Sequence Analysis and Genomics

5 · Hidden Markov models II: HMMs for alignment

Maria Waldl Elisa Israel

Bioinformatics, Leipzig University
maria.waldl@bioinf.uni-leipzig.de

09/11/2026 · WiSe 2026/27 · module 10-202-2207

Today

After today you can ...

- map the Gotoh recursion onto a three-state pair-HMM, and say what the probabilistic view adds
- explain why the single best alignment can mislead, using posterior match probabilities
- build a profile HMM from an MSA, and say how it differs from a PSSM, especially for gaps
- explain why a profile finds more distant homologs than a single-sequence search
- read HMMER output: bit scores, E-values, posterior probabilities
- choose between BLAST, phmmer and hmmsearch for a given question

Last time

Your turn

1. Markov chains and log-odds: how do you decide whether a window is GC-rich?
2. Viterbi: which question does it answer, and how?
3. Forward: what changes compared with Viterbi?
4. Posterior decoding: what does $P(\pi_i = k \mid x)$ tell you that the Viterbi path does not?
5. Training: where do T and E come from, with and without known paths?

Four questions, four algorithms

question	**algorithm**	**recursion**
Which path explains x best?	Viterbi	$V(l,i) = E(l,x_i)\ \max_k V(k,i-1)\ T(k,l)$
How probable is x?	forward	$\alpha(l,i) = E(l,x_i)\ \sum_k \alpha(k,i-1)\ T(k,l)$
How sure are we at i?	forward, backward	$P(\pi_i = k \mid x) = \alpha(k,i)\,\beta(k,i)/P(x)$
Where do the numbers	counting;	$T(k,l) = A_{kl}/\sum_{l'} A_{kl'}$
come from?	Baum-Welch	$E(k,b) = B_k(b)/\sum_{b'} B_k(b')$

Today every row comes back for alignments: Viterbi = Gotoh, forward = all alignments, posteriors = how sure each aligned pair is, counting = building a profile.

Log scores in practice

1. log of every T and E *once*, before the DP; then the DP only adds
2. $\log 0 = -\infty$: forbidden stays forbidden (`-inf` in code)
3. base 2: bits; base e: `math.log`; same path

$\log_2 T$	H	L
0	-1.00	-1.00
H	-0.51	-1.74
L	-2.32	-0.32

$\log_2 E$	A	C	G	T
H	-3.32	-1.32	-1.32	-3.32
L	-1.74	-2.32	-2.32	-1.74

Column 2 of the Viterbi table for TGGGGT, in both forms:

$$V(H,2) = 0.4 \cdot \max\{0.05 \cdot 0.7,\ 0.15 \cdot 0.2\} = 0.014$$

$$\log_2 V(H,2) = -1.32 + \max\{-4.32 - 0.51,\ -2.74 - 2.32\} = -6.16$$

$$V(L,2) = 0.2 \cdot \max\{0.05 \cdot 0.3,\ 0.15 \cdot 0.8\} = 0.024$$

$$\log_2 V(L,2) = -2.32 + \max\{-4.32 - 1.74,\ -2.74 - 0.32\} = -5.38$$

Log-odds: subtract the background as well, $\log_2 \frac{E(H,\mathrm{G})}{q_\mathrm{G}} = +0.68$, $\log_2 \frac{E(H,\mathrm{T})}{q_\mathrm{T}} = -1.32$ ($q = 1/4$): scores like $s(a,b)$ in lecture 2.

Log space for sums

Forward cell $\alpha(H, 2)$ in $\log_2$: the two ways in are $a = \alpha(H, 1)\, T(H, H)$ and $b = \alpha(L, 1)\, T(L, H)$,

$$\log_2 a = -4.837, \qquad \log_2 b = -5.059, \qquad \log_2(a + b) = ?$$

Careful: $\log(a + b) \neq \log a + \log b$

$$\log_2(a + b) = \log_2 a + \log_2\left(1 + 2^{\log_2 b - \log_2 a}\right) \quad (a \geq b)$$

$= -4.837 + \log_2(1 + 2^{-0.222}) = -4.837 + 0.893 = -3.943$; then $+\log_2 E(H, \mathtt{G}) = -1.322$ gives -5.265, i.e. $\alpha(H, 2) = 0.026$ as in lecture 4.

Your turn

Why does Viterbi in log space not need this?

Log space for sums

Forward cell $\alpha(H, 2)$ in $\log_2$: the two ways in are $a = \alpha(H, 1)\, T(H, H)$ and $b = \alpha(L, 1)\, T(L, H)$,

$$\log_2 a = -4.837, \qquad \log_2 b = -5.059, \qquad \log_2(a + b) = ?$$

Careful: $\log(a + b) \neq \log a + \log b$

$$\log_2(a + b) = \log_2 a + \log_2\left(1 + 2^{\log_2 b - \log_2 a}\right) \quad (a \geq b)$$

$= -4.837 + \log_2(1 + 2^{-0.222}) = -4.837 + 0.893 = -3.943$; then $+\log_2 E(H, \mathtt{G}) = -1.322$ gives -5.265, i.e. $\alpha(H, 2) = 0.026$ as in lecture 4.

Your turn

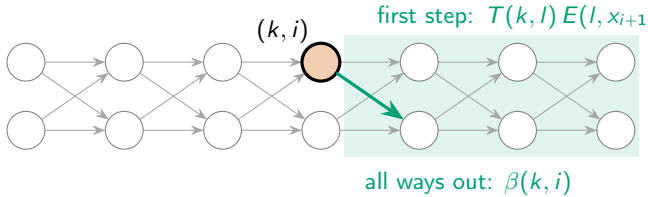
Why does Viterbi in log space not need this?

$\log \max(a, b) = \max(\log a, \log b)$: the log is monotone. A sum has no such shortcut.

Backward: what $\beta(k, i)$ means

Definition: Backward

$\beta(k, i) = P(x_{i+1} \dots x_L \mid \pi_i = k)$: the probability of the *rest* of x, given the state at i



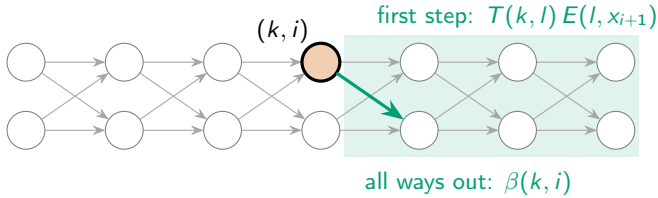
Your turn

Every way out starts with one step. Write $\beta(k, i)$ with $\beta(\cdot, i + 1)$.

Backward: what $\beta(k, i)$ means

Definition: Backward

$\beta(k, i) = P(x_{i+1} \ldots x_L \mid \pi_i = k)$: the probability of the *rest* of x, given the state at i



Your turn

Every way out starts with one step. Write $\beta(k, i)$ with $\beta(\cdot, i + 1)$.

$$\beta(k, i) = \sum_l T(k, l)\, E(l, x_{i+1})\, \beta(l, i + 1)$$

x_i itself is not in β: it is already in $\alpha(k, i)$.

Backward: why the last column is 1

Your turn

What is $\beta(k, L)$? After x_L, nothing is left to emit.

Backward: why the last column is 1

Your turn

What is $\beta(k, L)$? After x_L, nothing is left to emit.

- $\beta(k, L) = P(\text{empty rest} \mid \pi_L = k) = 1$: the empty rest is certain
- not 0 ("nothing happens") and not $E(k, x_L)$ (x_L belongs to α)
- mirror image of forward: the begin state has probability 1 in column 0

Backward: why the last column is 1

Your turn

What is $\beta(k, L)$? After x_L, nothing is left to emit.

- $\beta(k, L) = P(\text{empty rest} \mid \pi_L = k) = 1$: the empty rest is certain
- not 0 ("nothing happens") and not $E(k, x_L)$ (x_L belongs to α)
- mirror image of forward: the begin state has probability 1 in column 0

Definition: With an end state

The path still has to take the step to the end: $\beta(k, L) = T(k, \text{end})$. Without an end state, the model may stop anywhere for free: it says nothing about the length of x. The pair-HMM later today has an end state.

