Hidden Markov Models in Sequence Analysis

Data Analysis in Genome Biology
GEN242

Thomas Girke

May 29, 2018

Outline

Outline

Markov Chains and Hidden Markov Models

- Named after Russian mathematician Andrey Markov (1856-1922).
- Developed in speech recognition area where voice recording is transformed into words.
  - Speech recording represented in frames of 10-20 ms ($\sim$sequence).
  - Frames are assigned to large number of categories (*e.g.* 256).
  - Speech is represented in long sequence of category labels.
  - Challenge: variations in sound, word frequency, etc.

Utilities in Sequence Analysis

- Many focus on single sequence classification
- Utilities
  - Gene finding (GeneScan, HMMer, GeneMark, etc.)
  - Sequence classification to families
  - Alignment and searching (HMMER, SAM)
  - Splice site identification
  - Promoter finding
  - Tiling array analysis
  - Prediction of secondary structure: α helices and β sheets
  - ...

Outline

Markov Chains

- Simpler cousin of Hidden Markov models (HMMs)
- A Markov chain is a stochastic process with the Markov properties:
  - Given a present state, future states are independent of the past states.
  - The description of the present state fully captures all the information that could influence the future evolution of the process.
  - Future states will be reached through a probabilistic process instead of a deterministic one.

Markov Chain for DNA Sequence

- Sequence model where the probability of a symbol (residue) depends only on the previous symbol, and not the entire previous sequence.
- Figure 1: Graphical representation as states (symbols) and transitions between them (arrows):

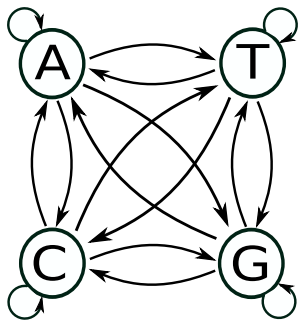


Figure 1: Note, start and end positions not considered.

- DNA example:
  - States: four symbols A, T, G, C
  - Transitions: arrows.
- Probability parameters: transition probabilities a_{st}

$$a_{st} = P(x_i = t | x_{i-1} = s) \tag{1}$$

- For a probabilistic model, the probability of a sequence x of length L can be defined as:

$$P(x) = P(x_L|x_{L-1})P(x_L - 1|x_{L-2})...P(x_2|x_1)P(x_1) = P(x_1)\prod_{i=2}^{L} a_{x_{i-1}x_i} \tag{2}$$

Example: CpG Islands

- In mammalian genomes Cs are frequently methylated, which makes them prone to mutate to Ts.
- Consequence: CpG dinucleotides are less frequent than expected by chance.
- Exception: Promoter boundary regions are enriched in CpG pairs (islands) for biological reasons.
- Questions: How can we identify these CpG islands?

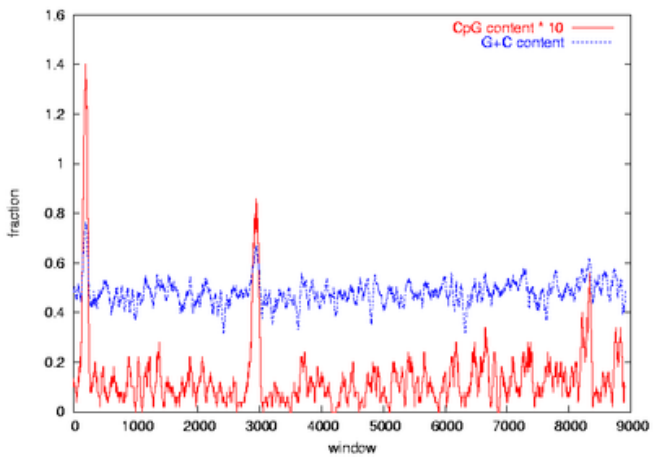


Figure 2: CpG Island Plot

- In Figure 2 the GC (blue) and CpG (red) content is plotted from the 5' to 3' end of a human gene using a sliding window of 500bp.

Problem: Identify CpG Islands in Sequences

- Markov chain to distinguish between CpG islands and non-CpG regions.
- Example: two sequence sets, one containing CpG islands (model+) and another without them (model-).
- Determine transition frequencies of A, C, G or T in two sequence sets for two Markov models:

CpG Islands

$$a_{st}^{+} = \frac{c_{st}^{+}}{\sum_{t'} c_{st'}^{+}}$$

+	A	C	G	T
A	0.18	0.27	0.43	0.12
C	0.17	0.37	0.27	0.19
G	0.16	0.34	0.38	0.13
T	0.07	0.36	0.38	0.18

non-CpG Islands

$$a_{st}^{-} = \frac{c_{st}^{-}}{\sum_{t'} c_{st'}^{-}}$$

-	A	C	G	T
A	0.30	0.21	0.29	0.21
C	0.32	0.30	0.08	0.30
G	0.25	0.25	0.30	0.21
T	0.18	0.24	0.29	0.29

Table 1: CpG islands and non-CpG islands.

- Note: X to C/G are more frequent in CpG island data set.

Problem: Identify CpG Islands in Sequences

- To discriminate between the two models, one can compute the *log-odds* or *log likelihood* ratios:

$$S(x) = log \frac{P(x|model+)}{P(x|model-)} = \sum_{i=1}^{L} log \frac{a^{+}_{x_{i-1}x_i}}{a^{-}_{x_{i-1}x_i}} = \sum_{i=1}^{L} \beta_{x_{i-1}x_i} \quad (3)$$

β*	A	C	G	T
A	-0.74	0.42	0.58	-0.80
C	-0.91	0.30	1.81	-0.69
G	-0.62	0.46	0.33	-0.73
T	-1.17	0.57	0.39	-0.68

*$\beta_{x_{i-1}x_i}$ are log likelihood ratios of transition probabilities; here in bit units (log_2)

- If $S(x) > 0$ then x is likely a CpG island.
- Note: multiplications of probabilities often result in underflow errors on computers. This can be avoided by using log likelihood ratios that are computed by additions and always result in reasonable values.

How to Identify CpG Island in New Sequences?

- Markov chain model (previous slide) could be used:
  - Calculate log-odds score for a given window size (*e.g.* 100)
  - CpG islands would stand out by positive scores.
  - Challenges: optimal window size, boundaries often not sharp, etc.
- Better solution: hidden Markov models (HMMs)

Outline

Hidden Markov Models

- Goal: simulate in one model the Markov chains of the "CpG Island" in the "Sea of Non-Island" genomic sequence.
- Solution: introduction of two states the sequences can be derived from:
  - CpG Island States: A_+, C_+, G_+, T_+
  - Sea of Non-Islands: A_-, C_-, G_-, T_-
- Transition probabilities:
  - Within groups the transition probabilities are set close to transition probabilities of the component model.
  - In addition, small probability to switch between components.
  - Higher probability of switching from '+' to '-' than vice versa.
- Consequence: model spends more time in the '-' non-island states than the '+' island states.

