Hidden Markov Models

P.Antal

antal@mit.bme.hu

BME

Overview

1. Motivations
2. HMMs
3. Inference
4. pFam
5. Learning
6. Genscan

Motivations, solutions

Position specific scoring of substitutions, inserts, deletions.

Hidden Markov Models

Stochastic Finite State Automaton

Stochastic grammars

Dynamic Bayesian Networks

Chomsky hierarchy of grammars (*:right/left, with/without ϵ;**:nondecreasing):

Grammar	Rule	Automaton	Parsing	Language
regular*	$W \longrightarrow aW$	FSA	linear	a reg.expressio
context-free	$W \longrightarrow \beta$	push-down	polynomial	palindromes
context-sensitive**	$\alpha_1 W \alpha_2 \longrightarrow \alpha_1 \beta \alpha_2$	linear bounded	exponential	copies
unrestricted		Turing machine (TM)	semidecidable	$KB - FOL$
-	-			halting TMs

HMM: definition

Markov chain models for sequence x (modeling sequence by states $s \in S$):

$$p(x) = \prod_{i=1}^{L} p(x_i|x_{i-1}) = p(x_1) \prod_{i=2}^{L} a_{x_{i-1},x_i} \tag{1}$$

Compare Bayes factors of different Markov chain models of DNA: homogeneous M_h vs inhomogenous-by-period-3 M_{i_3}

$$\frac{p(x|M_{i_3})}{p(x|M_h)} \tag{2}$$

Now fuse them into a single model $\Rightarrow$ hidden state
Hidden Markov Models (definitions/notations following DEKM)

1. π denotes a state sequence (of a Markov chain), π_i is the ith state
2. a_{kl} the transition probabilities $p(\pi_i = l|\pi_{i-1} = k)$ in the MC (extra state 0 for start/end)
3. $e_k(b)$ are the emission probabilities $p(x_i = b|\pi_i = k)$

Note, stochastic finite state automations/regular grammars, later we discuss the application of stochastic context-free grammars (SCFG) for RNA (3-D structure,..palindromes!).

Inferences in HMMs

Note $|\pi| = \mathcal{O}(|S|^L)$

-,L $p(x,\pi) = a_{0\pi_l} \prod_{i=1}^{L} e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}}$

?,L ”decoding": $\pi^* = \arg\max_\pi p(x,\pi)$

?,L sequence probability:$p(x) = \sum_\pi p(x,\pi)$ (or $p(x|M)$ ”model likelihood" or filtering)

?,L smoothing/posterior decoding:$p(\pi_i = k|x)$

?,OK? parametric inference (training/parameteresation)

?,OK? structural inference (model selection)

HMM: Viterbi algorithm

Goal: "decoding": $\pi^* = \arg\max_\pi p(x,\pi)$

Note: "best joint-state-sequence explanation"$\neq$ "joint sequence of best-state-explanations"

Inductive idea: extend most probable paths with length i to i+1

$v_k(i)$ denotes the probability of the most probable path ending in state k with observation i

Then

$$v_l(i+1) = e_l(x_{i+1}) \max_k (v_k(i) a_{kl}) \quad (3)$$

```
Require: HMM,x
Ensure: $\pi^* = \arg\max_\pi p(x,\pi)$
  Ini: (i=0): $v_0(0) = 1$,$v_k(0) = 0$ for 0<k
  for $i = 1$ to $L$ do
    $v_l(i) = e_l(x_i) \max_k(v_k(i-1)a_{kl})$
    $ptr_i(l) = \arg\max_k(v_k(i-1)a_{kl})$
  End: $p(x,\pi^*) = \max_k(v_k(L)a_{k0})$, $\pi^*_L = \arg\max_k(v_k(L)a_{k0})$
  for $i = L$ to $1$ do {Traceback}
    $\pi^*_{i-1} = ptr_i(\pi^*_i)$
```

Note, small probabilities may cause positive underflow (length can be up to 10^3 <)=> log

Note, $\pi^* = \arg\max_\pi p(x,\pi) = \arg\max_\pi p(\pi|x)$

HMM: forward algorithm

Goal: sequence probability:$p(x) = \sum_\pi p(x,\pi)$ (or $p(x|M)$ "model likelihood" or filtering)
Approximation: $p(x) = \sum_\pi p(x,\pi) \approx p(x,\pi^*) = a_{0\pi^*_l} \prod_{i=1}^{L} e_{\pi^*_i}(x_i) a_{\pi^*_i \pi^*_{i+1}}$ (π^* by Viterbi)
Inductive idea(dynamic programming): extend the probability of generating observations $x_{1:i}$ being in state k at i to i+1
By introducing $f_k(i) = p(x_{1:i}, \pi_i = k)$, we can proceed

$$f_l(i+1) = e_l(x_{i+1}) \sum_k (f_k(i) a_{kl}) \tag{4}$$

```
Require: HMM M,x
Ensure: $p(x|M)$
   Ini: (i=0): $f_0(0) = 1$,$f_k(0) = 0$ for 0<k
   for $i = 1$ to $L$ do
      $f_l(i) = e_l(x_i) \sum_k (f_k(i-1) a_{kl})$
   End: $p(x|M) = \sum_k (f_k(L) a_{k0})$
```

Note, we have to sum small probabilities! => log transformation is not enough, scaling methods..

HMM: backward algorithm

Goal: smoothing/posterior decoding $p(\pi_i = k|x)$

Idea: $p(\pi_i = k|x) = \frac{p(\pi_i = k, x)}{p(x)}$ ($p(x)$ can be computed by the forward algorithm)

$p(\pi_i = k, x) = p(\pi_i = k, x_{1:i})p(x_{i+1:L}|\pi_i = k, x_{1:i}) = f_k(i)\underbrace{p(x_{i+1:L}|\pi_i = k)}_{b_k(i)}$

```
**Ensure:** $b_k(i) = p(x_{i+1:L}|\pi_i = k)$
  Ini: (i=L): $b_k(L) = a_{k0}$ for all k
  **for** $i = L - 1$ to $1$ **do**
    $b_k(i) = \sum_l a_{kl}e_l(x_{i+1})b_l(i+1)$
  End: $p(x|M) = \sum_l a_{0l}e_l(x_1)b_l(1)$
```

