. Pieczynski, Multisensor triplet Markov chains and theory of evidence, International Journal of Approximate Reasoning, Vol. 45, No. 1, pp. 1-16, 2007.
- ↑ ^a ^b Boudaren et al. ^[↗], M. Y. Boudaren, E. Monfrini, W. Pieczynski, and A. Aissani, Dempster-Shafer fusion of multisensor signals in nonstationary Markovian context, EURASIP Journal on Advances in Signal Processing, No. 134, 2012.
- ↑ ^a ^b Lanchantin et al. ^[↗], P. Lanchantin and W. Pieczynski, Unsupervised restoration of hidden non stationary Markov chain using evidential priors, IEEE Trans. on Signal Processing, Vol. 53, No. 8, pp. 3091-3098, 2005.
- ↑ ^a ^b Boudaren et al. ^[↗], M. Y. Boudaren, E. Monfrini, and W. Pieczynski, Unsupervised segmentation of random















13. [Boudarene et al.](#), iv. T. Boudarene, E. Monimini, and vv. Pieczynski, Unsupervised segmentation of random discrete data hidden with switching noise distributions, *IEEE Signal Processing Letters*, Vol. 19, No. 10, pp. 619-622, October 2012.
14. [Lawrence R. Rabiner](#) (February 1989). "A tutorial on Hidden Markov Models and selected applications in speech recognition" [\(PDF\)](#). *Proceedings of the IEEE* **77** (2): 257–286. doi:10.1109/5.18626 [\[1\]](#) [\[PDF\]](#)
15. [Newberg, L.](#) (2009). "Error statistics of hidden Markov model and hidden Boltzmann model results" [\[PDF\]](#). *BMC Bioinformatics* **10**: 212. doi:10.1186/1471-2105-10-212 [\[PDF\]](#). PMC 2722652 [\[PDF\]](#). PMID 19589158 [\[PDF\]](#).
16. [NICOLAI, CHRISTOPHER](#) (2013). "SOLVING ION CHANNEL KINETICS WITH THE QuB SOFTWARE". *Biophysical Reviews and Letters* **8** (3n04): 191–211. doi:10.1142/S1793048013300053 [\[PDF\]](#).
17. [Stigler, J.; Ziegler, F.; Gieseke, A.; Gebhardt, J. C. M.; Rief, M.](#) (2011). "The Complex Folding Network of Single Calmodulin Molecules". *Science* **334** (6055): 512–516. doi:10.1126/science.1207598 [\[PDF\]](#). PMID 22034433 [\[PDF\]](#).
18. [Wong, W.; Stamp, M.](#) (2006). "Hunting for metamorphic engines". *Journal in Computer Virology* **2** (3): 211–229. doi:10.1007/s11416-006-0028-7 [\[PDF\]](#).
19. [Wong, K. -C.; Chan, T. -M.; Peng, C.; Li, Y.; Zhang, Z.](#) (2013). "DNA motif elucidation using belief propagation" [\[PDF\]](#). *Nucleic Acids Research* **41** (16): e153. doi:10.1093/nar/gkt574 [\[PDF\]](#). PMC 3763557 [\[PDF\]](#). PMID 23814189 [\[PDF\]](#).
20. [Baker, J.](#) (1975). "The DRAGON system—An overview". *IEEE Transactions on Acoustics, Speech, and Signal Processing* **23**: 24–29. doi:10.1109/TASSP.1975.1162650 [\[PDF\]](#).
21. [Jelinek, F.; Bahl, L.; Mercer, R.](#) (1975). "Design of a linguistic statistical decoder for the recognition of continuous speech". *IEEE Transactions on Information Theory* **21** (3): 250. doi:10.1109/TIT.1975.1055384 [\[PDF\]](#).
22. [Xuedong Huang; M. Jack; Y. Ariki](#) (1990). *Hidden Markov Models for Speech Recognition*. Edinburgh University Press. ISBN 0-7486-0162-7.
23. [Xuedong Huang; Alex Acero; Hsiao-Wuen Hon](#) (2001). *Spoken Language Processing*. Prentice Hall. ISBN 0-13-022616-5.
24. [M. Bishop and E. Thompson](#) (1986). "Maximum Likelihood Alignment of DNA Sequences". *Journal of Molecular Biology* **190** (2): 159–165. doi:10.1016/0022-2836(86)90289-5 [\[PDF\]](#). PMID 3641921 [\[PDF\]](#).
25. [Richard Durbin; Sean R. Eddy; Anders Krogh; Graeme Mitchison](#) (1999). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press. ISBN 0-521-62971-3.
26. [Ghahramani, Zoubin; Jordan, Michael I.](#) (1997). "Factorial Hidden Markov Models". *Machine Learning* **29** (2/3): 245–273. doi:10.1023/A:1007425814087 [\[PDF\]](#).
27. [Triplet Markov Chain](#) [\[PDF\]](#), W. Pieczynski, Chaînes de Markov Triplet, Triplet Markov Chains, Comptes Rendus de l'Académie des Sciences – Mathématique, Série I, Vol. 335, No. 3, pp. 275-278, 2002.