Example: HMM for CpG Islands

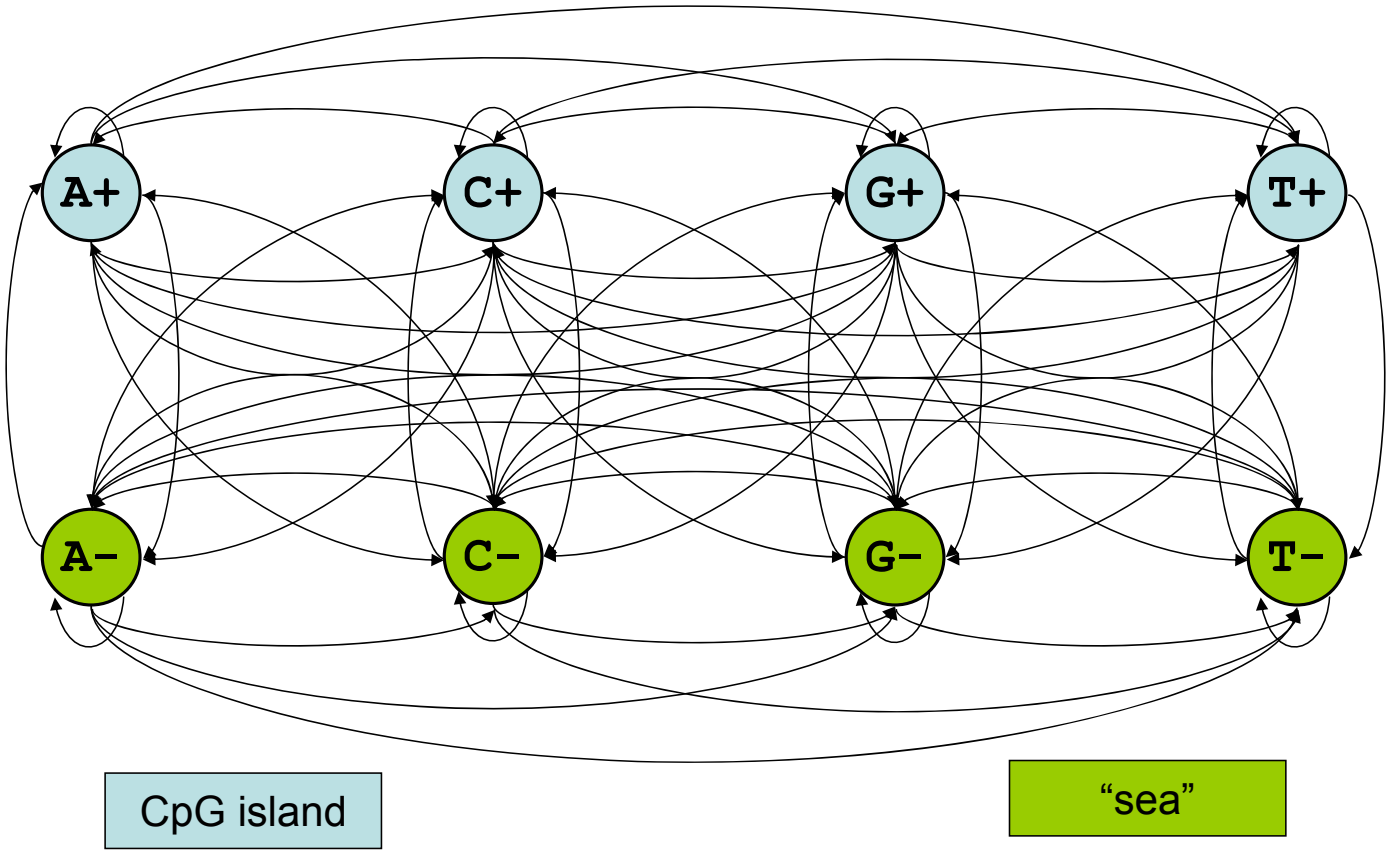


Figure 3: HMM for CpG Islands

Difference Markov Chain and Hidden Markov Models

- No one-to-one relationship between states and symbols.
- It is no longer possible to know what state the model was in when x_i was generated.
- CpG example: looking at a symbol, *e.g.* C, doesn't tell whether it was emitted by state C_+ or state C_-.

Definition of an HMM

Definitions of the probability of a sequence of symbols and states.

State Sequence or Paths

- The state sequence is called the *path* or π.
- The chain of states (k of length l) follows the transition probabilities:

$$a_{kl} = P(\pi_i = l | \pi_{i-1} = k) \tag{4}$$

- The probabilities from the start and end states are a_{0k} and a_{k0}.

Emission Probabilities

- The decoupling of symbol (b) from state (k) sequences is formalized with the *emission* probabilities, where a state emits a symbol from a distribution over all possible symbols:

$$e_k(b) = P(x_i = b | \pi_i = k) \tag{5}$$

- It defines the probability that symbol b is seen when in state k.
- Emission probabilities for CpG island model are 0 or 1.

What is Hidden in HMMs?

- HMMs can be used as generative models for strings, (*e.g.* CGCGCGCG), but the states remain always hidden.
- How do we generate strings from HMMs?
  - Choose a new state based on the transition probabilities.
  - Choose a symbol based on the emission probabilities.

CpG Islands with Emission Probabilities

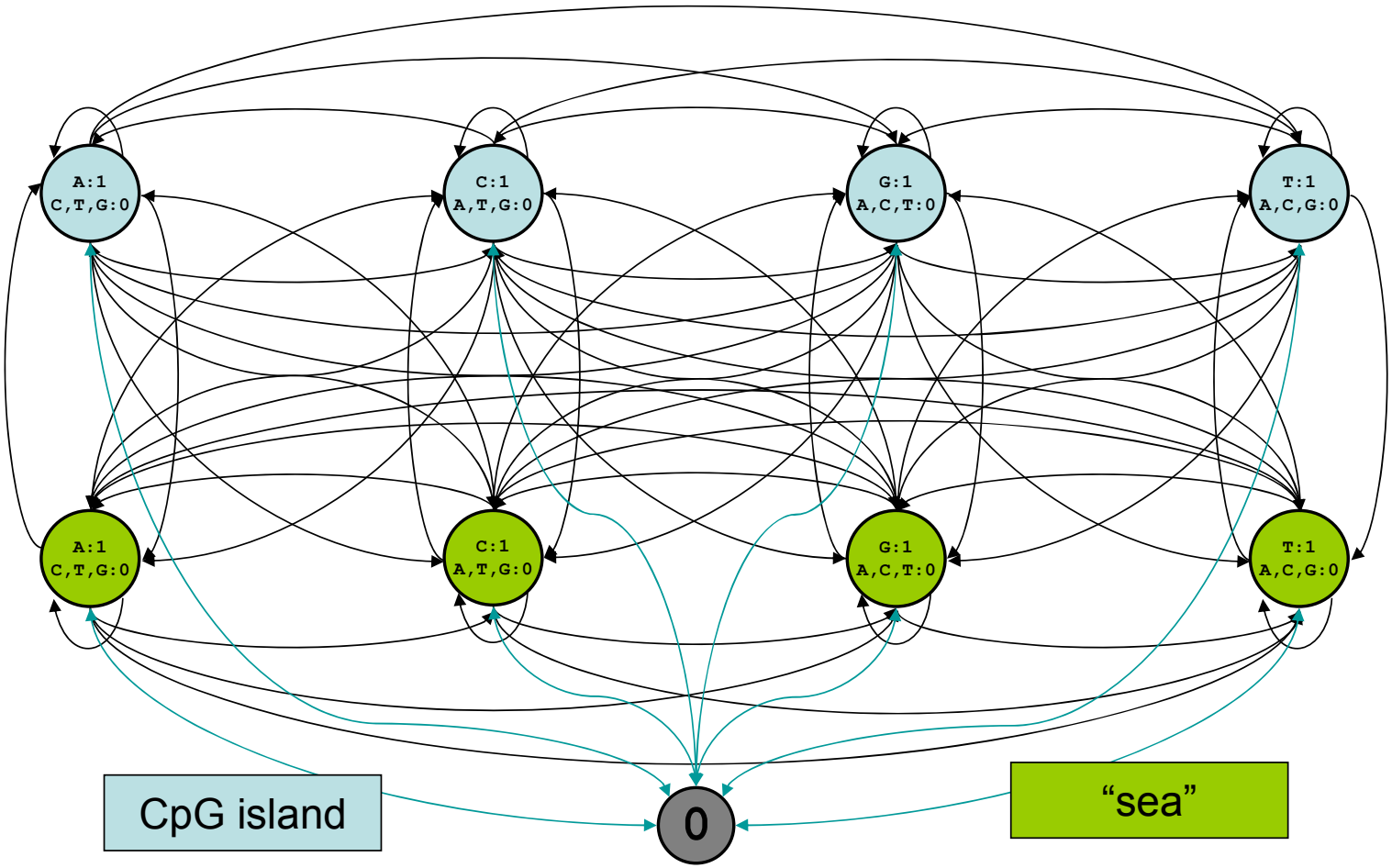


Figure 4: Emission Probabilities

Example: Dishonest Casino

- Casino switches from Fair to Loaded die with probability of 0.05 and back with 0.1.
- Loaded die has probability of 0.5 of a six and 0.1 for the other numbers.
- Entire process has different probabilities in each state (Markov chain).

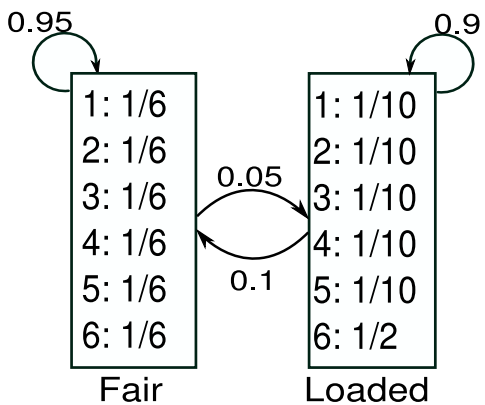


Figure 5: HMM for fair and loaded die.
Emission probabilities $e()$ given in boxes.

- It is an example of a hidden Markov model because the type of die is unknown for the observer - *state sequence is hidden*.

Outline

HMM Algorithms

- Viterbi algorithm (most probable path)
- Forward algorithms ($P(x)$)
- Backward algorithm (posterior probabilities)
- Baum-Welch algorithm (learning)

HMMs Are Generative Models

- An HMM is a generative model that emits sequences.
- A sequence can be generated from an HMM as follows:
  1. Choose a state π_1 according to probabilities a_{0i}.
  2. In π_1 an observation is emitted according to distribution e_{π_1}.
  3. A new state π_2 is chosen according to the transition probabilities $a_{\pi_1 i}$
  4. Iteration over steps 1-3 result in sequence of random, artificial observations.
- Expression: $P(x)$ *is the probability that sequence* x *was generated by HMM.*
- The joint probability of an observed sequence x and a state sequence π can now be defined as:

$$P(x, \pi) = a_{0\pi_1} \prod_{i=1}^{L} e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}}, \tag{6}$$

where we require: $\pi_{L+1} = 0$ and $\pi_0 = 0$

- Note: equation (eq 6) is HMM analogue of equation (eq 2).
- Example: probability of sequence CGCG emitted by state sequence C_+, G_-, C_-, G_+ (emission probabilities are 1):

$$a_{0,C_+} * 1 * a_{C_+,G_-} * 1 * a_{G_-,C_-} * 1 * a_{C_-,G_+} * 1 * a_{C_-,0}$$

Determining the Most Probable Path

- Problem: given a sequence x generated by HMM, how do we determine the most probable state path?
- Identifying the state path is called *decoding* in speech recognition.
- The most commonly used decoding method is the Viterbi algorithm, which is a dynamic programming approach.
- The path with the highest probability (Viterbi path) is:

$$\pi^* = argmax_\pi P(x, \pi) \tag{7}$$

- In other words: $argmax_\pi$ is the value of π for which $P(x, \pi)$ has the largest value.
- Note: the $P(x, \pi)$ component is defined in equation (eq 6).
- The most probable path π^* can be found recursively.