Note, conditionally most probable state at i $\neq$ state in most probable explanation at i.
Inference about properties of states. Continuous case: $T(s) : S \to \mathcal{R}$, then finding regions with average mean above a threshold

$$T_{i:i+l} = \frac{1}{l}\sum_{j=i}^{j=i+l}\sum_k T(k)p(\pi_j = k|x) \quad (5)$$

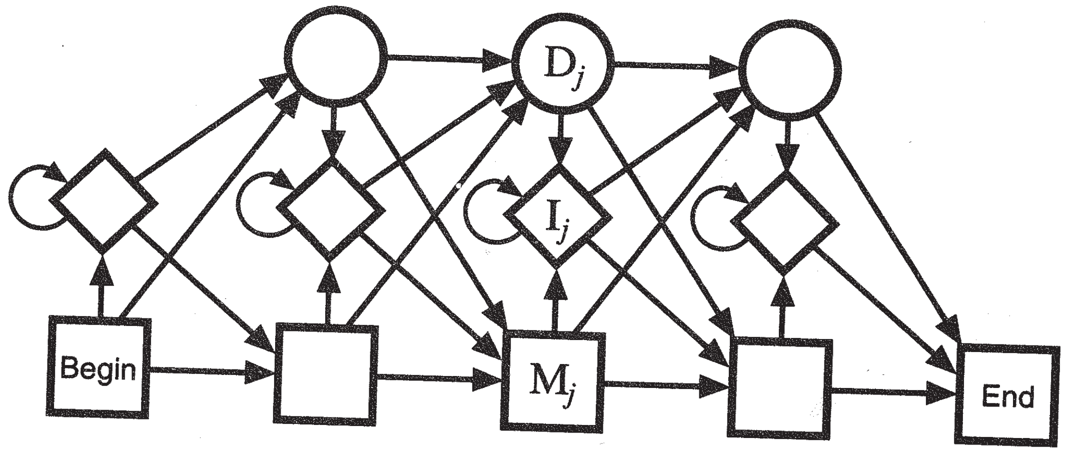
Discrete case: $1(s)$ is an indicator function of an interesting property... (why do not we build an MC/HMM over this state space?=> not first order, not available, not economic,...)

The ”profile" HMMs (pHMMs)

Define a structure (allowed transitions) over states with cardinality n. Note, $\mathcal{O}(n^2)$ parameters can be reduced to linear...)
Substitutions: match states (boxes). Note, level 1 implements already a position specific scoring.
Inserts: insert states (diamonds). Note that length distribution of inserts follows a geometric distribution with parameter p of probability of stay (mean $p/(1-p)$ and variance $p/(1-p)^2$).
Deletes: transitions ”jumping" over match states. Problem: high number of parameters. Solution: further parametric restriction over transition probabilities using silent delete states (circles). Note the possible reduction of $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ representing a position specific gap length penalty or even to 1 representing a gap length penalty.
Note that delete states are so called silent/null states without emission. If there are no loops as in pHMMs =>emulate their effect in Viterbi/forward/backward algs treating separately the probability of transitions without emissions, e.g. accumulating upward $f_l(i+1) += \sum_k f_k(i+1)a_{kl}$ through silent states $k < l$.

The ”profile" HMMs (pHMMs) II.

The profile HMM.



Usage: 1, exploration/visualization of a sequence family 2, deciding membership (for transferring annotations about functionality/structure) 3, (the most probable) multiple alignment
Application: see Pfam.

The probability of alignment

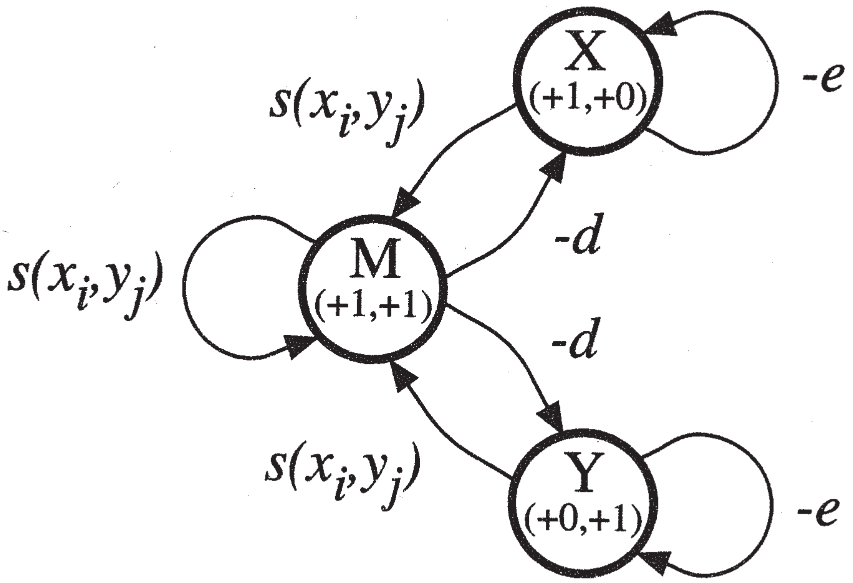
Earlier we see that without indels the pairwise model $p(x_i, y_j|M)$ and independent model R $q(x_i)$ allows the use of likelihood ratio/Bayes factor…
Goal: not PSS, but with indels a full probabilistic approach to alignment.
Imagine a simple global pairwise alignment problem with linear gap penalty: given $p(x_i, y_j)$, q_i and δ probabilities for matched symbols x_i, y_j and $q_i\delta$ for a gap-ith symbol pair. Using a log transformation the dynamic programing approach to global pairwise alignment problem gives the most probable alignment, i.e. the most probable path from (0,0) to (m,n). If we change the maximization to summation it gives the total probability of alignment at (m,n). Note the similarity to the Viterbi/forward algorithms.
Now consider the case of affine gap penalty $\gamma(g) = -d - (g-1)e$. The global pairwise alignment algorithm can be rewritten using three states and the formalism of the Viterbi algorithm as follows