External links [\[edit\]](#)

Concepts [\[edit\]](#)

- Teif V. B. and K. Rippe (2010) Statistical–mechanical lattice models for protein–DNA binding in chromatin. *J. Phys.: Condens. Matter*, **22**, 414105, <http://iopscience.iop.org/0953-8984/22/41/414105>
- [A Revealing Introduction to Hidden Markov Models](#)  by Mark Stamp, San Jose State University.
- [Fitting HMM's with expectation-maximization – complete derivation](#) 
- [Switching Autoregressive Hidden Markov Model \(SAR HMM\)](#) 
- [A step-by-step tutorial on HMMs](#)  (*University of Leeds*)
- [Hidden Markov Models](#)  (*an exposition using basic mathematics*)
- [Hidden Markov Models](#)  (*by Narada Warakagoda*)
- [Hidden Markov Models: Fundamentals and Applications](#) [Part 1](#) , [Part 2](#)  (*by V. Petrushin*)
- [Lecture on a Spreadsheet](#) by Jason Eisner, [Video](#)  and [interactive spreadsheet](#) 

Software [\[edit\]](#)

- [Hidden Markov Model \(HMM\) Toolbox for Matlab](#)  (*by Kevin Murphy*)
- [Hidden Markov Model Toolkit \(HTK\)](#)  (*a portable toolkit for building and manipulating hidden Markov models*)
- [Hidden Markov Model R-Package](#)  to set up, apply and make inference with discrete time and discrete space Hidden Markov Models
- [HMMlib](#)  (*an optimized library for work with general (discrete) hidden Markov models*)
- [parredHMMlib](#)  (*a parallel implementation of the forward algorithm and the Viterbi algorithm. Extremely fast for HMMs with small state spaces*)
- [zipHMMlib](#)  (*a library for general (discrete) hidden Markov models, exploiting repetitions in the input sequence to greatly speed up the forward algorithm. Implementation of the posterior decoding algorithm and the Viterbi algorithm are also provided.*)
- [GHMM Library](#)  (*home page of the GHMM Library project*)
- [Jahmm Java Library](#)  (*general-purpose Java library*)
- [HMM and other statistical programs](#)  (*Implementation in C by Tapas Kanungo*)
- [The hmm package](#)  [A Haskell](#)  library for working with Hidden Markov Models.
- [GT2K](#)  Georgia Tech Gesture Toolkit (referred to as GT2K)
- [Hidden Markov Models -online calculator for HMM – Viterbi path and probabilities](#). Examples with perl source code. 
- [A discrete Hidden Markov Model class, based on OpenCV.](#) 