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H							1
L							1

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H						0.16	1
L							1

$$\beta(H,5) = T(H,H)\,E(H,\mathtt{T})\,\beta(H,6) + T(H,L)\,E(L,\mathtt{T})\,\beta(L,6) = 0.7 \cdot 0.1 \cdot 1 + 0.3 \cdot 0.3 \cdot 1 = 0.16$$

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H						0.16	1
L						0.26	1

$$\beta(L,5) = T(L,H)\,E(H,\mathtt{T})\,\beta(H,6) + T(L,L)\,E(L,\mathtt{T})\,\beta(L,6) = 0.2 \cdot 0.1 \cdot 1 + 0.8 \cdot 0.3 \cdot 1 = 0.26$$

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H					?	0.16	1
L					?	0.26	1

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H					0.0604	0.16	1
L					0.0544	0.26	1

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H				0.0202	0.0604	0.16	1
L				0.0135	0.0544	0.26	1

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H			$6.46{\cdot}10^{-3}$	0.0202	0.0604	0.16	1
L			$3.78{\cdot}10^{-3}$	0.0135	0.0544	0.26	1

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0							
H		$2.04{\cdot}10^{-3}$	$6.46{\cdot}10^{-3}$	0.0202	0.0604	0.16	1
L		$1.12{\cdot}10^{-3}$	$3.78{\cdot}10^{-3}$	0.0135	0.0544	0.26	1

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Backward on TGGGGT

	$i=0$	$i=1$ T	$i=2$ G	$i=3$ G	$i=4$ G	$i=5$ G	$i=6$ T
0	$2.7{\cdot}10^{-4}$						
H		$2.04{\cdot}10^{-3}$	$6.46{\cdot}10^{-3}$	0.0202	0.0604	0.16	1
L		$1.12{\cdot}10^{-3}$	$3.78{\cdot}10^{-3}$	0.0135	0.0544	0.26	1

$\beta(0,0) = \sum_l T(0,l)\, E(l,\mathtt{T})\, \beta(l,1) = 2.7{\cdot}10^{-4} = P(x)$ (as forward)

Your turn 6 min

Fill the table from the right. Finish with $\beta(0,0)$: what should it be?

$T(k,l)$	H	L
0	0.5	0.5
H	0.7	0.3
L	0.2	0.8

$E(k,b)$	A	C	G	T
H	0.1	0.4	0.4	0.1
L	0.3	0.2	0.2	0.3

Posteriors: dividing by $P(x)$

$$P(\pi_i = k \mid x) = \frac{P(\pi_i = k,\, x)}{P(x)} = \frac{\alpha(k,i)\,\beta(k,i)}{P(x)}$$

i	1	2	3	4	5	6
x_i	T	G	G	G	G	T
$\alpha(H,i)\,\beta(H,i)$	$1.02{\cdot}10^{-4}$	$1.68{\cdot}10^{-4}$	$1.9{\cdot}10^{-4}$	$1.88{\cdot}10^{-4}$	$1.59{\cdot}10^{-4}$	$7.8{\cdot}10^{-5}$
$\alpha(L,i)\,\beta(L,i)$	$1.68{\cdot}10^{-4}$	$1.02{\cdot}10^{-4}$	$7.96{\cdot}10^{-5}$	$8.2{\cdot}10^{-5}$	$1.11{\cdot}10^{-4}$	$1.92{\cdot}10^{-4}$

Your turn

Before the next overlay: what does each column of $\alpha\beta$ add up to, and why?

Posteriors: dividing by $P(x)$

$$P(\pi_i = k \mid x) = \frac{P(\pi_i = k,\, x)}{P(x)} = \frac{\alpha(k,i)\,\beta(k,i)}{P(x)}$$

i	1	2	3	4	5	6
x_i	T	G	G	G	G	T
$\alpha(H,i)\,\beta(H,i)$	$1.02{\cdot}10^{-4}$	$1.68{\cdot}10^{-4}$	$1.9{\cdot}10^{-4}$	$1.88{\cdot}10^{-4}$	$1.59{\cdot}10^{-4}$	$7.8{\cdot}10^{-5}$
$\alpha(L,i)\,\beta(L,i)$	$1.68{\cdot}10^{-4}$	$1.02{\cdot}10^{-4}$	$7.96{\cdot}10^{-5}$	$8.2{\cdot}10^{-5}$	$1.11{\cdot}10^{-4}$	$1.92{\cdot}10^{-4}$
sum	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$

Your turn

Before the next overlay: what does each column of $\alpha\beta$ add up to, and why?

Every path goes through exactly one state at i: $\sum_k \alpha(k,i)\,\beta(k,i) = P(x) = 2.7{\cdot}10^{-4}$ in *every* column. A free test of your own forward and backward code.

Posteriors: dividing by $P(x)$

$$P(\pi_i = k \mid x) = \frac{P(\pi_i = k, x)}{P(x)} = \frac{\alpha(k,i)\,\beta(k,i)}{P(x)}$$

i	1	2	3	4	5	6
x_i	T	G	G	G	G	T
$\alpha(H,i)\,\beta(H,i)$	$1.02{\cdot}10^{-4}$	$1.68{\cdot}10^{-4}$	$1.9{\cdot}10^{-4}$	$1.88{\cdot}10^{-4}$	$1.59{\cdot}10^{-4}$	$7.8{\cdot}10^{-5}$
$\alpha(L,i)\,\beta(L,i)$	$1.68{\cdot}10^{-4}$	$1.02{\cdot}10^{-4}$	$7.96{\cdot}10^{-5}$	$8.2{\cdot}10^{-5}$	$1.11{\cdot}10^{-4}$	$1.92{\cdot}10^{-4}$
sum	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$	$2.7{\cdot}10^{-4}$
$P(\pi_i = H \mid x)$	0.38	0.62	0.71	0.70	0.59	0.29

Your turn

Before the next overlay: what does each column of $\alpha\beta$ add up to, and why?

Every path goes through exactly one state at i: $\sum_k \alpha(k,i)\,\beta(k,i) = P(x) = 2.7{\cdot}10^{-4}$ in *every* column. A free test of your own forward and backward code.

Baum-Welch, step by step (optional)

One iteration on `TGGGGT` with the small example model of lecture 4 (begin fixed):

1. posteriors from forward and backward (previous frame)
2. expected counts: $B_k(b) = \sum_{i:\, x_i = b} P(\pi_i = k \mid x)$,
   $A_{kl} = \frac{1}{P(x)} \sum_{i=1}^{L-1} \alpha(k,i)\, T(k,l)\, E(l, x_{i+1})\, \beta(l, i+1)$
3. normalise, as with known paths

	expected $B_k(b)$				new $E(k,b)$			
k	A	C	G	T	A	C	G	T
H	0.00	0.00	2.61	0.67				
L	0.00	0.00	1.39	1.33				

	expected A_{kl}		new $T(k,l)$	
k	H	L	H	L
H	2.24	0.74		
L	0.66	1.35		

Baum-Welch, step by step (optional)

One iteration on `TGGGGT` with the small example model of lecture 4 (begin fixed):

1. posteriors from forward and backward (previous frame)
2. expected counts: $B_k(b) = \sum_{i:\, x_i = b} P(\pi_i = k \mid x)$,
   $A_{kl} = \frac{1}{P(x)} \sum_{i=1}^{L-1} \alpha(k,i)\, T(k,l)\, E(l, x_{i+1})\, \beta(l, i+1)$
3. normalise, as with known paths

	expected $B_k(b)$				new $E(k,b)$			
k	A	C	G	T	A	C	G	T
H	0.00	0.00	2.61	0.67	0.00	0.00	0.80	0.20
L	0.00	0.00	1.39	1.33	0.00	0.00	0.51	0.49

	expected A_{kl}		new $T(k,l)$	
k	H	L	H	L
H	2.24	0.74	0.75	0.25
L	0.66	1.35	0.33	0.67

$P(x)$ grows from $2.7{\cdot}10^{-4}$ to 0.0217 (80-fold).

Careful

$E(H, \mathtt{A}) = E(H, \mathtt{C}) = 0$: six letters without an `A` or `C` train a model that forbids them. Real training uses many sequences and pseudocounts.

Check your understanding

- Why take the log of T and E before the DP, and not of the result?
- Why does forward in log space need log-sum-exp, and Viterbi not?
- What does $\beta(k, i)$ contain, and why not x_i?
- Why is $\beta(k, L) = 1$, and what changes with an end state?
- How do you test your forward and backward code with one sum per column?
- In Baum-Welch, why do the expected emission counts add up to L?

An alignment is a path

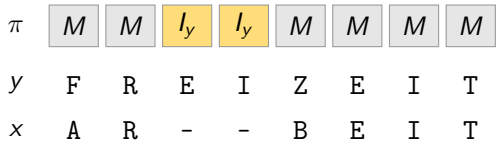
y	F	R	E	I	Z	E	I	T
x	A	R	-	-	B	E	I	T

An alignment is a path

π	M	M	I_y	I_y	M	M	M	M
y	F	R	E	I	Z	E	I	T
x	A	R	-	-	B	E	I	T

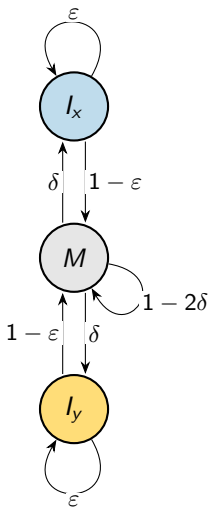
- each column is written by one hidden state: M (*match*: x_i with y_j), I_x (x_i against a gap), I_y (y_j against a gap)

An alignment is a path



- each column is written by one hidden state: M (*match*: x_i with y_j), I_x (x_i against a gap), I_y (y_j against a gap)
- last week the path labelled each letter of x; now it labels each column, and the path *is* the alignment
- in the DP matrix (lecture 1): M a diagonal step, I_x a step down, I_y a step right

Three kinds of column, three states



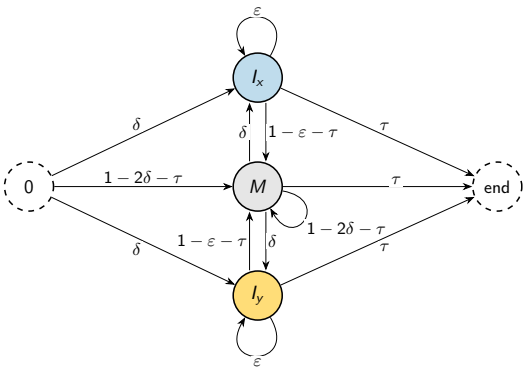
state	column	emission	Gotoh (lecture 2)
M	y_j x_i	$p_{x_i y_j}$	$M(i,j)$
I_x	$-$ x_i	q_{x_i}	$I_x(i,j)$
I_y	y_j $-$	q_{y_j}	$I_y(i,j)$

δ: open a gap (Gotoh: $-d$), ε: extend it ($-e$)

Your turn

Two arrows a general three-state HMM would have are missing. Which, and why?