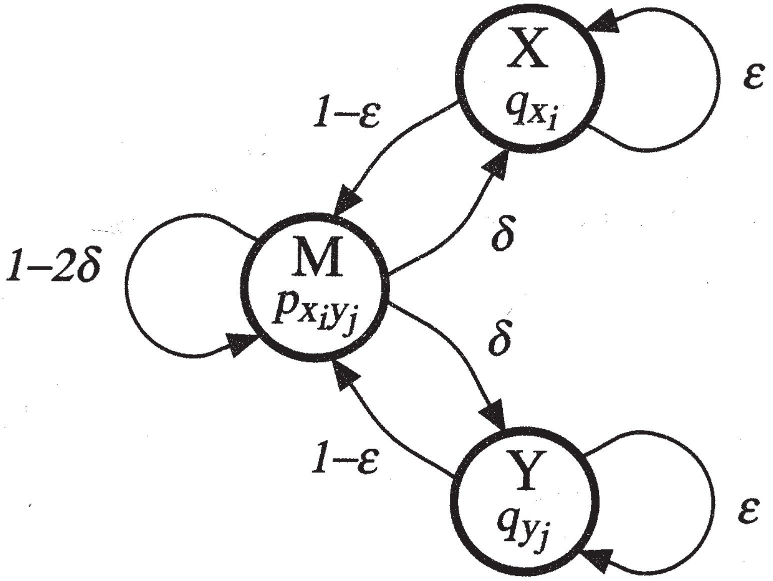
$$\begin{aligned}
V^M(i,j) &= s(x_i, y_j) + max(V^M(i-1, j-), V^X(i-1, j-1), V^Y(i-1, j-1)) && (6)\\
V^X(i,j) &= max(V^M(i-1, j) - d, V^X(i-1, j) - e) && (7)\\
V^Y(i,j) &= max(V^M(i, j-1) - d, V^Y(i, j-1) - e). && (8)
\end{aligned}$$

(Note that gaps cannot be merged, because of the affine gap penalty $\Rightarrow$ no transition between X, Y)

Pair HMMs I

This can be represented as a Finite State Automaton (FSA), which can be extended using the HMM representation to a so called ”pair HMMs" (e.g. with non-uniform indel emission probabilities q_i).





Pair HMMs II

Formally the pair HMMs have ”dual" symbols $Sx\{S \cup \varepsilon\}$/$\{S \cup \varepsilon\}xS$, specifically in the three-state pairwise alignment HMM X,Y and M state emits $Sx\varepsilon$, εxS and SxS symbols.
In the pair HMM using the Viterbi, forward and backward algorithms we can compute

= the most probable explanation $\pi^* = \arg\max_\pi p(\pi|(x,y))$ with its probability $p(x,y,\pi^*)$. This corresponds to the best global alignment from the score based dynamic programming approach.

+ (additionally) the full probability of the alignment $p(x,y) = \sum_\pi p(x,y,\pi)$. Clearly, the forward algorithm is the generalization of the score based algorithm with summations instead of maximizations.

+ (additionally) the posterior that x_i is aligned to y_j denoted with $p(x_i \diamond y_j|x,y)$.

Note that $p(\pi|x,y) = \frac{p(\pi,x,y)}{p(x,y)}$, so combining the results from the Viterbi and forward algorithms, the probability of the most probable alignment π^* can be computed.

Numerical issues I.

Positive underflow and precision in standard C/MATLAB:

1. MAX 1.7976931348623158e+308 Maximum value
2. MIN 2.2250738585072014e-308 Minimum positive value
3. $PREC$ 2.2204460492503131e-016 Smallest such that 1.0+$PREC$!=1.0

Problems:

1. Product of many probabilities $\Rightarrow$ log transformation.
2. Sums of products of many probabilities $\Rightarrow$ log transformation + scaling
3. Ratio of sums of products of many probabilities $\Rightarrow$ log transformation + scaling+normalizing

Numerical issues II.

Product of many probabilities ($\prod_i p_i \approx 10^{-10^6}$) $\Rightarrow$ log transformation $\tilde{p} = \log(p)$.

Sums of products of probabilities $\Rightarrow$ log transformation + scaling (+checking underflow).

Log transformation is not enough, because computationally and underflow ($\tilde{s} = \log(\sum_i \exp(\tilde{p}_i))$).

For example the log transformation of $s = \sum_i p_i$ can be rewritten with scaling as

$$s = p^* \sum_i p_i/p^*, \text{ where; } \mathrm{p}^* = \max_{\mathrm{i}} \mathrm{p_i}, \tag{9}$$

so

$$\tilde{s} = \tilde{p}^* + \log(\sum_i \exp(\tilde{p}_i - \tilde{p}^*)). \tag{10}$$

Iteratively, to compute $\log(s_{t+1}) = \log(s_t + p)$ from $\log(s_t)$ and $\log(p)$, assume $p < s_t$, so

$$\log(s_{t+1}) = \tilde{s}_t + \log(1 + \exp(\tilde{p} - \tilde{s}_t)) \text{ (still check for underflow!)}. \tag{11}$$

Ratio of sums of products of many probabilities $\Rightarrow \log(\frac{\sum_i p_i}{\sum_i q_i}) = \log(\frac{p^*}{q^*}\frac{\sum_i p_i/p^*}{\sum_i q_i/q^*})\ldots$

Useful inequalities/approximations:

$\log(x) \leq x - 1$

$e^{-\frac{x}{1-x}} \leq 1 - x \leq e^{-x}$

HMM learning

Assume n independent/exhangeable sequences $x^{(1)}, \ldots, x^{(n)}$

$$p(x^{(1)}, \ldots, x^{(n)}|\theta) = \prod_{i=1}^{n} p(x^{(i)}|\theta) \tag{12}$$

Note, here θ corresponds to a simplified (descriptive) model class (HMMs) relying on the ”molecular clock hypothesis, and not to the more general (generative/biologically inspired) model class of phylogenetic trees. Furthermore the sequences, in fact, can be (weakly?)dependent through the common evolutionary tree. . . .