- [depmixS4](#)  R-Package (Hidden Markov Models of GLMs and Other Distributions in S4)
- **MLPACK** contains a C++ implementation of HMMs

v · t · e	Stochastic processes	[hide]
Discrete time	Bernoulli process · Branching process · Chinese restaurant process · Galton–Watson process · Independent and identically distributed random variables · Markov chain · Moran process · Random walk (Loop-erased · Self-avoiding)	
Continuous time	Bessel process · Birth–death process · Brownian motion (Bridge · Excursion · Fractional · Geometric · Meander) · Cauchy process · Contact process · Continuous-time random walk · Cox process · Diffusion process · Empirical process · Feller process · Fleming–Viot process · Gamma process · Hunt process · Interacting particle systems · Itô diffusion · Itô process · Jump diffusion · Jump process · Lévy process · Local time · Markov additive process · McKean–Maslov process · Ornstein–Uhlenbeck process · Poisson process (Compound · Non-homogeneous · Point process) · Schramm–Loewner evolution · Semimartingale · Sigma-martingale · Stable process · Superprocess · Telegraph process · Variance gamma process · Wiener process · Wiener sausage	
Both	Branching process · Gaussian process · Hidden Markov model (HMM) · Markov process · Martingale (Differences · Local · Sub- · Super-) · Random dynamical system · Regenerative process · Renewal process · White noise	
Fields and other	Dirichlet process · Gaussian random field · Gibbs measure · Hopfield model · Ising model (Potts model · Boolean network) · Markov random field · Percolation · Pitman–Yor process · Point process (Cox · Poisson) · Random field · Random graph	
Time series models	Autoregressive conditional heteroskedasticity (ARCH) model · Autoregressive integrated moving average (ARIMA) model · Autoregressive (AR) model · Autoregressive–moving-average (ARMA) model · Generalized autoregressive conditional heteroskedasticity (GARCH) model · Moving-average (MA) model	
Financial models	Black–Derman–Toy · Black–Karasinski · Black–Scholes · Chen · Constant elasticity of variance (CEV) · Cox–Ingersoll–Ross (CIR) · Garman–Kohlhagen · Heath–Jarrow–Morton (HJM) · Heston · Ho–Lee · Hull–White · LIBOR market · Rendleman–Barter · SABR volatility · Vašíček · Wilkie	
Actuarial models	Bühlmann · Cramér–Lundberg · Risk process · Sparre–Anderson	
Queueing models	Bulk · Fluid · Generalized queueing network · M/G/1 · M/M/1 · M/M/c	
Properties	Càdlàg paths · Continuous · Continuous paths · Ergodic · Exchangeable · Feller-continuous · Gauss–Markov · Markov · Mixing · Piecewise deterministic · Predictable · Progressively measurable · Self-similar · Stationary · Time-reversible	
Limit theorems	Central limit theorem · Donsker's theorem · Doob's martingale convergence theorems · Ergodic theorem · Fisher–Tippett–Gnedenko theorem · Large deviation principle · Law of large numbers (weak/strong) · Law of the iterated logarithm · Maximal ergodic theorem · Sanov's theorem	
Inequalities	Burkholder–Davis–Gundy · Doob's martingale · Kunita–Watanabe	
Tools	Cameron–Martin formula · Convergence of random variables · Doléans-Dade exponential · Doob decomposition theorem · Doob–Meyer decomposition theorem · Doob's optional stopping theorem · Dynkin's formula · Feynman–Kac formula · Filtration · Girsanov theorem · Infinitesimal generator · Itô integral · Itô's lemma · Kolmogorov continuity theorem · Kolmogorov extension theorem · Lévy–Prokhorov metric · Malliavin calculus · Martingale representation theorem · Optional stopping theorem · Prohorov theorem · Quadratic variation · Reflection principle · Skorokhod integral · Skorokhod's representation theorem · Skorokhod space · Snell envelope · Stochastic differential equation (Tanaka) · Stopping time · Stratonovich integral · Uniform integrability · Usual hypotheses · Wiener space (Classical · Abstract)	
Disciplines	Actuarial mathematics · Econometrics · Ergodic theory · Extreme value theory (EVT) · Large deviations theory · Mathematical finance · Mathematical statistics · Probability theory · Queueing theory · Renewal theory · Ruin theory · Statistics · Stochastic analysis · Time series analysis · Machine learning	
List of topics · Category		
Authority control	GND: 4352479-5 	

Categories: [Bioinformatics](#) | [Hidden Markov models](#) | [Markov models](#)

This page was last modified on 28 August 2015, at 15:46.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Developers](#) [Mobile view](#)



a
WIKIMEDIA
project



Powered By
MediaWiki