Viterbi Algorithm

- Suppose the probability $v_k(i)$ of the most probable path ending in k with observation i is known for all states k, then the probabilities can be calculated for observations x_{i+1} as

$$v_l(i+1) = e_l(x_{i+1}) \max_k(v_k(i)a_{kl}) \tag{8}$$

- All sequences have to start in state 0 where $v_0(0) = 1$.
- Use dynamic programming to tabulate the values of matrix v.
- By keeping pointers backwards the state sequence can be found by backtracking.

Viterbi Algorithm

Initialization ($i = 0$):	$v_0(0) = 1$, $v_k(0)$ for $k > 0$
Recursion ($i = 1...L$):	$v_l(i) = e_l(x_i) max_k(v_k(i-1)a_{kl})$
	Keep back pointer for argmax.
Termination:	$P(x, \pi^*) = max_k(v_k(L)a_{k0})$
	Find π^* with traceback.

Example: Viterbi Table for CpG Islands

- The following table shows the full table of v values for the sequence CGCG and the CpG island model in Figure 3 using values from CpG frequency Table 1.

v		**C**	**G**	**C**	**G**
B	1	0	0	0	0
A_+	0	0	0	0	0
C_+	0	**0.13**	0	**0.012**	0
G_+	0	0	**0.034**	0	**0.0032**
T_+	0	0	0	0	0
A_-	0	0	0	0	0
C_-	0	0.13	0	0.0026	0
G_-	0	0	0.010	0	0.0002
T_-	0	0	0	0	0

- The most probable path π^* is given in bold face.
- If applied to a longer sequence then π^* will switch between '+' and '-' components of the model, which defines the boundaries between the predicted CpG islands.
- Note: multiplications of probabilities often result in underflow errors on computers. To avoid this, the Viterbi algorithms should always be computed in log space, *e.g.* $log(v_l(i))$, which turns products into sums and the numbers stay reasonable.

Example: Dishonest Casino

- For a sequence of dice rolls, one can now find the most probable path through the model shown in Figure 5.
- Example:

```
Rolls   651166453132651245636664631636663162326455236266666251511
Die     LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLFFFFFFF
Viterbi LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFF
```

- The dice rolls were either generated with the fair die (F) or the loaded one (L) as shown in the second line.
- The Viterbi algorithm was used to predict the state sequence (type of die for each roll). The results are given in the last line.

The Forward Algorithm: Computing $P(x)$

- As shown before, Markov chains can be used to calculate the probability of a sequence $P(x)$ with equation (eq 2).
- To calculate this probability for an HMM, one must add the probabilities for all possible paths to obtain the full probability for a sequence x:

$$P(x) = \sum_{\pi} P(x, \pi) = \sum_{\pi} \left(a_{0\pi_1} \prod_{i=1}^{L} e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}} \right) \tag{9}$$

- Brute force evaluation of equation (eq 9) is not practical, because the number of possible paths π increases exponentially with the length of the sequence.
- One possible solution is to use equation (eq 6) evaluated at the most probable path π^* (see previous slides) as an approximation to $P(x)$.
- This assumes that the only path with significant probability is π^*.

The Forward Algorithm

- The full probability of a sequence $P(x)$ can be calculated by the *forward algorithm*.
- It is a similar dynamic programming procedure as the Viterbi algorithm, but the maximisation steps are replaced by sums.
- The variable corresponding to $v_k(i)$ in the Viterbi algorithm is:

$$f_k(i) = P(x_1 \ldots x_i, \pi_i = k), \tag{10}$$

- This is the probability of the observed sequence up to x_i, where $\pi_i = k$.
- The recursion equation is:

$$f_l(i+1) = e_l(x_{i+1}) \sum_k f_k(i) a_{kl} \tag{11}$$

Forward Algorithm

- Suppose the probabilities $f_k(i)$ are known for all states k then one can compute $f_l(i+1)$ as shown here:

Forward Algorithm

Initialization ($i = 0$):	$f_0(0) = 1$, $f_k(0)$ for $k > 0$
Recursion ($i = 1 \ldots L$):	$f_l(i) = e_l(x_i) \sum_k f_k(i-1) a_{kl}$
Termination:	$P(x, \pi^*) = \sum_k f_k(L) a_{k0}$

- Again, dynamic programming allows tabulation of a matrix f.
- To avoid underflow error, perform calculations in log space.

Backward Algorithm

Problem

- How to identify the most probable state k for an observation x_i?
- More generally, we want to know the probability $P(\pi_i = k|x)$ that observation x_i came from state k given the observed sequence.
- $P(\pi_i = k|x)$ is the posterior probability of state k at time i when the emitted sequence is known.

Computing Posterior Probabilities

- First calculate the probability of producing the entire sequence with the i-th symbol being produced by state k:

$$\begin{aligned} P(x, \pi_i = k) &= P(x_i \ldots x_i, \pi_i = k) P(x_{i+i} \ldots x_L | x_1 \ldots x_i, \pi_i = k) \\ &= P(x_i \ldots x_i, \pi_i = k) P(x_{i+i} \ldots x_L | \pi_i = k) \\ &= f_k(i) b_k(i) \end{aligned} \tag{12}$$

- Second row, because everything after k depends only on the state at k.
- The frist term in the 2nd line of the above equation (eq 12) is identical to $f_k(i)$ from equation (eq 10) and the second term is called $b_k(i)$.

Backward Algorithm

Initialization ($i = L$):	$b_k(L) = a_{k0}$ for all k
Recursion ($i = L-1, \ldots, 1$):	$b_k(i) = \sum_l a_{kl} e_l(x_{i+1}) b_l(i+1)$
Termination:	$P(x) = \sum_l a_{0l} e_l(x_1) b_l(1)$

- From equation (eq 12) the posterior probabilities can be calculated by conditioning where $P(x)$ is obtained by the forward or backward algorithms:

$$P(\pi_i = k | x) = \frac{f_k(i) b_k(i)}{P(x)} \tag{13}$$

Parameter Estimation for HMMs

- The greatest challenge when using HMMs is the definition of the model itself.
- There are two major steps in this process:
  1. Structure design: order of states and their connections.
  2. Assignment of parameter values: transition probabilities a_{kl} and emission probabilities $e_k(b)$.

Estimation when State Paths Are Known

- Examples:
  - HMM for predicting CpG islands with training sequences where islands have been idenified experimentally.
  - HMM for predicting secondary protein structures with training sequences of known structure.
  - HMM for predicting genes with cDNAs as training sequences with known transcript structures.
- When all the state paths are known, the estimation is simple: count the number of times a particular transition (A_{kl}) or emission ($E_k(b)$) is used in the set of training sequences.
- The maximum likelihood estimators are given by:

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}} \qquad e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')} \tag{14}$$

Estimation when Paths Are Unknown: Baum-Welch

- If the paths are unknown, then there is no closed-form equation for the parameter estimation available anymore.
- Solution: iterative procedure for parameter estimation.
- Many algorithms exist for this, but the Baum-Welch iteration algorithm is the most commonly used one.
- Important considerations:
  - If the paths are known, then the transition and emission probabilities can be calculated.
  - If the transition and emission probabilities are known, then the paths can be calculated.

Baum-Welch

- Informal outline of iterative Baum-Welch algorithm:
  1. Estimate the most probable paths for the training sequences using the current values of a_{kl} and $e_k(b)$.
  2. Use equation (eq 14) to estimate new parameters for the as and es.
  3. Repeat steps 1 and 2.
- It can be shown that the overall log likelihood of the model increases with each iteration.
- Hence, the iteration process converges to a local maximum.
- Problem: often there are many local maxima, especially with large HMMs. The local maxima will vary with the starting values of the parameters.

Baum-Welch

- The algorithm calculates A_{kl} and $E_k(b)$ as the expected counts of transitions and emissions given by the training sequences.
- It uses the same forward and backward values as the posterior probability decoding method.
- The probability that a_{kl} is used at position i in sequence x is:

$$P(\pi_i = k, \pi_{i+1} = l | x, \theta) = \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{P(x)} \tag{15}$$

$$\theta = \text{all current parameters in model}$$

- From this, one can derive the expected number of times that a_{kl} is used by summing over all positions and over all training sequences:

$$A_{kl} = \sum_j \frac{1}{P(x^j)} \sum_i f_k^j(i) a_{kl} e_l(x_{i+1}^j) b_l^j(i+1) \tag{16}$$

- $f_k^j(i)$ is the forward variable $f_k(i)$ defined in equation (eq 10) calculated for sequence j, and $b_l^j(i)$ is the corresponding backward variable.