1. structure known, state sequences are known: ML parameter computation from counts
2. structure known, state sequences are unknown
   (a) manual/heuristic matching: ML parameter computation from counts
   (b) : Viterbi training: iterative ”multiple alignment-ML parameter computation from counts"
   (c) : Baum-Welch training: iterative computation of mean counts and improved parameters from mean counts (EM-based)
3. structure unknown, state is unknown

Estimation using known state sequences

Recall relative frequency is a maximum likelihood estimator in multinomial sampling. Assume $i = 1, \ldots K$ outcomes assuming multinomial sampling with parameters $\theta = \{\theta_i\}$ and observed occurrencies $n = \{n_i\}$ ($N = \sum_i n_i$). Then

$$\log \frac{p(n|\theta^{ML})}{p(n|\theta)} = \log \frac{\prod_i (\theta_i^{ML})^{n_i}}{\prod_i (\theta_i)^{n_i}} = \sum_i n_i \log \frac{\theta_i^{ML}}{\theta_i} = N \sum_i \theta_i^{ML} \log \frac{\theta_i^{ML}}{\theta_i} > 0 \quad (13)$$

because $0 < KL(\theta^{ML}||\theta)$

$$-KL(p||q) = \sum_i p_i \log(q_i/p_i) \leq \sum_i p_i((q_i/p_i) - 1) = 0 \quad (14)$$

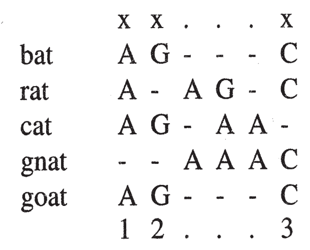
using $\log(x) \leq x - 1$.
Thus using the counts of state transitions A_{kl} and emissions $E_k(b)$

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}} \text{ and } \mathrm{e_k(b)} = \frac{\mathrm{E_k(b)}}{\sum_{\mathrm{b'}} \mathrm{E_k(b')}} \quad (15)$$

So called *pseudocounts* to avoid imprecise estimates (e.g. divison by 0) and *prior counts* to incorporate bias/expertise.
$\Rightarrow A'_{kl} = A'_{kl} + r_{kl} \; E'_k(b) = E_k(b) + r_k(b)$

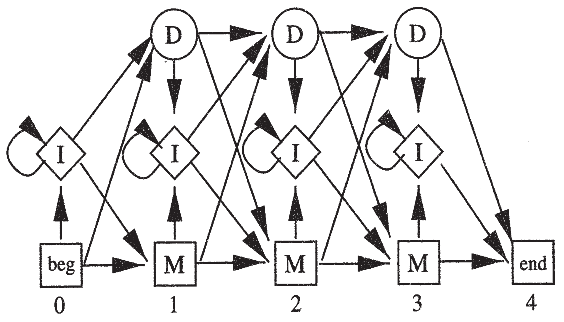
pHMM parameter learning: heuristic

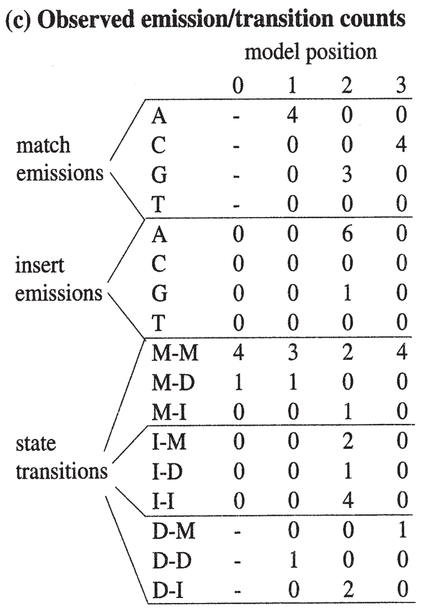
Assume an external (manual,biologically inspired) multiple alignment for sequences $x^{(1)}, \ldots, x^{(n)}$ (by evaluating the characteristics of substituted amino acids w.r.t. the secondary, tertiary structure and also considering homology, phylogenetic aspects, i.e. by adopting a system biology approach to the evolution of funtional/structural entities.
Note, that for a profile HMM (pHMM) the marking of columns with match or insert labels $M_0, I_0^+, \ldots, M_i, I_i^+$ determines the state sequence ($M_i \rightarrow \{m_i, d_i\}$,$I_i \rightarrow \{i_i\}$).
Basic profile HMM parameterisation: majority-based match/insert marking (2^L for length L)



```
        x  x  .  .  .  x
bat     A  G  -  -  -  C
rat     A  -  A  G  -  C
cat     A  G  -  A  A  -
gnat    -  -  A  A  A  C
goat    A  G  -  -  -  C
        1  2  .  .  .  3
```

(b) Profile-HMM architecture:





(c) Observed emission/transition counts

		model position 0	1	2	3
match emissions	A	-	4	0	0
	C	-	0	0	4
	G	-	0	3	0
	T	-	0	0	0
insert emissions	A	0	0	6	0
	C	0	0	0	0
	G	0	0	1	0
	T	0	0	0	0
state transitions	M-M	4	3	2	4
	M-D	1	1	0	0
	M-I	0	0	1	0
	I-M	0	0	2	0
	I-D	0	0	1	0
	I-I	0	0	4	0
	D-M	-	0	0	1
	D-D	-	1	0	0
	D-I	-	0	2	0

A MAP approach: compute the MAP probability of column i is a match.

HMM parameter learning: Viterbi

Idea: using the actual parameters compute the most probable paths $\pi^*(x^{(1)}), \dots, \pi^*(x^{(n)})$ for the sequences and select ML parameters based on these.

```
Require: HMM structure, $x^{(1)}, \dots, x^{(n)}$
Ensure: $\approx \arg\max_\theta p(x^{(1)}, \dots, x^{(n)} | \theta, \pi^*(x^{(1)}, \theta), \dots, \pi^*(x^{(n)}, \theta))$
  Ini: draw random model parameters $\theta_0$ (e.g. from Dirichlet)
  repeat
    set A and E values to their pseudocount
    for $i = 1$ to $n$ do
      Compute $\pi^*(x^{(i)})$ using $\theta_t$ with the Viterbi algorithm
    Set new ML parameters $\theta_{t+1}$ based on current counts A and E from
    $x^{(1)}, \dots, x^{(n)}, \pi^*(x^{(1)}), \dots, \pi^*(x^{(n)})$
    Compute model likelihood $L_{t+1} = p(x^{(1)}, \dots, x^{(n)} | \theta_{t+1})$
  until NoImprovement($L_{t+1}$,$L_t$,t)
```

Note, that this finds a θ maximizing $p(x^{(1)}, \dots, x^{(n)} | \theta, \pi^*(x^{(1)}, \theta), \dots, \pi^*(x^{(n)}, \theta))$ and not the original goal $p(x^{(1)}, \dots, x^{(n)} | \theta)$.