Begin and end states

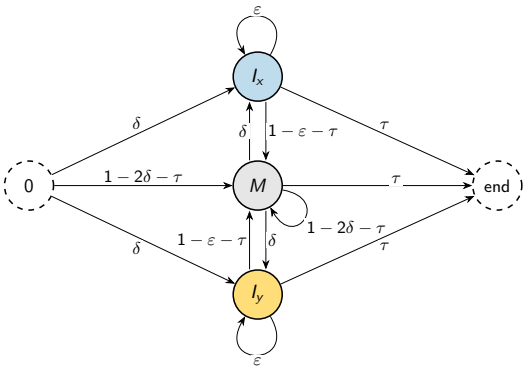


- begin 0: silent, the transitions of M (and τ straight to end: the empty alignment)
- end: silent, reached from every state with probability τ
- every other transition gives up τ, so each row of T still sums to 1

Your turn

Without the end state: add $P(x, y, \pi)$ over all alignments with 1 column, then over all with 2 columns, ... What is the total?

Begin and end states



- begin 0: silent, the transitions of M (and τ straight to end: the empty alignment)
- end: silent, reached from every state with probability τ
- every other transition gives up τ, so each row of T still sums to 1

Your turn

Without the end state: add $P(x, y, \pi)$ over all alignments with 1 column, then over all with 2 columns, ... What is the total?

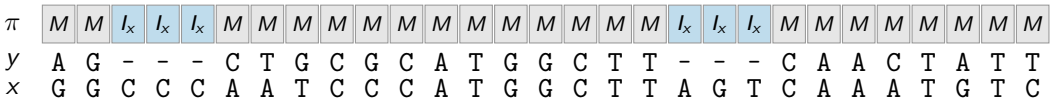
$1 + 1 + \ldots$: not a probability distribution. With the end state the number of columns is geometric, $1/\tau$ on average.

The model writes two sequences



$\delta = 0.05$, $\varepsilon = 0.5$ (gaps of 2 columns on average), $\tau = 0.05$; emissions from lecture 2 ("What does $+1/-1$ assume?"): $p_{aa} = 3/16$, $p_{ab} = 1/48$, $q_a = 1/4$.

The model writes two sequences



$\delta = 0.05$, $\varepsilon = 0.5$ (gaps of 2 columns on average), $\tau = 0.05$; emissions from lecture 2 ("What does $+1/-1$ assume?"): $p_{aa} = 3/16$, $p_{ab} = 1/48$, $q_a = 1/4$.

The model writes two sequences

π	M	M	I_x	I_x	I_x	M	M	M	M	M	M	M	M	M	M	M	M	M	I_x	I_x	I_x	M	M	M	M	M	M	M	M
y	A	G	-	-	-	C	T	G	C	G	C	A	T	G	G	C	T	T	-	-	-	C	A	A	C	T	A	T	T
x	G	G	C	C	C	A	A	T	C	C	C	A	T	G	G	C	T	T	A	G	T	C	A	A	A	T	G	T	C

y = AGCTGCGCATGGCTTCAACTATT

x = GGCCCAATCCCATGGCTTAGTCAAATGTC

$\delta = 0.05$, $\varepsilon = 0.5$ (gaps of 2 columns on average), $\tau = 0.05$; emissions from lecture 2 ("What does $+1/-1$ assume?"): $p_{aa} = 3/16$, $p_{ab} = 1/48$, $q_a = 1/4$.

We see only x and y. The path, and with it the alignment, is hidden.

Your turn

Only x and y given: where would you put the gaps?

One alignment, one probability

With $\pi_0 = 0$ (begin) and L columns:

$$P(x, y, \pi) = \Big[\prod_{c=1}^{L} T(\pi_{c-1}, \pi_c)\, E_c \Big]\; T(\pi_L, \text{end}), \qquad E_c = p_{x_i y_j},\; q_{x_i} \text{ or } q_{y_j}$$

column	A G	G G	- C	- C	- C	C A	⋯	T C	end
state	M	M	I_x	I_x	I_x	M		M	
transition							⋯		
emission							⋯		

One alignment, one probability

With $\pi_0 = 0$ (begin) and L columns:

$$P(x, y, \pi) = \Big[\prod_{c=1}^{L} T(\pi_{c-1}, \pi_c)\, E_c \Big]\; T(\pi_L, \text{end}), \qquad E_c = p_{x_i y_j},\; q_{x_i} \text{ or } q_{y_j}$$

column	A G	G G	- C	- C	- C	C A	⋯	T C	end
state	M	M	I_x	I_x	I_x	M		M	
transition	0.85	0.85	0.05	0.5	0.5	0.45	⋯	0.85	$\tau = 0.05$
emission							⋯		

One alignment, one probability

With $\pi_0 = 0$ (begin) and L columns:

$$P(x, y, \pi) = \Big[\prod_{c=1}^{L} T(\pi_{c-1}, \pi_c)\, E_c \Big]\; T(\pi_L, \text{end}), \qquad E_c = p_{x_i y_j},\ q_{x_i} \text{ or } q_{y_j}$$

column	A G	G G	- C	- C	- C	C A	⋯	T C	end
state	M	M	I_x	I_x	I_x	M		M	
transition	0.85	0.85	0.05	0.5	0.5	0.45	⋯	0.85	$\tau = 0.05$
emission	1/48	3/16	1/4	1/4	1/4	1/48	⋯	1/48	

One alignment, one probability

With $\pi_0 = 0$ (begin) and L columns:

$$P(x, y, \pi) = \Big[\prod_{c=1}^{L} T(\pi_{c-1}, \pi_c)\, E_c \Big]\; T(\pi_L, \text{end}), \qquad E_c = p_{x_i y_j},\ q_{x_i} \text{ or } q_{y_j}$$

column	A G	G G	- C	- C	- C	C A	⋯	T C	end
state	M	M	I_x	I_x	I_x	M		M	
transition	0.85	0.85	0.05	0.5	0.5	0.45	⋯	0.85	$\tau = 0.05$
emission	1/48	3/16	1/4	1/4	1/4	1/48	⋯	1/48	

The true alignment of the generated pair: 29 columns, 59 factors, $P(x, y, \pi) = 5.62{\cdot}10^{-36}$.

The random model

R: **unrelated.** x and y are written independently, letter by letter from q; after each letter a sequence ends with probability η.

$$P(x, y \mid R) = \eta\,(1-\eta)^n \prod_{i=1}^{n} q_{x_i} \cdot \eta\,(1-\eta)^m \prod_{j=1}^{m} q_{y_j}$$

- no columns, no alignment: R never pairs a letter of x with one of y
- $\eta = \tau = 0.05$: both models expect sequences of about the same length

Your turn

The random model of lecture 2 ("Two hypotheses") had no η. Why does it need one now?