Baum-Welch

- Similarly, we can find the expected number of times that symbol b is emitted in state k

$$E_k(b) = \sum_j \frac{1}{P(x^j)} \sum_{\{i|x_i^j=b\}} f_k^j(i) b_k^j(i) \tag{17}$$

- The inner sum is only over the positions i for which emitted symbol is b.
- New model parameters are calculated as before in equation (eq 14), but this time based on the *expected* number of times, instead of counts.
- Then iterate using the new values of the parameters to obtain new values of As and Es.

Summary of Baum-Welch Algorithm

Initialisation

- Pick arbitrary model parameters.

Recurrence

- Set all A and E variables to zero (or some pseudocount r)
- For each sequence $j = 1...n$:
  - Compute $f_k(i)$ for sequence j using the forward algorithm.
  - Compute $b_k(i)$ for sequence j using the backward algorithm.
  - Add the contribution of sequence j to A (eq 16) and E (eq 17).
- Calculate the new model parameters using (eq 14).
- Calculate the new log likelihood of the model

Termination

- Stop when the change in the log likelihood is less than some predefined threshold or maximum iterations are reached.

Modeling Labelled Sequences: CpG Islands

- So far we predicted labels for unannotated sequences by training two separate models: one for CpG islands and one for non-CpG islands.
- Separate models can become quickly complicated, *e.g.* ambiguous transitions between submodels and when more than two classes are involved.
- More efficient method is a combined model of all the classes.

CpG Island Example

- A class label is assigned for each state: '+' for island and '-' for non-island states.
- Labels $y = y_1, ..., y_L$ for observatios $x = x_1, ..., x_L$.
- y_i is '+' if x_i is part of a CpG island, or '-' if part of non-island.
- Allow only valid paths through the model when calculating the fs and bs.
- Valid path is where state labels and sequence labels are the same, e.g. π_i with label y_i.
- During the forward and backward algorithm this corresponds to setting $f_l(i) = 0$ and $b_l()i = 0$ for all the states l with a label different from y_i.

Modeling Labelled Sequences: CpG Islands

- Forward table for a model with four states each labelled '+' and '-'.

Sequence			x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	...
Labels			–	–	–	+	+	+	+	–	–	–	...
S	1	-	f & b calculated as usual			f & $b = 0$				f & b calculated as usual			
t	2	-											
a	3	-											
t	4	-											
e	5	+	f & $b = 0$			f & b calculated as usual				f & $b = 0$			
s	6	+											
	7	+											
	8	+											

- Each column corresponds to an observation and each row to a state of the model.
- The first ten residues $x_1, ..., x_{10}$ are assumed to be labelled:

  - - - + + + + - - -

Outline

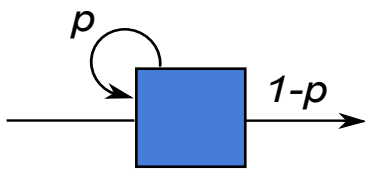
HMM Model Structures

- So far we have only considered fully connected models where all transitions are allowed.
- Although, it is tempting to let the model decide which transitions to use, it almost never works like this in practise.
- This is even the case with abundant training data.
- The problem is not overfitting, but local maxima.
- The less constrained a model, the more severe the local maxima problem.
- In practice the most successful HMM are constructed by carefully deciding which transitions to allow.
- To disable transitions from state k to state l in Baum-Welch algorithm, one can set $a_{kl} = 0$ and never change it in the iterative process.
- In general one should choose a model topology based on the understanding of the problem, *e.g.* sequences from CpG island and non-island states.

Duration Model Topologies

- The simplest model is a state with a transition to itself with probability p. The emission probabilities are disregarded.
- Example: model for DNA string where nucleotide distribution does not change.
- After entering the state there is a probability of $1 - p$ of leaving it. The probability of staying in the state for l residues is:

$$P(l \text{ residues}) = (1 - p)p^{l-1} \tag{18}$$

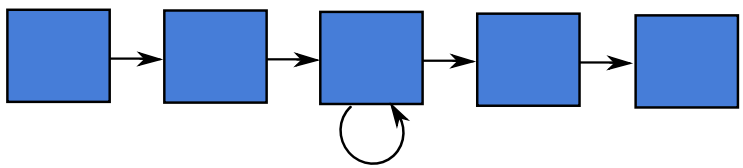


- The emission probabilities are disregarded in this example.
- This exponentially decaying distribution on length, called geometric distribution, is not always appropriate, especially when the length distribution is important.

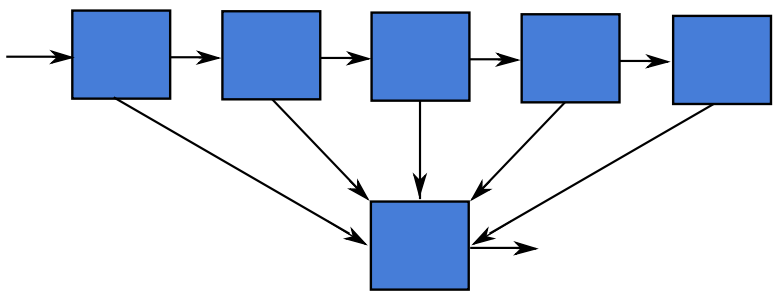
Duration Model Topologies

- More complex length distributions can be modeled by introducing several states with the same distribution between residues and transitions.
- For example, the following model will give a sequence of a minimum length of 5 residues and an exponentially decaying distribution of longer sequences.

$$P(l \text{ residues}) = (1 - p)p^{l-1} \tag{19}$$

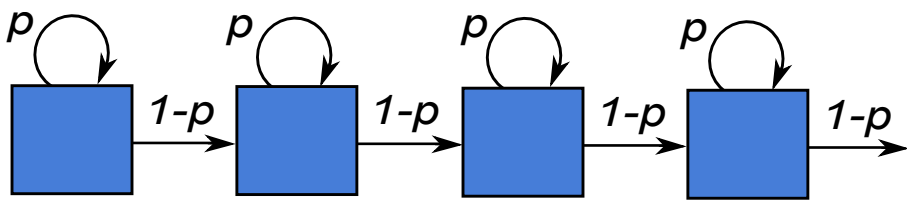


- This structure can model any distribution of lengths between 2-6:



Duration Model Topologies

- A more subtle way of obtaining a non-geometrical length distribution is to use an array of n states, each with a transition to itself of probability p and a transition to the next of probability $1 - p$:



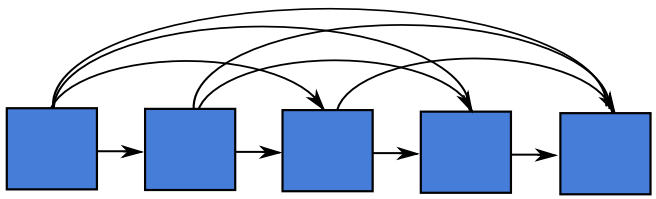
- The shortest sequence that can be modeled by this topology is n.
- For any given path of length l through the model, the probability of all its transitions is $p^{l-n}(1-p)^n$ (emission probabilities disregarded).
- The number of possible state paths is $\binom{l-1}{n-1}$, and the total probability summed over all possible paths is:

$$P(l) = \binom{l-1}{n-1} p^{l-n}(1-p)^n \tag{20}$$

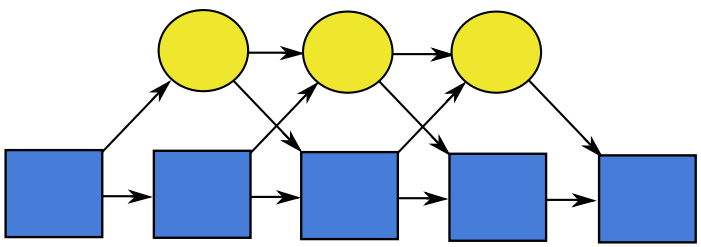
- It follows a negative binomial distribution.

Model Topologies with Silent States

- Silent states do not emit symbols in an HMM. Previous examples of silent states are the begin and end states.
- They are often introduced to reduce the number of possible transitions in an HMM. Example: profile HMMs of multiple alignments (see later).
- For instance, to allow any number of deletions, a chain of states needs to be completely forward connected. Problem: the number of required connections increases exponentially ($O(m^2)$).



- To reduce the number of connection, one can connect all states to a parallel chain of silent states. Number of connection increases by $O(m)$.

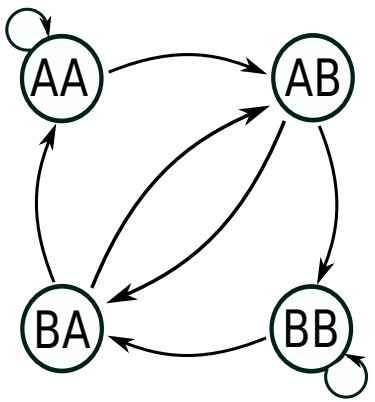


- Because the silent states do not emit any letters, it is possible to get from any 'real' state to any later 'real' state without emitting any letters.