HMM parameter learning: Baum-Welch

Idea: compute the expected number of transitions/emissions A_t,E_t based on θ_t, then update to θ_{t+1} based on A_t,$E_t\ldots$

The probability of $k \to l$ transition at position i in sequence x is

$$p(\pi_i = k, \pi_{i+1} = l|x) \tag{16}$$

$$= \frac{p(\overbrace{x_1,\ldots,x_i,\pi_i = k}^{f_k(i)},x_{i+1},\overbrace{\pi_{i+1} = l, x_{i+2},\ldots,x_L}^{b_l(i+1)})}{p(x)} = \frac{f_k(i)a_{kl}e_l(x_{i+1})b_l(i+1)}{p(x)} \tag{17}$$

The mean of the number of this transition and the mean of the number of emission b from state k is

$$A_{kl} = \sum_j \frac{1}{p(x^{(j)})} \sum_i f_k^{(j)}(i)a_{kl}e_l(x_{i+1}^{(j)})b_l^{(j)}(i+1) \tag{18}$$

$$E_k(b) = \sum_j \frac{1}{p(x^{(j)})} \sum_{i|x_i^{(j)}=b} f_k^{(j)}(i)b_k^{(j)}(i), \tag{19}$$

Apply the same iterative algorithm as in Viterbi traning ($\theta_t \to A_t, E_t \to \theta_{t+1} \to \ldots$)

Why does it converge? Baum-Welch is an Expectation-Maximization algorithm

Derivation of Baum-Welch I: EM

Goal: from observed x, missing π: $\theta^* = \arg\max_\theta \log(p(x|\theta))$
Idea: improve "expected data log-likelihood" $Q(\theta|\theta_t) = \sum_\pi p(\pi|x,\theta_t)\log(p(x,\pi|\theta))$
Using $p(x,\pi|\theta) = p(\pi|x,\theta)p(x|\theta)$ we can write that

$$\log(p(x|\theta)) = \log(p(x,\pi|\theta)) - \log(p(\pi|x,\theta)) \tag{20}$$

Multiplying with $p(\pi|x,\theta_t)$ and summing over π gives

$$\log(p(x|\theta)) = \underbrace{\sum_\pi p(\pi|x,\theta_t)\log(p(x,\pi|\theta))}_{Q(\theta|\theta_t)} - \sum_\pi p(\pi|x,\theta_t)\log(p(\pi|x,\theta)) \tag{21}$$

We want to increase the likelihood, i.e. want this difference to be positive

$$\log(p(x|\theta)) - \log(p(x|\theta_t)) = Q(\theta|\theta_t) - Q(\theta_t|\theta_t) + \underbrace{\sum_\pi p(\pi|x,\theta_t)\log(\frac{p(\pi|x,\theta_t)}{p(\pi|x,\theta)})}_{KL(p(\pi|x,\theta_t)||p(\pi|x,\theta))} \tag{22}$$

Derivation of Baum-Welch II: EM

Goal: from observed x, missing π: $\theta^* = \arg\max_\theta \log(p(x|\theta))$
Because $0 \geq KL(p||q)$, so

$$\log(p(x|\theta)) - \log(p(x|\theta_t)) \geq Q(\theta|\theta_t) - Q(\theta_t|\theta_t). \quad (23)$$

Generalised E-M: if we can select a better θ w.r.t. $Q(\theta|\theta_t)$ then asymptotically it converges to a local or global maximum (note that the target θ has to be continuous).
E-M: select best

$$\theta_{t+1} = \arg\max_\theta Q(\theta|\theta_t) \quad (24)$$

Derivation of Baum-Welch III: EM

The probability of a given path π and observation x is

$$p(x,\pi|\theta) = \prod_{k=1}^{M} \prod_{b} [e_k(b)]^{E_k(b,\pi)} \prod_{k=0}^{M} \prod_{l=1}^{M} a_{kl}^{A_{kl}(\pi)} \tag{25}$$

using this we can rewrite $Q(\theta|\theta_t) = \sum_\pi p(\pi|x,\theta_t) \log(p(x,\pi|\theta))$ as

$$Q(\theta|\theta_t) = \sum_\pi p(\pi|x,\theta_t) \sum_{k=1}^{M} \sum_{b} E_k(b,\pi) \log(e_k(b)) + \sum_{k=0}^{M} \sum_{l=1}^{M} A_{kl}(\pi) \log(a_{kl}) \tag{26}$$

Note that the expected value of A_{kl} and $E_k(b)$over πs for a given x is

$$E_k(b) = \sum_\pi p(\pi|x,\theta_t) E_k(b,\pi) \quad \mathrm{A_{kl}} = \sum_\pi \mathrm{p}(\pi|\mathrm{x},\theta_\mathrm{t}) \mathrm{A_{kl}}(\pi), \tag{27}$$

Doing the sum first over πs gives (also over multiple sequences in the general case)

$$Q(\theta|\theta_t) = \sum_{k=1}^{M} \sum_{b} E_k(b) \log(e_k(b)) + \sum_{k=0}^{M} \sum_{l=1}^{M} A_{kl} \log(a_{kl}) \tag{28}$$

Derivation of Baum-Welch IV: EM

Recall that A_{kl} and $E_k(b)$ are computable with forward/backward algorithms using current θ_t, whereas the $a_k l$ and $b_k(l)$ parameters form the new candidate θ .
The $Q(\theta|\theta_t)$ is maximized by $a^0_{kl} = \frac{A_{ij}}{\sum_k A_{ik}}$, because the difference for example for the A term is

$$\sum_{k=0}^{M}\sum_{l=1}^{M} A_{kl}\log(\frac{a^0_{kl}}{a_{kl}}) = \sum_{k=0}^{M}(\sum_{l'} A_{kl'})\sum_{l=1}^{M} a^0_{kl}\log(\frac{a^0_{kl}}{a_{kl}}) \tag{29}$$

which is a KL distance, so not negative.
(see EM slides earlier in the Bayesian package).

HMM ”structure" learning I.