Scores from the model

Divide $P(x, y, \pi)$ by $P(x, y \mid R)$ and take the log, column by column. For *every* alignment π:

$$\log \frac{P(x, y, \pi)}{P(x, y \mid R)} = \text{Gotoh score of } \pi + \log \frac{\tau}{\eta^2}$$

$$s(a, b) = \log \frac{p_{ab}}{q_a q_b} + \log \frac{1 - 2\delta - \tau}{(1 - \eta)^2}, \qquad d = -\log \frac{\delta\,(1 - \varepsilon - \tau)}{(1 - \eta)(1 - 2\delta - \tau)}, \qquad e = -\log \frac{\varepsilon}{1 - \eta}$$

Scores from the model

Divide $P(x, y, \pi)$ by $P(x, y \mid R)$ and take the log, column by column. For *every* alignment π:

$$\log \frac{P(x, y, \pi)}{P(x, y \mid R)} = \text{Gotoh score of } \pi + \log \frac{\tau}{\eta^2}$$

$$s(a, b) = \log \frac{p_{ab}}{q_a q_b} + \log \frac{1 - 2\delta - \tau}{(1 - \eta)^2}, \qquad d = -\log \frac{\delta\,(1 - \varepsilon - \tau)}{(1 - \eta)(1 - 2\delta - \tau)}, \qquad e = -\log \frac{\varepsilon}{1 - \eta}$$

	$s(a, a)$	$s(a, b)$	d	e
bits ($\log_2$)	1.50	−1.67	5.17	0.93
$\log_3$	0.95	−1.05	3.26	0.58
lecture 2	+1	−1	1.5	0.3

Scores from the model

Divide $P(x, y, \pi)$ by $P(x, y \mid R)$ and take the log, column by column. For *every* alignment π:

$$\log \frac{P(x, y, \pi)}{P(x, y \mid R)} = \text{Gotoh score of } \pi + \log \frac{\tau}{\eta^2}$$

$$s(a, b) = \log \frac{p_{ab}}{q_a q_b} + \log \frac{1 - 2\delta - \tau}{(1 - \eta)^2}, \qquad d = -\log \frac{\delta\,(1 - \varepsilon - \tau)}{(1 - \eta)(1 - 2\delta - \tau)}, \qquad e = -\log \frac{\varepsilon}{1 - \eta}$$

	$s(a, a)$	$s(a, b)$	d	e
bits ($\log_2$)	1.50	−1.67	5.17	0.93
$\log_3$	0.95	−1.05	3.26	0.58
lecture 2	+1	−1	1.5	0.3

What does the score mean? 5 / 5

The Gotoh score of an alignment is a log-odds score: pair-HMM path against the random model (up to $\log(\tau/\eta^2)$). d and e are no longer chosen: they follow from δ, ε, τ, η.

Viterbi = Gotoh

Gotoh (lecture 2) ▸ recursion	**pair-HMM Viterbi, log-odds**
$M(i,j) = s(x_i, y_j) + \max\{M, I_x, I_y\}(i-1, j-1)$	$V(M,i,j) = s(x_i, y_j) + \max_k V(k, i-1, j-1)$
$I_x(i,j) = \max\{M(i-1,j) - d,\ I_x(i-1,j) - e\}$	$V(I_x,i,j) = \max\{V(M,i-1,j) - d,\ V(I_x,i-1,j) - e\}$
$I_y(i,j) = \max\{M(i,j-1) - d,\ I_y(i,j-1) - e\}$	$V(I_y,i,j) = \max\{V(M,i,j-1) - d,\ V(I_y,i,j-1) - e\}$
$M(0,0) = 0$	begin: $V(M,0,0) = 0$
score: best of the three at (n,m)	$+\log(\tau/\eta^2)$ at the end

Viterbi = Gotoh

Gotoh (lecture 2) ▸ recursion	**pair-HMM Viterbi, log-odds**
$M(i,j) = s(x_i, y_j) + \max\{M, I_x, I_y\}(i-1, j-1)$	$V(M,i,j) = s(x_i, y_j) + \max_k V(k, i-1, j-1)$
$I_x(i,j) = \max\{M(i-1,j) - d,\ I_x(i-1,j) - e\}$	$V(I_x,i,j) = \max\{V(M,i-1,j) - d,\ V(I_x,i-1,j) - e\}$
$I_y(i,j) = \max\{M(i,j-1) - d,\ I_y(i,j-1) - e\}$	$V(I_y,i,j) = \max\{V(M,i,j-1) - d,\ V(I_y,i,j-1) - e\}$
$M(0,0) = 0$	begin: $V(M,0,0) = 0$
score: best of the three at (n,m)	$+\log(\tau/\eta^2)$ at the end

Generated pair: Gotoh with these s, d, e finds the Viterbi alignment, score -1.76 bits; $+4.32 = 2.6$ bits $= \log_2 \frac{P(x,y,\pi^*)}{P(x,y|R)}$.

Viterbi = Gotoh

Gotoh (lecture 2) ▸ recursion	**pair-HMM Viterbi, log-odds**
$M(i,j) = s(x_i, y_j) + \max\{M, I_x, I_y\}(i-1, j-1)$	$V(M,i,j) = s(x_i, y_j) + \max_k V(k, i-1, j-1)$
$I_x(i,j) = \max\{M(i-1,j) - d,\ I_x(i-1,j) - e\}$	$V(I_x,i,j) = \max\{V(M,i-1,j) - d,\ V(I_x,i-1,j) - e\}$
$I_y(i,j) = \max\{M(i,j-1) - d,\ I_y(i,j-1) - e\}$	$V(I_y,i,j) = \max\{V(M,i,j-1) - d,\ V(I_y,i,j-1) - e\}$
$M(0,0) = 0$	begin: $V(M,0,0) = 0$
score: best of the three at (n,m)	$+\log(\tau/\eta^2)$ at the end

Generated pair: Gotoh with these s, d, e finds the Viterbi alignment, score -1.76 bits; $+4.32 = 2.6$ bits $= \log_2 \frac{P(x,y,\pi^*)}{P(x,y|R)}$.

Careful

Gotoh finds the most *probable* alignment only if s, d, e come from a model like this one. And it answers "which alignment?", not "are x and y related?".

All alignments of GA and A

alignment	path

Your turn

Write down all alignments of $x = \texttt{GA}$ and $y = \texttt{A}$ (lecture 1: how many are there?) and the path of each.

All alignments of GA and A

alignment	path
-A GA	$I_x\, M$
A- GA	$M\, I_x$
A-- -GA	$I_y\, I_x\, I_x$
-A- G-A	$I_x\, I_y\, I_x$
--A GA-	$I_x\, I_x\, I_y$

Your turn

Write down all alignments of $x = \texttt{GA}$ and $y = \texttt{A}$ (lecture 1: how many are there?) and the path of each.

All alignments of GA and A

alignment	path	$P(x, y, \pi)$	
-A GA	$I_x\, M$	$\delta\, q_{\mathtt{G}} \cdot (1 - \varepsilon - \tau)\, p_{\mathtt{AA}} \cdot \tau$	$5.27{\cdot}10^{-5}$
A- GA	$M\, I_x$	$(1 - 2\delta - \tau)\, p_{\mathtt{GA}} \cdot \delta\, q_{\mathtt{A}} \cdot \tau$	$1.11{\cdot}10^{-5}$
A-- -GA	$I_y\, I_x\, I_x$	$T(I_y, I_x) = 0$	0
-A- G-A	$I_x\, I_y\, I_x$	$T(I_x, I_y) = 0$	0
--A GA-	$I_x\, I_x\, I_y$	$T(I_x, I_y) = 0$	0

Your turn

Three of the 5 alignments have $P = 0$. Why?

All alignments of GA and A

alignment	path	$P(x, y, \pi)$	
-A GA	$I_x\, M$	$\delta\, q_{\mathrm{G}} \cdot (1-\varepsilon-\tau)\, p_{\mathrm{AA}} \cdot \tau$	$5.27{\cdot}10^{-5}$
A- GA	$M\, I_x$	$(1-2\delta-\tau)\, p_{\mathrm{GA}} \cdot \delta\, q_{\mathrm{A}} \cdot \tau$	$1.11{\cdot}10^{-5}$
A-- -GA	$I_y\, I_x\, I_x$	$T(I_y, I_x) = 0$	0
-A- G-A	$I_x\, I_y\, I_x$	$T(I_x, I_y) = 0$	0
--A GA-	$I_x\, I_x\, I_y$	$T(I_x, I_y) = 0$	0
	sum $= P(x, y)$, the forward value		$6.38{\cdot}10^{-5}$

Your turn

Three of the 5 alignments have $P = 0$. Why?

Forward computes this sum without listing the alignments.

Viterbi picks $I_x\, M$: 0.83 of $P(x, y)$.

Are x and y related at all?

$$\log_2 \frac{P(x,y)}{P(x,y \mid R)} = \log_2 \frac{\sum_\pi P(x,y,\pi)}{P(x,y \mid R)}$$

evidence that x and y are related, summed over all alignments: no single alignment has to be right.

log-odds (bits)	best alignment (Viterbi)	all alignments (forward)
generated pair, related	2.6	6.7
29 and 23 random letters	−13.6	−6.4

Your turn

Why is the forward score never lower than the Viterbi score?

Are x and y related at all?

$$\log_2 \frac{P(x,y)}{P(x,y \mid R)} = \log_2 \frac{\sum_\pi P(x,y,\pi)}{P(x,y \mid R)}$$

evidence that x and y are related, summed over all alignments: no single alignment has to be right.

log-odds (bits)	best alignment (Viterbi)	all alignments (forward)
generated pair, related	2.6	6.7
29 and 23 random letters	−13.6	−6.4

Your turn

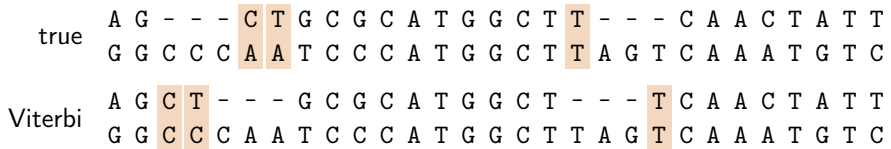
Why is the forward score never lower than the Viterbi score?