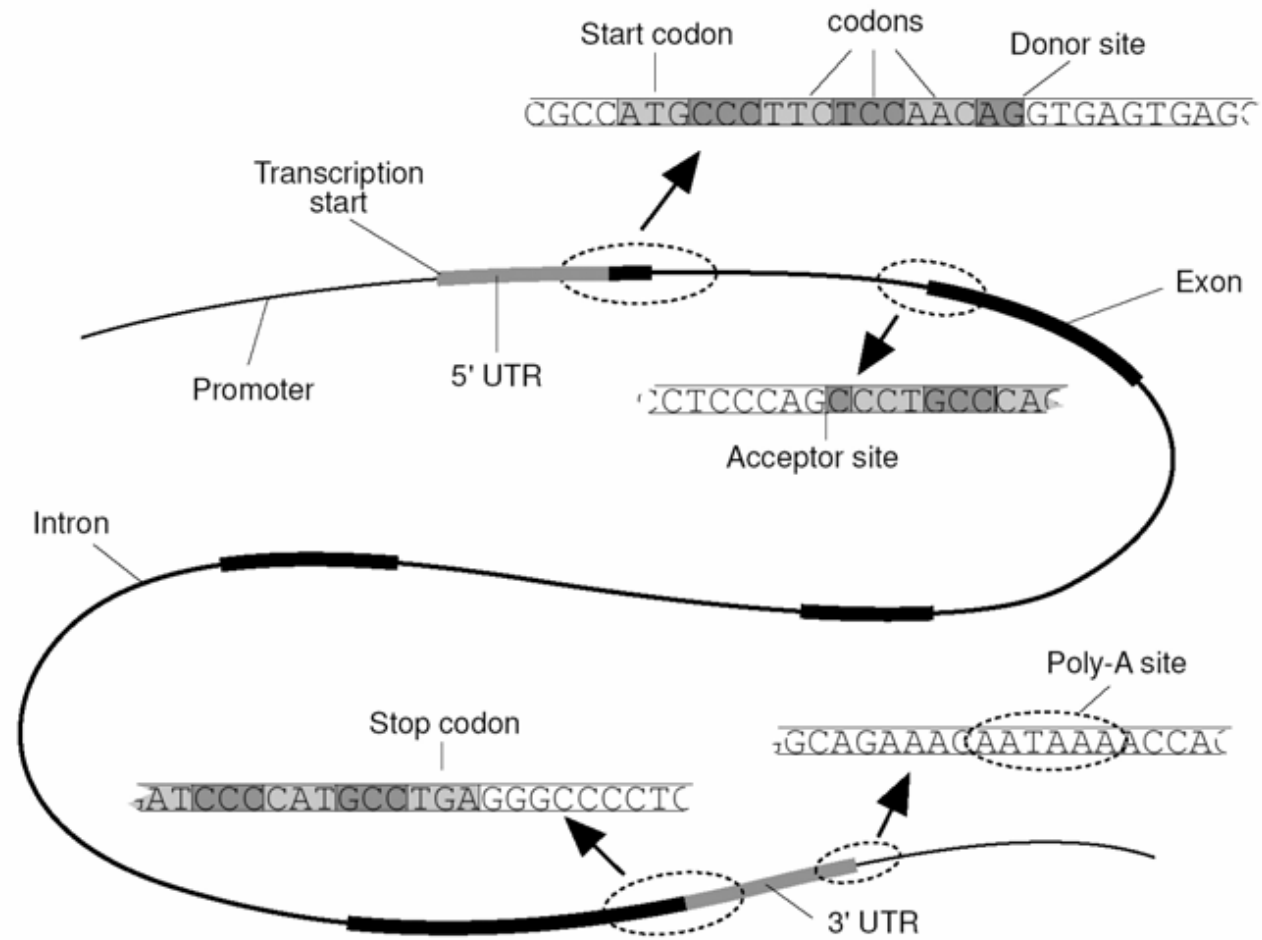
Outline

More Complex Markov Chains

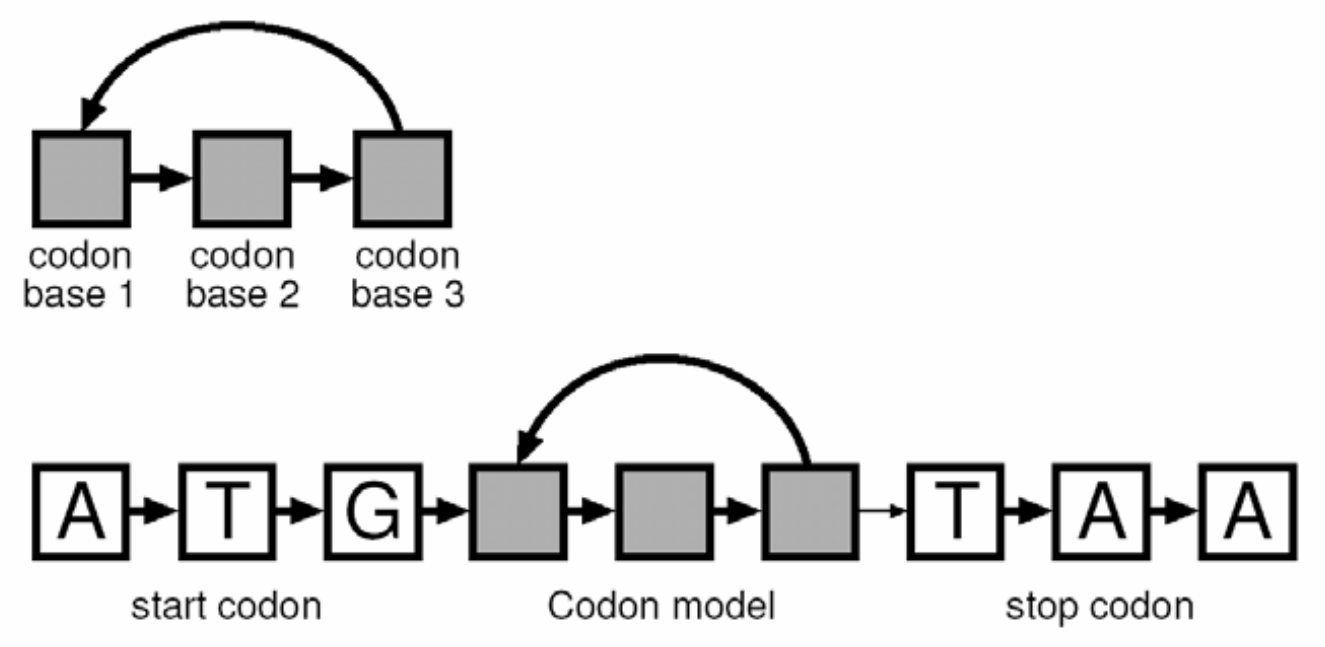
- So far, only first order Markov were considered.
- An nth order Markov chain over an alphabet $\mathcal{A}$ is equivalent to a first order Markov chain over the alphabet $\mathcal{A}^n$ of n elements.
- High order models are treated in the same way as first order models.
- Example: a second order Markov chain for a sequence of letter pairs from an alphabet with only two letters. Here a sequence can be translated to a sequence of pairs. *E.g.*: the sequence ABBAB becomes AB-BB-BA-AB. The corresponding four-state first order Markov chain looks like this:



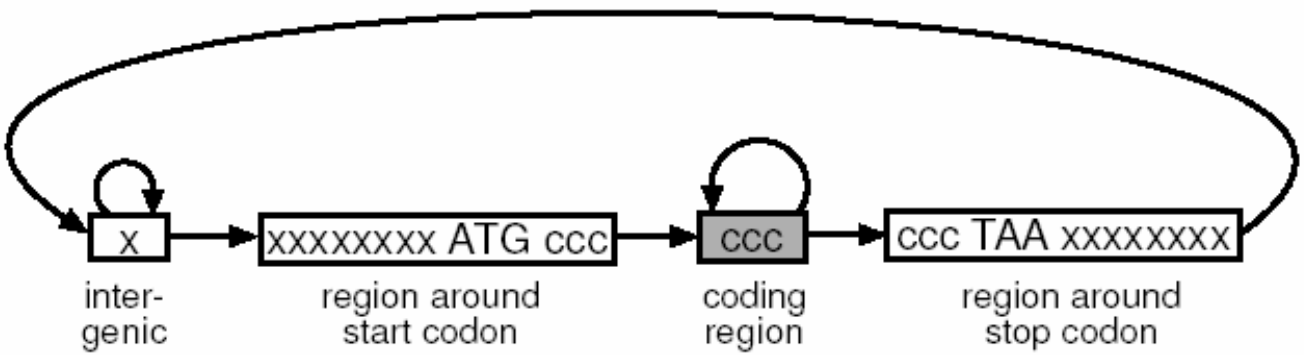
Problem: Gene Prediction



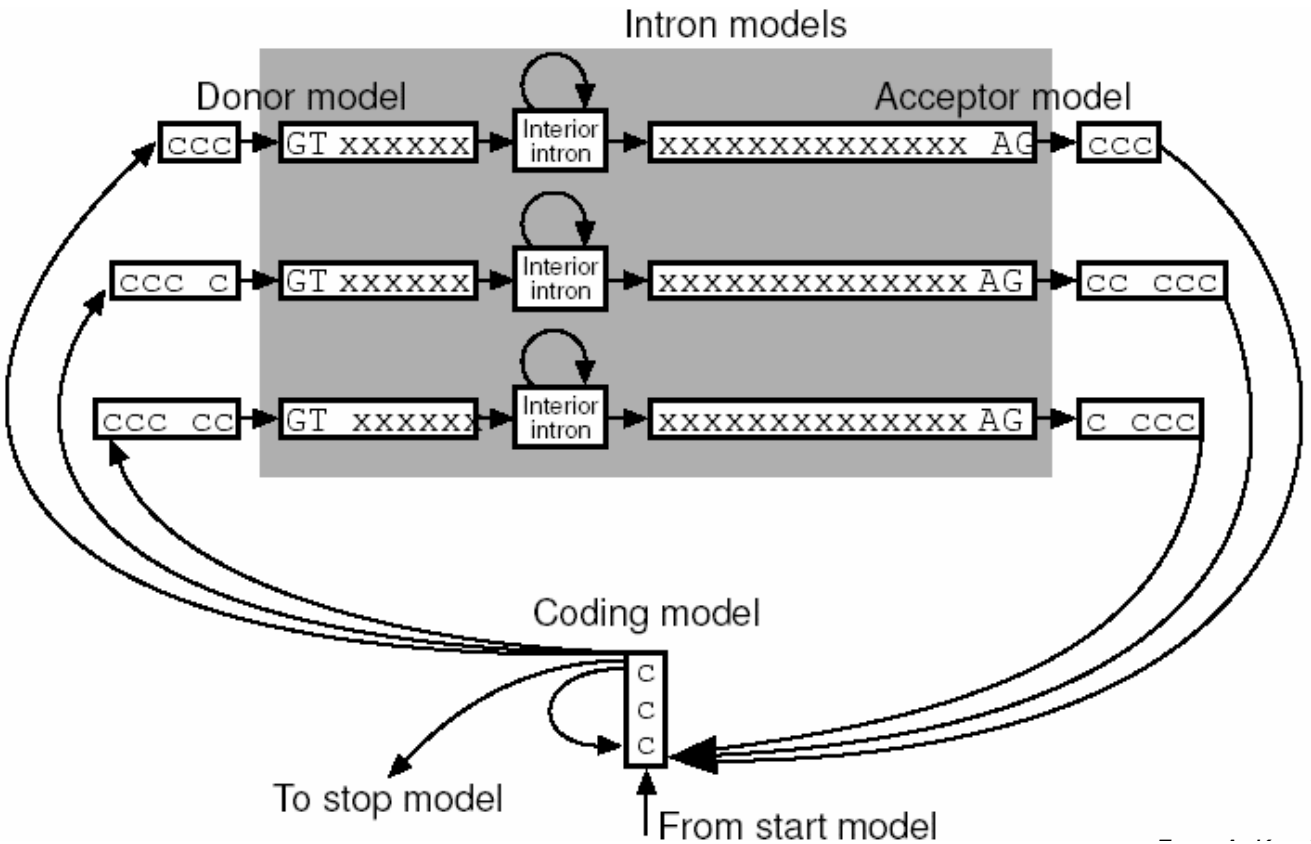
Example: HMM for Coding Sequence



Example: HMM for Genes without Introns



Example: HMM for Introns



Outline

Outline

Background

- The most powerful and insightful sequence analysis methods are based on the analysis of the relationships among sequence family members.
- Sequences in a family have diverged during evolution by separation due to gene duplication in a genome or by speciation in related organisms.
- The proper assignment of sequences to families is an important step for most analysis procedures.
- Family members can be identified by pairwise alignment-based sequence similarity search methods.
- More sensitive approaches focus on statistically conserved features in related sequences.

Background

Example: Partial P450 Multiple Alignment

```
P10632          KNGNFKKSDYFMPFSAGKRICAGEGLARMELFLFLTTILQNFNLKSVDD.LKNLNTT
P11712          EGGNFKKSKYFMPFSAGKRICVGEALAGMELFLFLTSILQNFNLKSLVD.PKNLDTT
P08686          PGKNSRA....LAFGCGARVCLGEPLARLELFVVLTRLLQAFTLLPSGDALPSLQPL
O15528          EGPTP.HPFASLPFGFGKRSCMGRRLAELELQMALAQILTHFEVQP.EPGAAPVRPK
P08684          KNKDNIDPYIYTPFGSGPRNCIGMRFALMNMKLALIRVLQNFSFKPCKETQIPLKLS
consensus       .............F..G.R.C.G...A.......L...L..F..............
```

- Certain positions are more conserved than others.
- Most conserved positions are often the functionally relevant ones.
- Gaps line up among each other, leaving blocks of ungapped sequences.

- When assigning new family members the analysis should focus on conserved residues, because these are the relevant features.
- Question: how can we develop an HMM for modeling multiple alignments?
- Answer: profile HMMs
- Profile HMMs are the most commonly used HMM application in bioinformatics.
- They are very similar to standard non-probabilistic profiles introduced by [Gribscov 1987]

Outline

Ungapped Score Matrices

- Problem: Build a model from a *correct* multiple alignment that can be used to find and score matches to new sequences.
- A probabilistic model for an ungapped multiple alignment can be specified by independent probabilities $e_i(a)$ of observing amino acid a at position i.
- The probability of a new sequence x according to this model is:

$$P(x|M) = \prod_{i=1}^{L} e_i(x_i) \tag{21}$$

x = sequence x

L = length of alignment block

e = will be emission probability

- Usually, we are more interested in the ratio of this probability to the probability of x under a random model. To test for membership in the family, the log-odds ratio can be calculated:

$$S = \sum_{i=1}^{L} log \frac{e_i(x_i)}{q_{x_i}} \tag{22}$$

q_{x_i} = frequency of position i

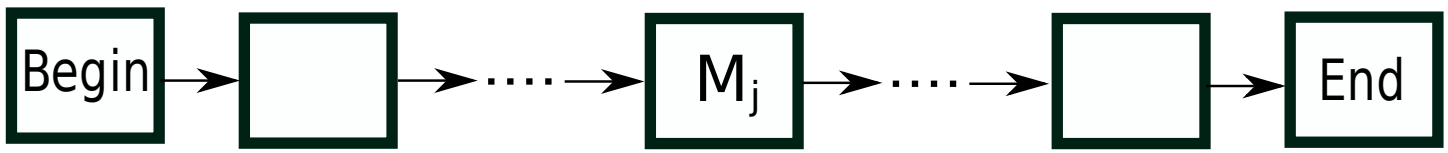
Outline

Adding Insert and Delete States to Obtain Profile HMMs

- The values $log \frac{e_i(a)}{q_{x_i}}$ behave like a score matrix $s(a, b)$, where the second index position is i rather than amino acid b.
- This representation is known as *position specific scoring matrix* (PSSM).
- A PSSM can be used to search for a match in a longer sequence x of length N by evaluating the score S_j for each starting point j in x from 1 to $N - L + 1$, where L is the length of the PSSM.
- Although a PSSM captures conservation information, it is an incomplete representation of all the information in a multiple alignment.
- A model is needed that accounts for gaps in multiple alignments.
- Possible approach: allow gaps at any position in alignment using the same gap score $\gamma(g)$ at each position as introduced for pairwise alignments.
- Problem: the position specific information about gap locations in the alignment is not captured by this model.
- Solution: build an HMM with a repetitive structure of states, but different probabilities in each position.
- This provides a full probabilistic model for the sequences in a family.

Match States

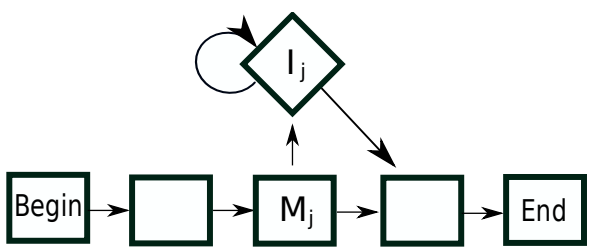
- Start with PSSM which can be considered as a trivial HMM with a series of identical states that we will call match states, separated by probability 1.



- The alignment is trivial here because there are no decisions to make.
- The emission probabilities for the match states are defined as $e_{M_i}(a)$.

Insertions

- Insertions are segments of x that do not match anything in the model.
- To account for insertions, the insert states I_i are introduced. They are used to match insertions after the residue matching the ith column of the multiple alignment.
- The I_i states have emission distribution $e_{I_i}(a)$, but these are set to the background distribution q_a, just as for seeing an unaligned insert residue in a pairwise alignment.
- What is needed are:
  1. Transitions form M_i to I_i.
  2. Loop transition from I_i to itself to allow multi-residue insertions.
  3. Transition back from I_i to M_{i+1}.
- A single insert state is given in the following diagram with the insert state denoted by a diamond.