Question:Where is the structure in an HMM class? Parameter learning is not enough?

1. Learning of ”HMM length" (??, infinite)
2. Learning of order (??, HMM is based on a first-order Markov chain (note the conversion of nth-order MC to first-order using increased state space)
3. Learning degree of inhomogeneity (??, we discuss mainly homogeneous transitions)
4. Learning the cardinality of the state space: yes, additionally the constraints of the transition/emission probabilities using SFSA (or BN) representations.

Recall the speciality of the silent states (elimination&induced parametric constraints for transitions between real states).
Note the difference between the stochastic FSA representation of an HMM and the represented HMM. For example in case of profile HMMs, the SFSA representation of a profile HMM has a length L and the represented profile HMM has potentially infinite length and state space with size 3+3L.
Dynamic/Temporal Bayesian networks: further generalization of sparse representation for intra-state, inter-state and emission probabilistic dependencies => dynamic-BNs see (MI-ch-15). Assume n binary state descriptor resulting in 2^n states and 2^{2n} transitions in $p(s_{t+1}|s_t)$. A BN can represent efficiently this dependency (similarly to an SFSA, (is there a transparent transformation between these representations?)

HMM ”structure" learning II.

Goal-SFSA: Learning a SFSA from a given model class $\mathcal{M}$. Note that the unrestricted class of FSAs with n states contains 2^{n^2} models, though the profile HMM class $pHMM$ contains SFSA models with different length (i.e. one for each valid state space).

Goal-DBN: Learning a D-BN including a subnetwork M’ representing $p(s_{t+1}|s_t, M')$ from a given model class $\mathcal{M}$.

Maximum likelihood approach:

$M^{ML} = \arg\max_M p(x^{(1)}, \ldots, x^{(n)}|\theta^{ML}, M)$, where

$\theta^{ML} = \arg\max_\theta p(x^{(1)}, \ldots, x^{(n)}|\theta, M)$

Bayesian approach:

$M^{MAP} = \arg\max_M p(M|x^{(1)}, \ldots, x^{(n)})$ (parameters averaged out(?)). Problem: lack of closed form for parameters averaging and numerically hard (practicall ”Precall" (as wee will see): in a Dynamic BN approach, the usage of multinomial local conditionals (in intra-/inter-transition/emission dependencies) with Dirichlet parameter priors guarantees an efficiently computable closed form (for a ”hypothetical" full observation) approximation:

$M^{MAP} = \arg\max_M p(M, |\theta^{MAP}, |x^{(1)}, \ldots, x^{(n)})$, where

$\theta^{MAP} = \arg\max_\theta p(\theta|x^{(1)}, \ldots, x^{(n)}|, M)$

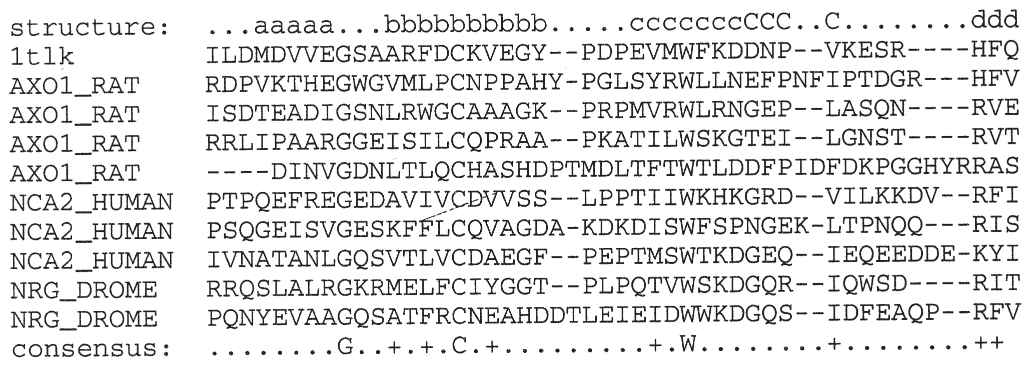
Practical questions for profile HMM: ”structure prior"?: , ”parameter prior": Dirichlet...

Learning any conserved subsequence called **motifs** M (with length L):

$\arg\max_M \prod_i \max_j p(x^{(i)}_{j:j+L}|M) + ComplexityPenalty(M)$ (the best subsequence model occurring in each sequence i with high probability)

Multiple alignment

Recall that the primary goal of pairwise alignment was the scoring of homology.
Goal of multiple alignment: understanding detailed (position-level) homology relations of sequences
Ideal solution: reconstruction of evolutionary tree of sequences (at position level) Problem: complex model (statistical and computational complexity).
Approximate solution: identification of homologous positions (i.e. grouping and alignment in columns), i.e. scoring function and algorithm



```
structure:   ...aaaaa...bbbbbbbbbb.....cccccccCCC..C........ddd
1tlk         ILDMDVVEGSAARFDCKVEGY--PDPEVMWFKDDNP--VKESR----HFQ
AXO1_RAT     RDPVKTHEGWGVMLPCNPPAHY-PGLSYRWLLNEFPNFIPTDGR---HFV
AXO1_RAT     ISDTEADIGSNLRWGCAAAGK--PRPMVRWLRNGEP--LASQN----RVE
AXO1_RAT     RRLIPAARGGEISILCQPRAA--PKATILWSKGTEI--LGNST----RVT
AXO1_RAT     ----DINVGDNLTLQCHASHDPTMDLTFTWTLDDFPIDFDKPGGHYRRAS
NCA2_HUMAN   PTPQEFREGEDAVIVCDVVSS--LPPTIIWKHKGRD--VILKKDV--RFI
NCA2_HUMAN   PSQGEISVGESKFFLCQVAGDA-KDKDISWFSPNGEK-LTPNQQ---RIS
NCA2_HUMAN   IVNATANLGQSVTLVCDAEGF--PEPTMSWTKDGEQ--IEQEEDDE-KYI
NRG_DROME    RRQSLALRGKRMELFCIYGGT--PLPQTVWSKDGQR--IQWSD----RIT
NRG_DROME    PQNYEVAAGQSATFRCNEAHDDTLEIEIDWWKDGQS--IDFEAQP--RFV
consensus:   ........G..+.+.C.+.........+.W........+........++
```

Methods

1. Sum of pairs approach
2. Progressive multiple alignment
3. Profile HMM based multiple alignment

MA:Sum of pairs approach

Idea: use a general, not position specific scoring matrix in a pairwise manner.
The SP score of a multiple alignment m containing columns m_i using $s(.,)$ substitution matrix, which includes a linear gap is

$$S(m) = \sum_i S(m_i) = \sum_i \sum_{1 \leq k < l \leq N} s(m_i, m_i). \quad (30)$$

Paradoxon: with increasing N the relative effect of a single (e.g. non-conserved) residue is decreasing (whereas biological certainty is increasing)
Algorithm:The generalization of the global pairwise alignment algorithm: multidimensional dynamic programming for constructing m with optimal SP score for N sequences. Problem: $2^N - 1$ alternatives at L^N position combinations ($\mathcal{O}(2^N L^N)$).