This is how HMMER scores a hit (later today).

How much does the best alignment carry?

true	A	G	-	-	-	C	T	G	C	G	C	A	T	G	G	C	T	T	-	-	-	C	A	A	C	T	A	T	T
	G	G	C	C	C	A	A	T	C	C	C	A	T	G	G	C	T	T	A	G	T	C	A	A	A	T	G	T	C

How much does the best alignment carry?



How much does the best alignment carry?

true	A G - - - C T G C G C A T G G C T T - - - C A A C T A T T	
	G G C C C A A T C C C A T G G C T T A G T C A A A T G T C	
Viterbi	A G C T - - - G C G C A T G G C T - - - T C A A C T A T T	
	G G C C C A A T C C C A T G G C T T A G T C A A A T G T C	

- $P(\pi^* \mid x, y) = P(x, y, \pi^*)/P(x, y) = 0.058$; the true alignment: 0.0065
- Viterbi is 9 times as probable as the truth; 6 matched pairs differ (marked)
- 6 alignments tie for the best; together they carry 0.35

How much does the best alignment carry?

true	A G - - - C T G C G C A T G G C T T - - - C A A C T A T T	
	G G C C C A A T C C C A T G G C T T A G T C A A A T G T C	
Viterbi	A G C T - - - G C G C A T G G C T - - - T C A A C T A T T	
	G G C C C A A T C C C A T G G C T T A G T C A A A T G T C	

- $P(\pi^* \mid x, y) = P(x, y, \pi^*)/P(x, y) = 0.058$; the true alignment: 0.0065
- Viterbi is 9 times as probable as the truth; 6 matched pairs differ (marked)
- 6 alignments tie for the best; together they carry 0.35

Your turn

The model wrote the true alignment. Why does it prefer another one?

Backward with an end state

Definition: Backward, pair-HMM

$\beta(k,i,j)$: probability of the rest $x_{i+1}\ldots x_n$, $y_{j+1}\ldots y_m$, given state k at (i,j)

$$\beta(k,i,j) = T(k,M)\, p_{x_{i+1}y_{j+1}}\, \beta(M,i{+}1,j{+}1) + T(k,I_x)\, q_{x_{i+1}}\, \beta(I_x,i{+}1,j) + T(k,I_y)\, q_{y_{j+1}}\, \beta(I_y,i,j{+}1)$$

end: $\beta(k,n,m) = T(k,\text{end}) = \tau$, instead of 1

$\beta(M,i,j)$

	-	A
-		
G		
A		$\tau = 0.05$

$\beta(I_x,i,j)$

	-	A
-		
G		
A		$\tau = 0.05$

$\beta(I_y,i,j)$

	-	A
-		
G		
A		$\tau = 0.05$

Backward with an end state

Definition: Backward, pair-HMM

$\beta(k,i,j)$: probability of the rest $x_{i+1}\ldots x_n$, $y_{j+1}\ldots y_m$, given state k at (i,j)

$$\beta(k,i,j) = T(k,M)\, p_{x_{i+1}y_{j+1}}\, \beta(M,i{+}1,j{+}1) + T(k,I_x)\, q_{x_{i+1}}\, \beta(I_x,i{+}1,j) + T(k,I_y)\, q_{y_{j+1}}\, \beta(I_y,i,j{+}1)$$

end: $\beta(k,n,m) = T(k,\text{end}) = \tau$, instead of 1

$\beta(M,i,j)$

	-	A
-	$6.38{\cdot}10^{-5}$	
G		$6.25{\cdot}10^{-4}$
A		$\tau = 0.05$

$\beta(I_x,i,j)$

	-	A
-		
G	$4.22{\cdot}10^{-3}$	$6.25{\cdot}10^{-3}$
A	0	$\tau = 0.05$

$\beta(I_y,i,j)$

	-	A
-		0
G		0
A		$\tau = 0.05$

Backward with an end state

Definition: Backward, pair-HMM

$\beta(k,i,j)$: probability of the rest $x_{i+1}\dots x_n$, $y_{j+1}\dots y_m$, given state k at (i,j)

$$\beta(k,i,j) = T(k,M)\, p_{x_{i+1}y_{j+1}}\, \beta(M,i{+}1,j{+}1) + T(k,I_x)\, q_{x_{i+1}}\, \beta(I_x,i{+}1,j) + T(k,I_y)\, q_{y_{j+1}}\, \beta(I_y,i,j{+}1)$$

end: $\beta(k,n,m) = T(k,\text{end}) = \tau$, instead of 1

$\beta(M,i,j)$

	-	A
-	$6.38{\cdot}10^{-5}$	
G		$6.25{\cdot}10^{-4}$
A		$\tau = 0.05$

$\beta(I_x,i,j)$

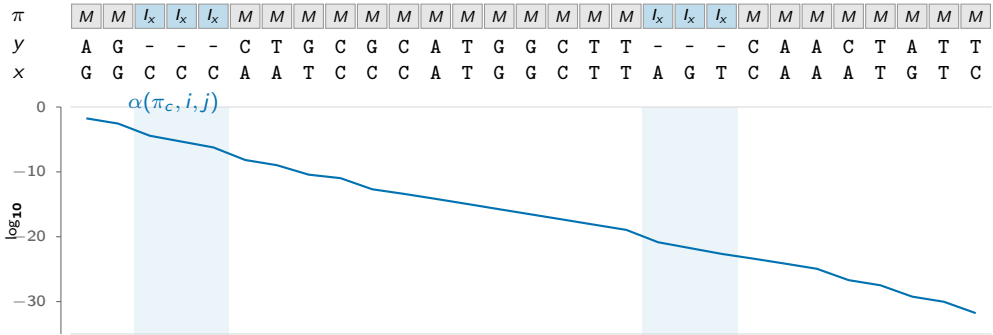
	-	A
-		
G	$4.22{\cdot}10^{-3}$	$6.25{\cdot}10^{-3}$
A	0	$\tau = 0.05$

$\beta(I_y,i,j)$

	-	A
-		0
G		0
A		$\tau = 0.05$

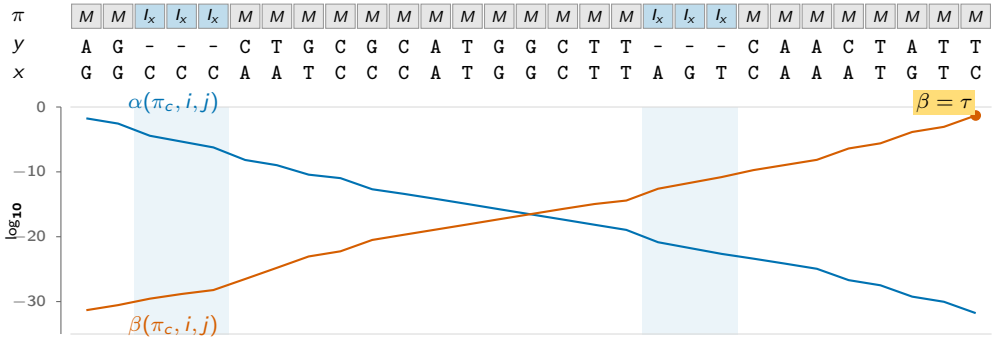
$\beta(M,0,0)$: begin state, $= P(x,y)$

Forward and backward along an alignment



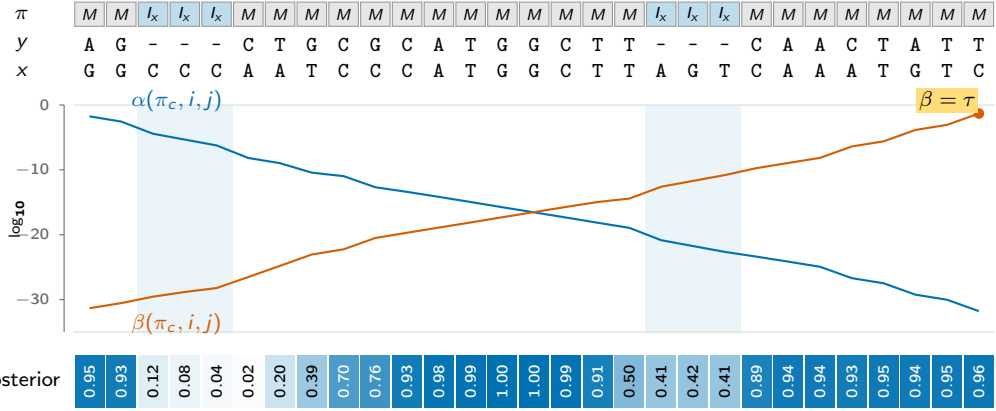
The true alignment, column by column: α of the state at that cell (everything up to here),

Forward and backward along an alignment



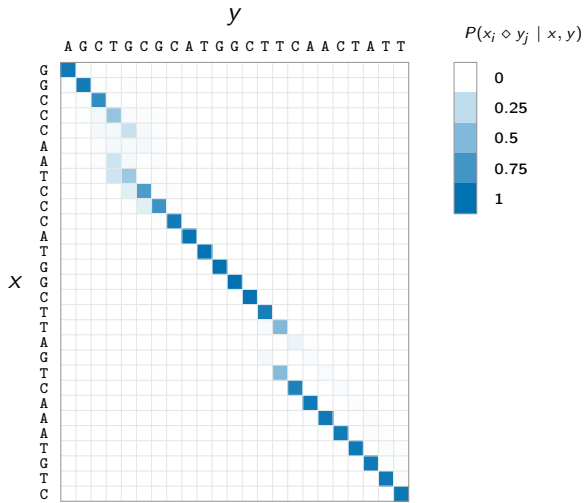
The true alignment, column by column: α of the state at that cell (everything up to here), β (everything after it),

Forward and backward along an alignment



The true alignment, column by column: α of the state at that cell (everything up to here), β (everything after it), and the posterior $\alpha\beta/P(x, y)$: how sure the model is of this column, given only x and y.