Insertions

- The log-odds cost of an insert is the sum of the costs of the relevant transitions and emissions.
- Assuming that $e_{Ii}(a) = q_a$ (see above), there is no log-odds contribution from the emission, and the score of a gap of length k is:

$$\log a_{M_j I_j} + \log a_{I_j M_{j+1}} + (k-1)\log a_{I_j I_j} \tag{23}$$

- This corresponds to an affine gap scoring model.

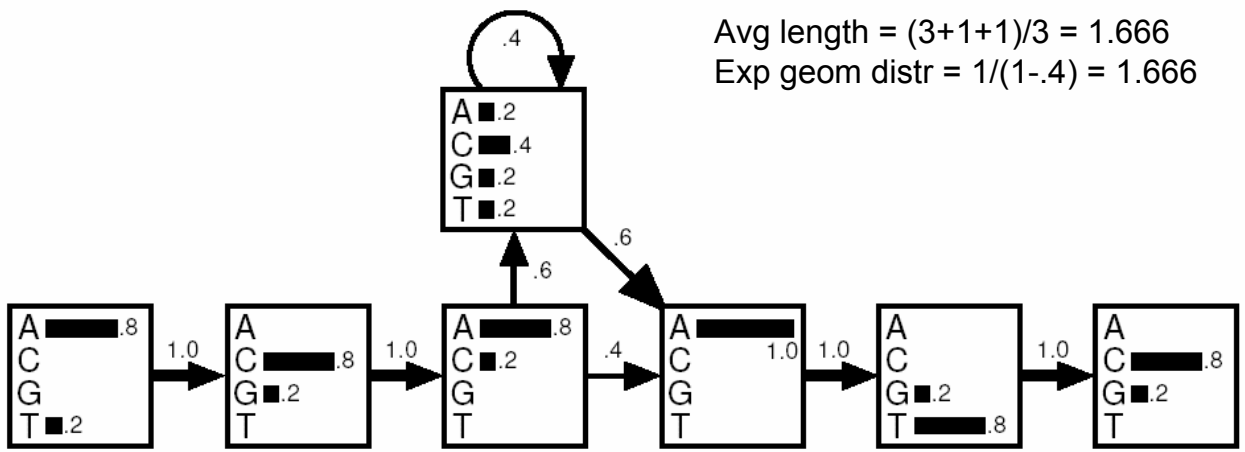
Example for Insertion

Alignment

```
A C A - - - A T G
T C A A C T A T C
A C A C - - A G C
A G A - - - A T C
A C C G - - A T C
```

Rigid pattern

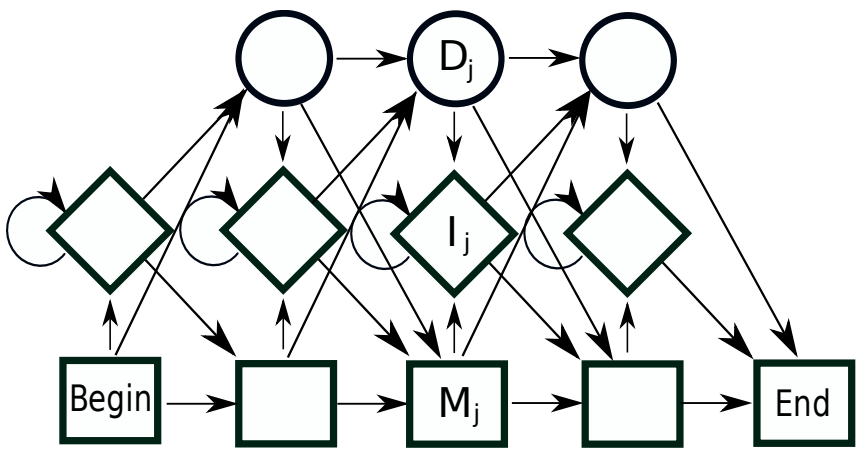
[AT] [CG] [AC] [ACGT]* A [TG] [GC]



Avg length = (3+1+1)/3 = 1.666
Exp geom distr = 1/(1-.4) = 1.666

Deletions

- Deletions are segments of the multiple alignment model that have no matches in x.
- They can be handled by silent states as introduced before, which are called here deletion states Dj:
- Because silent states do not emit any residues, this model structure allows insertions of any length at any position of a sequence.
- Compared to insertions (self replications), the deletion state allows more complex probabilities, due to transitions between deletion initiation, extension and termination states.
- The transition structure of the complete profile HMM with match (squares), insert (diamonds) and delete (circles) states has the following structure. It was first introduced by Haussler and Krogh [Krogh 1994].



Deriving Profile HMMs from Multiple Alignments

- By setting the transition and emission probabilities of the profile HMM structure, one can capture specific information about each position in a multiple alignment of a whole family.
- The goal is to model the consensus sequence of a protein family, and not just the sequence of a specific member.

```
P10632          PFSAGKRICAGEGLARMELFGGLFLTTILQNFNL
P11712          PFSAGKRICVGEALAGMELF..LFLTSILQNFNL
P08686          AFGCGARVCLGEPLARLELF..VVLTRLLQAFTL
P08684          PFGSGPRNCIGMRFALMNMK..LALIRVLQNFSF
O15528          PFGFGKRSCMGRRLAELELQMALGGAQILTHFEV
consensus       .F..G.R.C.G...A.............L..F..
```

- There are a number of ways to derive the required parameter values for a profile HMM from a multiple alignment.
  1. Non-probabilistic profiles
  2. Probabilistic profile HMMs

Non-Probabilistic Profiles

- Related to profile HMMs, but not a probabilistic model.
- PSSM approach from Gribskov et al [Gribscov 1987].
- Scored columns in multiple alignment as follows:
  1. Set the scores for each column to the averages of the substitution scores among all residues in this column. *E.g.* column one in alignment of previous slide: $\frac{4}{5}s(P, col_1) + \frac{1}{5}s(A, col_1)$.
  2. Gap penalties for each column calculated by equation that decreases the cost of the gap by the length of the longest gap spanning this column.
- Criticisms of this approach:
  - The degree of conversations of a residue among the sequences is not well captured by this approach. *E.g.* a conserved cysteine (C) in an alignment of 100 sequences has the same probability distribution as in one with two sequences. In reality the likelihood of a cysteine should increase with the number of observations.
  - Gap probabilities for columns with a single instance are scored similar/identical as columns with many gap instances.

Profile HMM Parameterisation

- Profile HMMs have, like all HMMs, emission and transition probabilities.
- They can model any possible sequence of residues and they define a probability distribution over a given sequence set.
- The length of the model depends on the decision on which columns in an alignment to assign to match states and which to insert states.
- Simple match/insert state assignment rule: columns with more than half of gap characters are modeled by inserts.
- The probability parameters can be estimated according to equation (eq 14) by counting the number of times each transition and emission is used and assigning probabilities according to:

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}} \qquad e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')} \tag{24}$$

k and l = indices over states

a_{kl} and e_k = transition and emission probabilities

A_{kl} and E_k = transition and emission frequencies

Opportunities and Limitations

Differences to non-probabilistic approach

- No previous knowledge about alignments is used, such as empirical substitution matrix.

Limitations

- Only large number of sequences in training alignment will give accurate estimate of transition and emission probabilities.
- Certain transitions and emissions may never occur in training data set. Those would have zero probability, and therefore would never be allowed by the model.

How to avoid zero probabilities?

- Simplest approach is Laplace's rule: add pseudo-count of one to each frequency.

Example: Multiple Alignment

Multiple alignment slice of seven globin sequences

```
GLOBIN1            ..VGA..HAGAY..
GLOBIN2            ..V....NVDEV..
GLOBIN3            ..VEA..DVAGH..
GLOBIN4            ..VKG.....D..
GLOBIN5            ..VYS..TYETS..
GLOBIN6            ..FNA..NIPKH..
GLOBIN7            ..IAGADNGAGN..
consensus
```

Fig 72: Orange parts will be treated as matches in profile HMM of Fig 6.

Parameterisation of profile HMM

Emission probabilities for first column:

$e_{M1}(V) = 6/27$, $e_{M1}(I) = 2/27$, $e_{M1}(F) = 2/27$
$e_{M1}(a) = 1/27$ for all residue types a other than V, I, F

Transition probabilities following first column:

$a_{M_1 M_2} = 7/10$, $a_{M_1 D_2} = 2/10$, and $a_{M_1 I_1} = 1/10$

Following column 1 there are six sequences with transitions from match to match and one transition to a delete state (Globin2).