MA:Progressive multiple alignment

Idea: build a binary tree representing approximately the pairwise distances and align sequences and groups of sequences upwards according to this tree.
First we need a new operation: the pairwise alignment of two multiple alignments (or ”profiles") containing sequences 1 to n and $n+1$ to N. Note that the linear gap penalty based SP score for such profiles can be directly applied and the earlier pairwise alignment algorithm as well (simply working with blocks).

- Construct distance matrix over $N(N-1)/2$ pairs using pairwise alignment and converting observed differences to substitution numbers.
- Construct a guide tree using the *neighbour-joining algorithm* (see phylogenetic tree learning slides).
- Progressively align at nodes in order of decreasing similarity, using sequence-sequence, sequence-profile and profile-profile alignment.

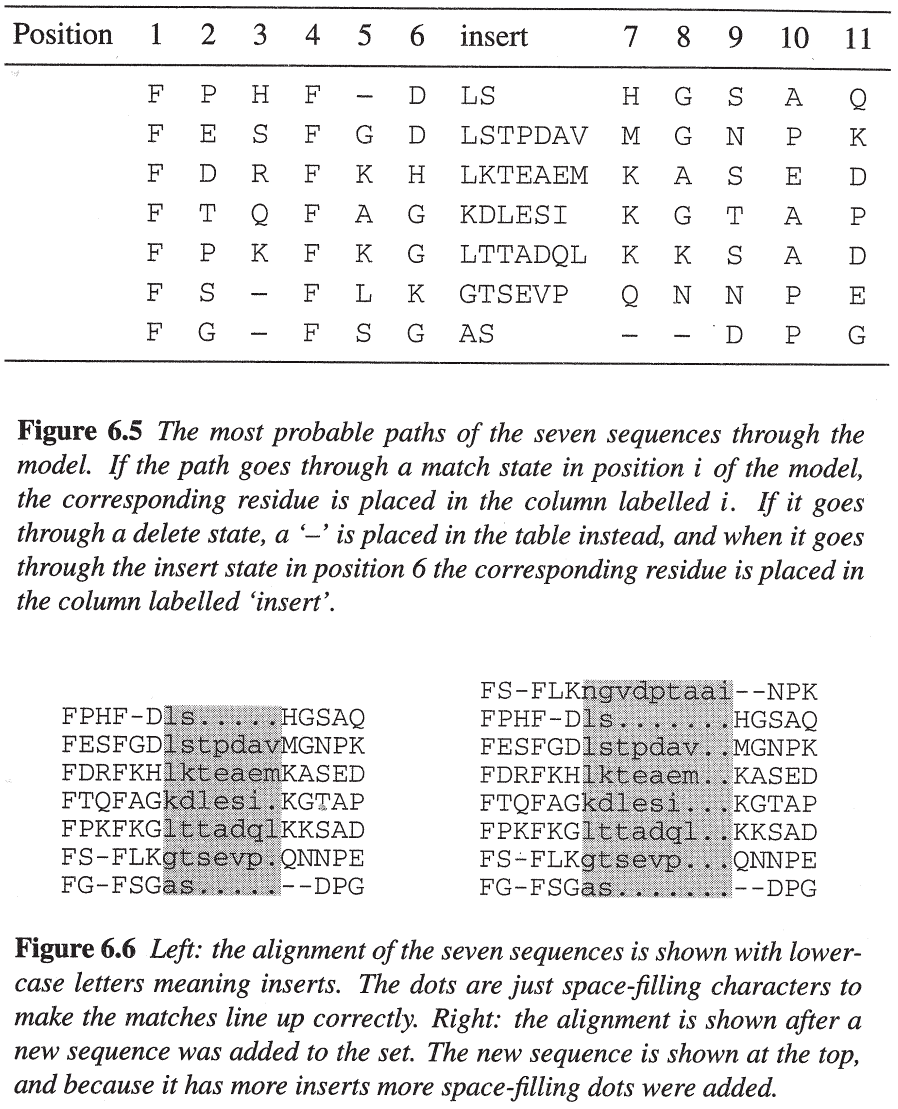
Note the circularity: to reconstruct the pairwise distances for tree learning, we have to apply the scoring matrix corresponding to the target distance.

MA:Profile HMM based

Idea: Learn a pHMM, align (”decode") each sequence with Viterbi and construct multiple alignment (with some conventions).
Observe: in case of a given pHMM the multiple alignment can be reconstructed from $(x^{(1)}, \pi^{(1)}), \ldots, (x^{(N)}, \pi^{(N)})$ pairs, except alignments within each insert blocks (e.g. $(ABAC, m_S m_1 i_1 i_1 m_2 m_T), (ABBAC, m_S m_1 i_1 i_1 i_1 m_2 m_T)$ in pHMM with length 2).

```
Require: pHMM length, $X = x^{(1)}, \ldots, x^{(n)}$
Ensure: multiple alignment of $X$
  Construct pHMM with specified length and ini parameters.
  Estimate parameters from $x^{(1)}, \ldots, x^{(n)}$.
  for $i = 1$ to $n$ do
    Compute $\pi^{(i)}$ by aligning $x^{(i)}$.
  Construct multiple alignment from $(x^{(1)}, \pi^{(1)}), \ldots, (x^{(N)}, \pi^{(N)})$.
```