Posterior match probabilities



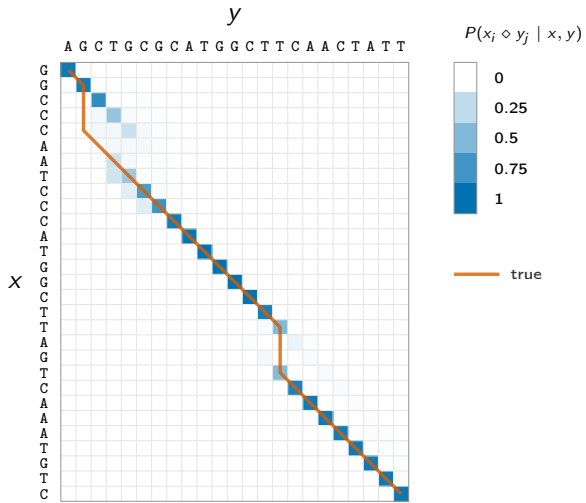
$$P(x_i \diamond y_j \mid x, y) = \frac{\alpha(M, i, j)\,\beta(M, i, j)}{P(x, y)}$$

the probability that x_i and y_j are aligned to each other, over all alignments

Your turn

Where is the model sure, where not? Compare with where the true and the Viterbi alignment differ.

Posterior match probabilities



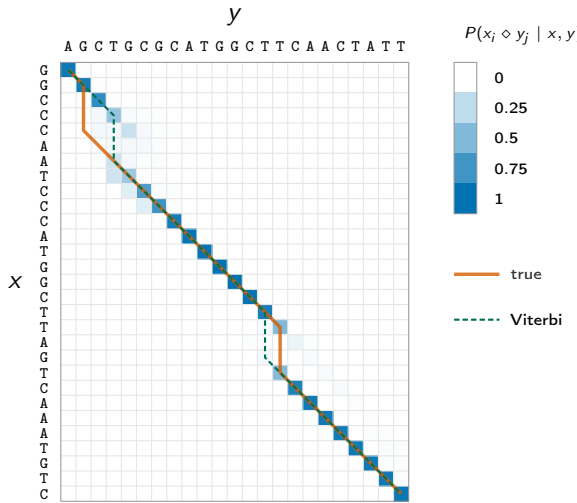
$$P(x_i \diamond y_j \mid x, y) = \frac{\alpha(M, i, j)\,\beta(M, i, j)}{P(x, y)}$$

the probability that x_i and y_j are aligned to each other, over all alignments

Your turn

Where is the model sure, where not? Compare with where the true and the Viterbi alignment differ.

Posterior match probabilities



$$P(x_i \diamond y_j \mid x, y) = \frac{\alpha(M, i, j)\,\beta(M, i, j)}{P(x, y)}$$

the probability that x_i and y_j are aligned to each other, over all alignments

Your turn

Where is the model sure, where not? Compare with where the true and the Viterbi alignment differ.

HMMER prints these per residue: the PP line (later today).

Where pair-HMMs are used

- **MSA:** ProbCons computes posterior match probabilities for every pair of sequences, makes them consistent through a third sequence (as T-Coffee did, lecture 3), then aligns
- **read mapping:** GATK's PairHMM: $P(\text{read} \mid \text{haplotype})$ summed over all alignments, with the base qualities as emission probabilities (lecture 6)
- **database search:** phmmer turns one query into a profile HMM with BLOSUM62 scores and gap probabilities (later today)

A small family

	1	2	3	4	5	6	7	8	9
s1	T	A	G	T	-	C	A	T	T
s2	T	A	G	A	-	C	A	T	T
s3	T	A	G	A	-	C	A	T	T
s4	T	A	G	G	-	G	A	T	T
s5	T	A	G	G	C	G	A	T	T
s6	T	A	G	C	-	-	A	G	T

- 6 DNA sequences, simulated from one ancestor: the true alignment is known
- columns 4–6 are a loop: many substitutions, and the only place where letters were inserted or deleted
- s1, s2, s3 are almost identical; s6 is the most distant

Your turn

A new sequence $x =$ `TAGAGCATT`. Does it belong to the family? Which of its letters go in which column?

Column by column: a PSSM

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
A	0	6	0	2	0	6	0	0
C	0	0	0	1	3	0	0	0
G	0	0	6	2	2	0	1	0
T	6	0	0	1	0	0	5	6
n_j	6	6	6	6	5	6	6	6

- counts $n_j(b)$ of letter b in column j; column 5 (one letter in six) is left out

Column by column: a PSSM

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
A	1/10	7/10	1/10	3/10	1/9	7/10	1/10	1/10
C	1/10	1/10	1/10	1/5	4/9	1/10	1/10	1/10
G	1/10	1/10	7/10	3/10	1/3	1/10	1/5	1/10
T	7/10	1/10	1/10	1/5	1/9	1/10	3/5	7/10
n_j	6	6	6	6	5	6	6	6

- counts $n_j(b)$ of letter b in column j; column 5 (one letter in six) is left out
- one pseudocount per letter (lecture 4): $p_j(b) = \dfrac{n_j(b) + 1}{n_j + 4}$

Column by column: a PSSM

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
A	-1.32	$+1.49$	-1.32	$+0.26$	-1.17	$+1.49$	-1.32	-1.32
C	-1.32	-1.32	-1.32	-0.32	$+0.83$	-1.32	-1.32	-1.32
G	-1.32	-1.32	$+1.49$	$+0.26$	$+0.42$	-1.32	-0.32	-1.32
T	$+1.49$	-1.32	-1.32	-0.32	-1.17	-1.32	$+1.26$	$+1.49$
n_j	6	6	6	6	5	6	6	6

- counts $n_j(b)$ of letter b in column j; column 5 (one letter in six) is left out
- one pseudocount per letter (lecture 4): $p_j(b) = \dfrac{n_j(b) + 1}{n_j + 4}$
- log-odds against the background $q_b = 1/4$: $s_j(b) = \log_2 \dfrac{p_j(b)}{q_b}$ bits, one score table per column, where lecture 2 had one $s(a, b)$ for all positions

Where the PSSM breaks

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	sum
x without the insert	T	A	G	A	C	A	T	T	9.78
	+1.49	+1.49	+1.49	+0.26	+0.83	+1.49	+1.26	+1.49	

Score of a sequence = sum of its column scores, without gaps (like a hit in lecture 2).

- cut the extra letter out of x, and it scores 9.78 bits

Where the PSSM breaks

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	sum
x without the insert	T	A	G	A	C	A	T	T	9.78
	+1.49	+1.49	+1.49	+0.26	+0.83	+1.49	+1.26	+1.49	
x, letters 1–8	T	A	G	A	G	C	A	T	3.98
	+1.49	+1.49	+1.49	+0.26	+0.42	−1.32	−1.32	+1.49	

Score of a sequence = sum of its column scores, without gaps (like a hit in lecture 2).

- cut the extra letter out of x, and it scores 9.78 bits
- the PSSM cannot skip a letter, it can only slide x along: best window 3.98 bits; every letter after the insertion sits in the wrong column (shaded)

Where the PSSM breaks

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	sum
x without the insert	T	A	G	A	C	A	T	T	9.78
	+1.49	+1.49	+1.49	+0.26	+0.83	+1.49	+1.26	+1.49	
x, letters 1–8	T	A	G	A	G	C	A	T	3.98
	+1.49	+1.49	+1.49	+0.26	+0.42	−1.32	−1.32	+1.49	
x, letters 2–9	A	G	A	G	C	A	T	T	1.36
	−1.32	−1.32	−1.32	+0.26	+0.83	+1.49	+1.26	+1.49	

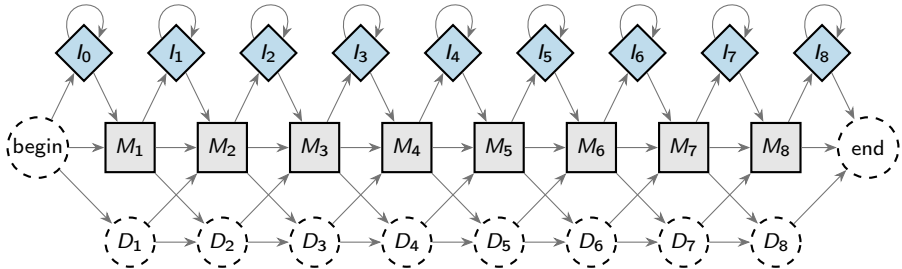
Score of a sequence = sum of its column scores, without gaps (like a hit in lecture 2).