Profile HMM for Multiple Alignment

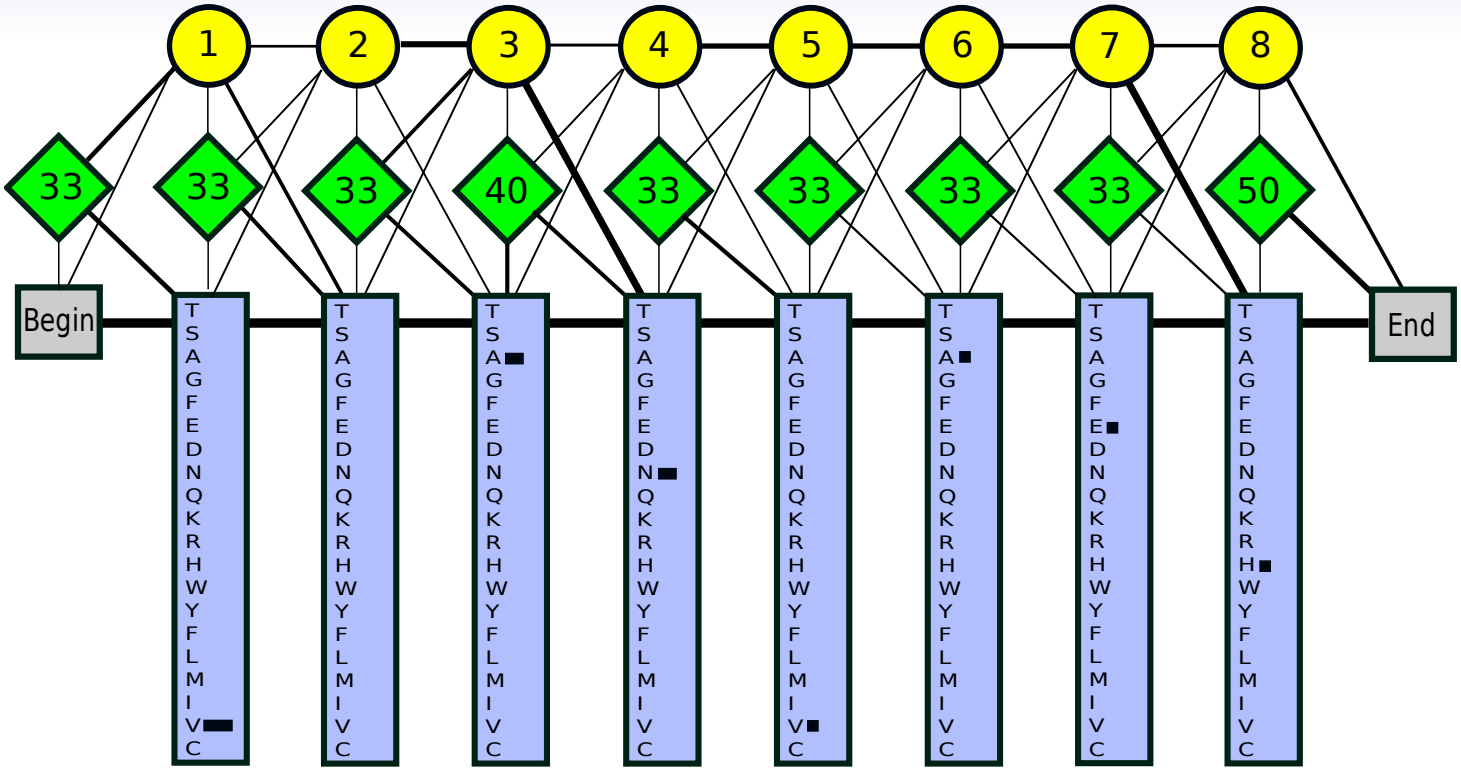


Figure 6: complete parameters for profile HMM of globin alignment in Fig 72.

- Emission probabilities: bars next to amino acids; transition probabilities: thickness of lines.
- $I \rightarrow I$ transition probabilities times 100 are shown in insert states. They are 33 for states with 3 and 50 for states (last position) with only 2 transitions.

Searching with Profile HMMs

- The main purpose of profile HMMs is to detect potential membership of a sequence to a sequence family.
- There are many methods to align and score sequence matches to an HMM. For instance:
  - The Viterbi equation can give the most probable alignment π^* of a sequence x together with its probability $P(x, \pi^*|M)$.
  - The forward equations can provide the full probability of x summed over all possible paths $P(x|M)$.
- In all cases, one will consider the log-odds ratio of the resulting probability to the probability of x given the standard random model:

$$P(x|R) = \prod_i q_{x_i} \tag{25}$$

- For more details see [Durbin 1999].

Creating Multiple Alignments with Profile HMMs

- Another important use of profile HMMs is to generate a multiple alignment of a sequence family
- The simplest solution is to take the highest scoring alignment to the HMM. This can be obtained by tracing back the Viterbi variables $V_j^{'}(i)$.

Iterative Family Modeling with Profile HMMs

Obtain profile HMMs from multiple alignments (Training)

- Compute the transition probabilities and amino acid compositions for each match and insert state in a provided multiple alignment.
- The most probable paths through the model for generating each sequence is computed by the Baum-Welch algorithm.

Use profile HMMs for aligning sequences

- The Viterbi algorithm and dynamic programming are used for finding for each unaligned sequence the most probable alignment (path through profile HMM)
- The collection of paths with match, insert and delete states provides a multiple alignment for the sequences.
- Repeat all steps to improve HMM and multiple alignment iteratively.

Profile HMM database searching (both directions)

- Profile HMMs can be used to search sequence databases.
- A length corrected Z-score is calculated that provides the number of standard deviations a query sequence scores above a background distribution obtained from sequences that do not belong to this family. The score indicates how well a sequence fits the model and whether it is related to the sequences used for training the model.

Outline

Software Packages for HMM-Based Family Modeling

- Sequence Alignment and Modeling System (SAM): https://compbio.soe.ucsc.edu/sam.html
  - Developed by R. Hughey and A. Krogh (UCSC)
  - Utilities: profile HMM generation, alignment and searching.
  - Advantages: can incorporate structural information about proteins.
  - Download Manual
- HMMER2 and HMMER3: http://hmmer.janelia.org
  - Developed by S. Eddie
  - Utilities: profile HMM generation, alignment and searching.
  - Advantages: associated with PFAM database of profile HMMs.
  - Download Manual

Example: HMMER2

hmmbuild: generates HMM profile from alignment

```
hmmbuild my.hmm myalignment.msf
# for global alignments with default setting and local alignment with -f
hmmcalibrate myhmm
# calibrates HMM from hmmbuild for database search with hmmsearch
```

hmmsearch: search a sequence database with an HMM

```
hmmsearch -E 0.1 my.hmm mypepDB > myoutput
# -E: e-value cutoff; -A0: turns off alignment output
```

hmmpfam: search with a query sequence an HMM database

```
hmmpfam -E 0.1 --acc myhmmDB mypep
# –acc: reports PF accession numbers instead of domain names
```

hmmalign: align sequences to an HMM profile

```
hmmalign -o myalignment.stock my.hmm mypep.fasta
hmmalign --withali myalignment -o myoutput.stock my.hmm
mypep.fasta
# aligns sequences to an existing alignment.
```

Example: HMMER3

What's new in HMMER3 [Eddy 2008]?

Performance

Sensitivity: much higher than HMMER2 due to more accurate statistical model for sequence scoring (*e.g.* e-values more accurate).

Time performance: about 100x faster than HMMER2 by using a heuristic search approach.

Utility

phmmer: searches a sequence against a sequence database (not in HMMER2)

jackhmmer: as phmmer but iterative model building and population of family (not in HMMER2)

hmmbuild: generates HMM profile from alignment

hmmsearch: search a sequence database with an HMM

hmmscan: search with a query sequence an HMM database (hmmpfam counterpart in HMMER2)

hmmalign: align sequences to an HMM profile

Example: Profile HMM for P450 Family

Fetch P450 model from Pfam-A HMM database on HPCC cluster

```
$ module load hmmer/3.1b2
$ hmmfetch /srv/projects/db/pfam/2017-06-11-Pfam31.0/Pfam-A.hmm p450 | less

$ hmmscan -E 0.1 --acc --domtblout output.pfam

/srv/projects/db/pfam/2017-06-11-Pfam31.0/Pfam-A.hmm p450.fasta
```

Locate the following components in the model

- Match emission line. Note: probability parameters are given as negative natural log probabilities.
- Insert emission line
- State transition line
- Invariant cysteine in position 412. Compare with HMM logo here.
- For details see section 8 (page 106) of HMMER3 manual here.

Outline

Summary

General Features

- Probabilistic alignment and family modeling approach.

Advantages

- Useful in many areas of computational biology.
- Efficient model of sequence family.
- Model can be used for domain mapping, sequence searching and generation of high-quality multiple alignments.

Disadvantages

- Typically, $>$20 sequences are required to build reliable profile HMMs.
- Only practical for domain-focused protein family analyses.

Outline

References

Durbin R, Eddy S, Krogh A and Mitchison G (1999) Biological Sequence Analysis - Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press. Chapters 3-5.

Eddy, S (2008) A probabilistic model of local sequence alignment that simplifies statistical significance estimation. PLoS Comput Biol, 4: e1000069.

Gribskov M, McLachlan AD, Eisenberg D(1987) Profile analysis: detection of distantly related proteins. Proc Natl Acad Sci U S A, 84: 4355-4358.
URL `http://www.hubmed.org/display.cgi?uids=3474607`

Krogh A (1998) An introduction into hidden Markov models for biological sequences. in *Computational Biology: Pattern Analysis and Machine Learning Methods.* Elsevier, Capter 4.

Krogh, A, Brown, M, Mian, I S, Sjolander, K, Haussler, D A (1994) Hidden Markov models in computational biology. Applications to protein modeling. J Mol Biol, 235: 1501-1531.
URL `http://www.hubmed.org/display.cgi?uids=8107089`.