MA:Profile HMM based II



Position	1	2	3	4	5	6	insert	7	8	9	10	11
	F	P	H	F	–	D	LS	H	G	S	A	Q
	F	E	S	F	G	D	LSTPDAV	M	G	N	P	K
	F	D	R	F	K	H	LKTEAEM	K	A	S	E	D
	F	T	Q	F	A	G	KDLESI	K	G	T	A	P
	F	P	K	F	K	G	LTTADQL	K	K	S	A	D
	F	S	–	F	L	K	GTSEVP	Q	N	N	P	E
	F	G	–	F	S	G	AS	–	–	D	P	G

Figure 6.5 *The most probable paths of the seven sequences through the model. If the path goes through a match state in position i of the model, the corresponding residue is placed in the column labelled i. If it goes through a delete state, a '–' is placed in the table instead, and when it goes through the insert state in position 6 the corresponding residue is placed in the column labelled 'insert'.*

```
FPHF-Dls.....HGSAQ
FESFGDlstpdavMGNPK
FDRFKHlkteaemKASED
FTQFAGkdlesi.KGTAP
FPKFKGlttadqlKKSAD
FS-FLKgtsevp.QNNPE
FG-FSGas.....--DPG
```

```
FS-FLKngvdptaai--NPK
FPHF-Dls.......HGSAQ
FESFGDlstpdav..MGNPK
FDRFKHlkteaem..KASED
FTQFAGkdlesi...KGTAP
FPKFKGlttadql..KKSAD
FS-FLKgtsevp...QNNPE
FG-FSGas.......--DPG
```

Figure 6.6 *Left: the alignment of the seven sequences is shown with lower-case letters meaning inserts. The dots are just space-filling characters to make the matches line up correctly. Right: the alignment is shown after a new sequence was added to the set. The new sequence is shown at the top, and because it has more inserts more space-filling dots were added.*

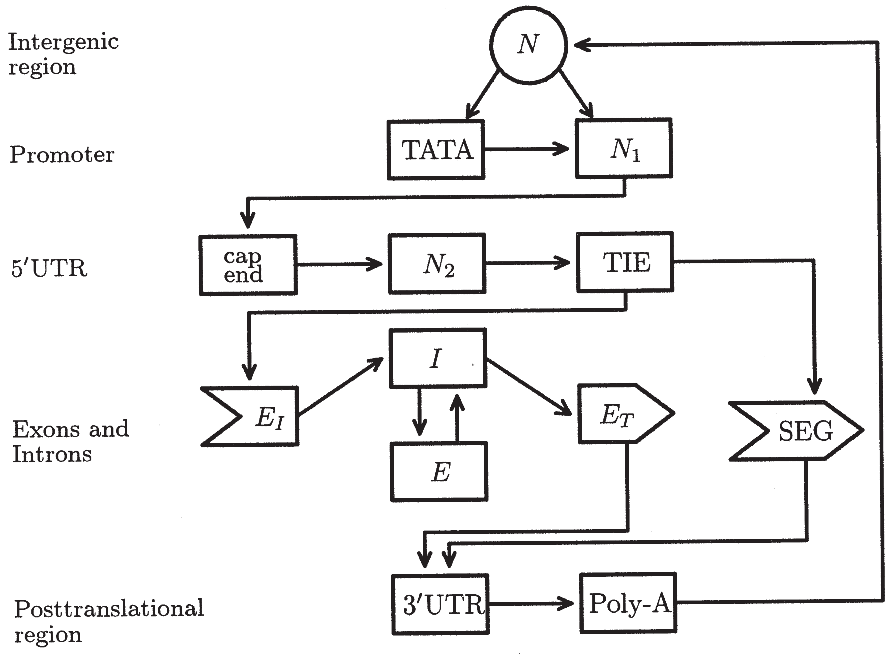
Gene finding:GENSCAN

Semihidden HMM a state can emit words with arbitrary length distribution L_S and symbols $Y_{S,l}$ (not just a symbol or words with length following a geometric distribution). A parse ϕ is a sequence of states and corresponding lengths (partition of observation is not trivial in such case!). HMM algorithms are more complex.
Application: parse of a DNA-segment with Viterbi.
Burge(1997)/EG:GENSCAN, human
Recall: what is a gene? (here we follow a protein-coding interpretation)
The training data: 380 genes, 142 single-exon genes, 1492 exons, coding region of 1619 genes

Gene finding:GENSCAN

Structural elements(of a protein coding gene): see slides.

1. Upstream region: promoter region: **TATA box**: present in 70% of genes at 28-34 bases upstream from the start of transcription.
2. **5’ untranslated region** (5’UTR): follows the promoter starting with the **cap end** region (8 bases) and ending with **translation initiation end** (TIE) (18 bases).
3. $Exon - [intron - exon]^*$: recall intron types and structure
4. **3’ untranslated region** (3’UTR) contains one or more **Poly-A** signal (6 bases).



Gene finding:GENSCAN II.

The **transition** probabilities are estimated from the data (to TATA, to SEG and to multi-exon).
Intergenic region: Distance between genes is modeled by a geometric distribution with mean $p/1-p = |genome|/|genes|$ and the sequence is generated with a fifth-order MC with parameters $3 \cdot 4^5$ called **intergenic null model** (INM).
TATA box is modeled with a 15-base weight matrix (independent multinomials). N_1 is from the INM with length distributed uniformly from 28 to 34. **Cap end** is modeled with an 8-base weight matrix. N_1 is from the INM with length from a geometric distribution with mean 735 bases. **TIE** is modeled with a 18-base weight matrix.
Single exon gene (SEG) is modeled with a nonhomogeneous (3-phase) fifth-order MC generating first the start codon atg and ending with the three stop codons taa, tag, tga. Length follows the empirical distribution.
Multiexon gene is modeled with the SEG model for the exons. The length of the introns are modeled with empirical distribution independently for initial, internal and terminal introns. The intron sequence generation starts with splitting a random codon with 1/3 probability to 0/3, 1/2 or 2/1. This prefix starts the intron, then the donor splice signal is modeled with a decomposed weight matrix with length 6, then the INM generates the intron, finally the acceptor splice signal is again modeled with a decomposed weight matrix with length 20, which is closed with postfix part of the splitted codon.
The **3'UTR** is modeled with the INM with geometric length of mean 450. The **Poly-A** is modeled with a 6-base weight matrix.