- cut the extra letter out of x, and it scores 9.78 bits
- the PSSM cannot skip a letter, it can only slide x along: best window 3.98 bits; every letter after the insertion sits in the wrong column (shaded)

Your turn

What would the model need to put the extra letter of x aside?

Match, insert, delete



- M_j: column j of the PSSM, emits with $E(M_j, b)$
- I_j: extra letters of x between positions j and $j{+}1$; emits with the background q, can repeat
- D_j: skips position j; emits nothing (silent, like begin and end)
- every arrow has its own probability at every position j

Which columns become match states?

	1	2	3	4	5	6	7	8	9
s1	T	A	G	T	-	C	A	T	T
s2	T	A	G	A	-	C	A	T	T
s3	T	A	G	A	-	C	A	T	T
s4	T	A	G	G	-	G	A	T	T
s5	T	A	G	G	C	G	A	T	T
s6	T	A	G	C	-	-	A	G	T

Which columns become match states?

	1	2	3	4	5	6	7	8	9
s1	T	A	G	T	-	C	A	T	T
s2	T	A	G	A	-	C	A	T	T
s3	T	A	G	A	-	C	A	T	T
s4	T	A	G	G	-	G	A	T	T
s5	T	A	G	G	C	G	A	T	T
s6	T	A	G	C	-	-	A	G	T
residues (of 6)	6	6	6	6	1	5	6	6	6

- a column becomes a match state if at least half of the sequences have a letter there (hmmbuild `--symfrac 0.5`, the default)

Which columns become match states?

	1	2	3	4	5	6	7	8	9
s1	T	A	G	T	-	C	A	T	T
s2	T	A	G	A	-	C	A	T	T
s3	T	A	G	A	-	C	A	T	T
s4	T	A	G	G	-	G	A	T	T
s5	T	A	G	G	C	G	A	T	T
s6	T	A	G	C	-	-	A	G	T
residues (of 6)	6	6	6	6	1	5	6	6	6
state	M_1	M_2	M_3	M_4	I_4	M_5	M_6	M_7	M_8

- a column becomes a match state if at least half of the sequences have a letter there (hmmbuild `--symfrac 0.5`, the default)
- letters in the other columns go to I_j of the match state before them
- a gap in a match column: the row passes through D_j

Your turn

What changes with `--symfrac 0.1`? With `0.9`?

Counting with pseudocounts

node j	0	1	2	3	4	5	6	7	8
$M_j \to M_{j+1}$									
$M_j \to I_j$									
$M_j \to D_{j+1}$									–
$I_j \to M_{j+1}$									
$I_j \to I_j$									
$D_j \to M_{j+1}$	–								
$D_j \to D_{j+1}$	–								–

Your turn

Follow each row of the MSA through the model and count its steps. Start with node 4.

Counting with pseudocounts

node j	0	1	2	3	4	5	6	7	8
$M_j \to M_{j+1}$	6	6	6	6	4	5	6	6	6
$M_j \to I_j$	0	0	0	0	1	0	0	0	0
$M_j \to D_{j+1}$	0	0	0	0	1	0	0	0	–
$I_j \to M_{j+1}$	0	0	0	0	1	0	0	0	0
$I_j \to I_j$	0	0	0	0	0	0	0	0	0
$D_j \to M_{j+1}$	–	0	0	0	0	1	0	0	0
$D_j \to D_{j+1}$	–	0	0	0	0	0	0	0	–

Training with known paths (lecture 4): the MSA gives every row's path; count, add 1 per allowed transition, normalise

Counting with pseudocounts

node j	0	1	2	3	4	5	6	7	8
$M_j \to M_{j+1}$	7/9	7/9	7/9	7/9	5/9	3/4	7/9	7/9	7/8
$M_j \to I_j$	1/9	1/9	1/9	1/9	2/9	1/8	1/9	1/9	1/8
$M_j \to D_{j+1}$	1/9	1/9	1/9	1/9	2/9	1/8	1/9	1/9	–
$I_j \to M_{j+1}$	1/2	1/2	1/2	1/2	2/3	1/2	1/2	1/2	1/2
$I_j \to I_j$	1/2	1/2	1/2	1/2	1/3	1/2	1/2	1/2	1/2
$D_j \to M_{j+1}$	–	1/2	1/2	1/2	1/2	2/3	1/2	1/2	1
$D_j \to D_{j+1}$	–	1/2	1/2	1/2	1/2	1/3	1/2	1/2	–

Training with known paths (lecture 4): the MSA gives every row's path; count, add 1 per allowed transition, normalise

$$T(k,l) = \frac{A_{kl} + 1}{\sum_{l'} (A_{kl'} + 1)}, \qquad E(M_j, b) = p_j(b) \text{ (PSSM frame)}$$

HMMER replaces the $+1$ with a prior learned from many families (Dirichlet mixture).

Sequence weights

- s1, s2, s3 are nearly one sequence: counted three times, they outvote s6 three to one
- **relative weights**: a sequence that is like many others counts less (hmmbuild: Henikoff position-based weights; ClustalW weighted sequences too, lecture 3)
- **effective sequence number**: how many independent sequences the MSA is worth; hmmbuild scales the weighted counts to it before the prior is added

```
NAME   globins4
LENG   149
NSEQ   4
EFFN   0.964844
```

from `globins4.hmm` (HMMER tutorial)

Your turn

Four globins, and HMMER counts them as less than one sequence. Why would it do that?

Position-specific gap costs

node j	0	1	2	3	4	5	6	7	8
open insert	3.17	3.17	3.17	3.17	2.17	3.00	3.17	3.17	3.00
extend insert	1.00	1.00	1.00	1.00	1.58	1.00	1.00	1.00	1.00
open delete	3.17	3.17	3.17	3.17	2.17	3.00	3.17	3.17	–
extend delete	–	1.00	1.00	1.00	1.00	1.58	1.00	1.00	–

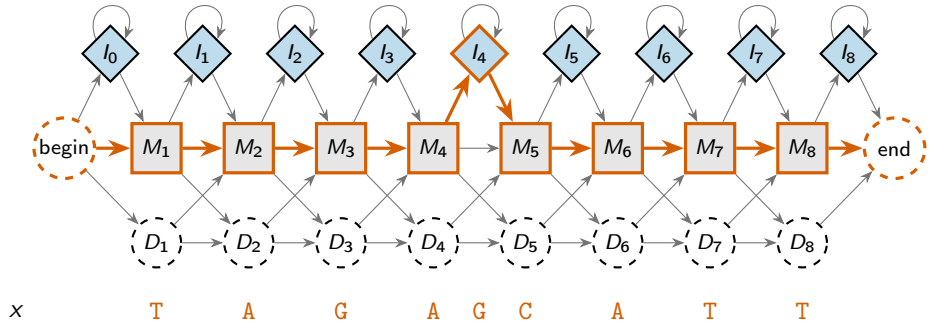
cost in bits $= -\log_2 T$: open an insert $T(M_j, I_j)$, extend it $T(I_j, I_j)$; open a deletion $T(M_j, D_{j+1})$, extend it $T(D_j, D_{j+1})$

- gaps are cheapest where the family had them: node 4, 2.17 instead of 3.17 bits
- Gotoh (lecture 2): one d and one e for every position of both sequences

Your turn

Extending an insert at node 4 costs more than anywhere else. Why?

Aligning a sequence to the profile

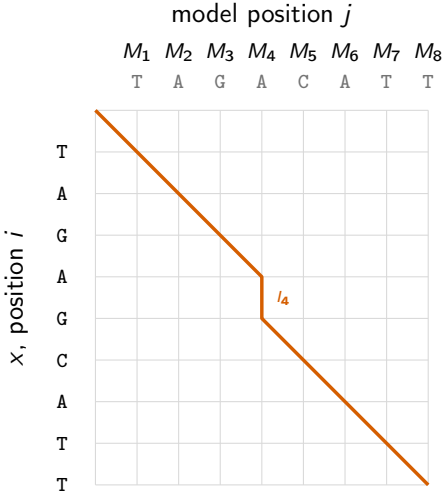


x as a path: $M_1 \ldots M_4$, the extra G in I_4, then $M_5 \ldots M_8$ (how x was simulated). As for every HMM: $P(x, \pi)$ = product of the transitions and emissions along the path. In bits:

emissions	transitions	length of x under the null model	log-odds
$+9.78$	-5.54	$+4.69$	8.94

▸ aligning alignments, lecture 3

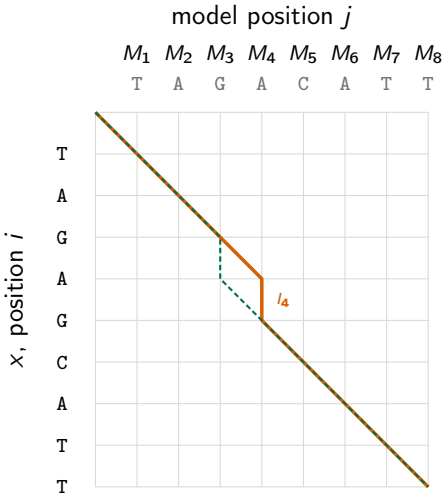
Viterbi and forward over the profile



DP over (model position j, letter i of x), three states per cell, as for the pair-HMM: diagonal M_j, down I_j, right D_j; T and E change with j.

- — Viterbi: the true path, 8.94 bits
- forward, over all 4570 paths: 9.75 bits

Viterbi and forward over the profile



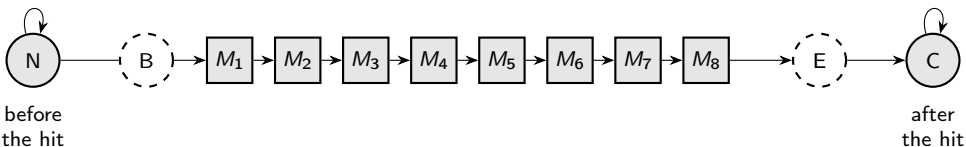
DP over (model position j, letter i of x), three states per cell, as for the pair-HMM: diagonal M_j, down I_j, right D_j; T and E change with j.

- — Viterbi: the true path, 8.94 bits
- forward, over all 4570 paths: 9.75 bits
- ----- the A in I_3, the G in M_4: 7.03 bits

Your turn

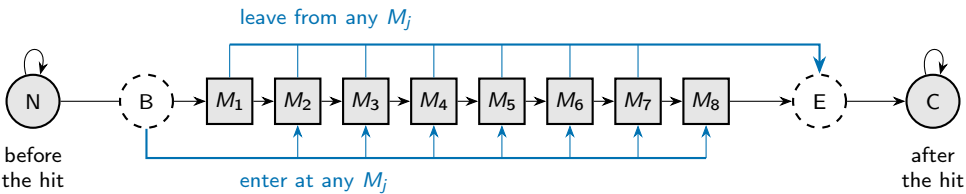
Both paths emit the same letters with the same scores. Why does Viterbi prefer I_4?

Local and glocal



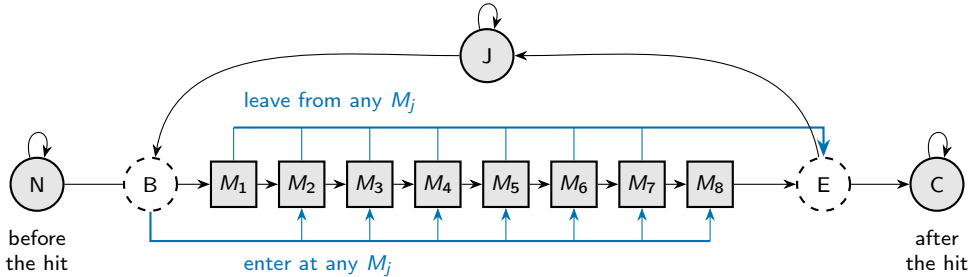
- **global** (so far): all of x against all of the model
- **glocal**: the whole model, anywhere in x; N and C emit the rest of x with the background (free end gaps in x, lecture 1)

Local and glocal



- **global** (so far): all of x against all of the model
- **glocal**: the whole model, anywhere in x; N and C emit the rest of x with the background (free end gaps in x, lecture 1)
- **local**: enter at any M_j, leave from any M_j (Smith–Waterman); a fragment of the family is enough

Local and glocal



- **global** (so far): all of x against all of the model
- **glocal**: the whole model, anywhere in x; N and C emit the rest of x with the background (free end gaps in x, lecture 1)
- **local**: enter at any M_j, leave from any M_j (Smith–Waterman); a fragment of the family is enough
- **multihit**: through J back to B, for a second domain in the same sequence. HMMER's default (hmmsearch, hmmscan): local and multihit

And for proteins?

Your turn

The toy family: 4 letters, 6 sequences, 8 positions. A protein family: 20 letters, often a few dozen sequences, a few hundred positions. What changes?

And for proteins?

Your turn

The toy family: 4 letters, 6 sequences, 8 positions. A protein family: 20 letters, often a few dozen sequences, a few hundred positions. What changes?

- 19 free emission parameters per match state instead of 3
- $+1$ per letter would add 20 pseudocounts to a column of a few sequences: the prior swamps the data
- a better prior knows which amino acids replace each other (Dirichlet mixtures, learned from many families; compare BLOSUM, lecture 2)
- `globins4.hmm`: 4 sequences, 149 positions, EFFN 0.96: most of each column comes from the prior

The HMMER toolbox

program	query	searched against	question
`hmmbuild`	MSA	—	build a profile (earlier today)
`hmmsearch`	profile	sequence database	where else does the family occur?
`hmmpress`, `hmmscan`	sequence	profiles (Pfam)	which families are in my protein?
`phmmer`	one protein	sequence database	like blastp (lecture 2)
`jackhmmer`	one protein	sequence database	iterated: hits make the next profile
`nhmmer`, `nhmmscan`	DNA	DNA	the same for DNA
`hmmalign`	sequences	profile	an MSA of many sequences

Your turn

`hmmsearch` and `hmmscan` compare a profile with a sequence in exactly the same way. Why have both?

Forward, not Viterbi

What does the score mean? 5 / 5

HMMER's score of a sequence x: $S = \log_2 P(x \mid \text{profile}) / P(x \mid \text{null model})$ bits, with $P(x \mid \text{profile})$ from forward: summed over all paths of the local, multihit profile. Null model: as many letters as x has, drawn independently with Swiss-Prot frequencies.

Your turn

BLAST and Viterbi score the best alignment, HMMER the sum over all of them. What does the sum gain for a distant homolog?

Forward, not Viterbi

What does the score mean? 5 / 5

HMMER's score of a sequence x: $S = \log_2 P(x \mid \text{profile}) / P(x \mid \text{null model})$ bits, with $P(x \mid \text{profile})$ from forward: summed over all paths of the local, multihit profile. Null model: as many letters as x has, drawn independently with Swiss-Prot frequencies.

Your turn

BLAST and Viterbi score the best alignment, HMMER the sum over all of them. What does the sum gain for a distant homolog?

- a distant homolog has no single convincing alignment, but many plausible ones, and the sum collects them (posterior match probabilities, earlier today)
- unrelated sequences: $P(S \geq s) \approx e^{-\lambda(s-\tau)}$ with $\lambda \approx \ln 2$ (`globins4.hmm`: 0.70955): one bit more halves P, no fitting per search
- Viterbi still runs, as a fast filter (in two frames)

The null model and composition bias

`globins4.hmm` against 9000 sequences that are not globins, all filters off:

9000 sequences	$E \leq 10$	$E \leq 1$	best score
expected from the null model	10	1	
random letters, Swiss-Prot frequencies	6	0	7.7 bits
shuffled globins: globin letters, random order	77	9	11.4 bits

Your turn

The shuffled globins have no globin left in them except their letters. Why do they score better than random sequences?

The null model and composition bias

`globins4.hmm` against 9000 sequences that are not globins, all filters off:

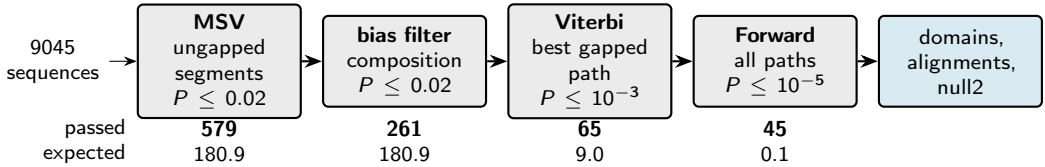
9000 sequences	$E \leq 10$	$E \leq 1$	best score
expected from the null model	10	1	
random letters, Swiss-Prot frequencies	6	0	7.7 bits
shuffled globins: globin letters, random order	77	9	11.4 bits

Your turn

The shuffled globins have no globin left in them except their letters. Why do they score better than random sequences?

- the null model expects Swiss-Prot letters; a sequence with the profile's composition fits the profile better than the null model, without any homology (lecture 2: then E is too optimistic). Worse for low-complexity regions, transmembrane helices, Cys-rich proteins
- *null2*: HMMER builds a second null model from the composition of each hit; the `bias` column says how many bits that took off (3.0 at most for the 45 globins)

Filters: MSV → Viterbi → Forward



`globins4.hmm` against the 45 globins plus 9000 shuffled globins, default settings.

- each stage is slower and more exact than the one before; the full forward runs on 65 of 9045 sequences
- MSV: the best ungapped segments, like BLAST's ungapped extension without the seeds (lecture 2)
- MSV passed three times as many as expected: the shuffled globins again; the bias filter catches part of them

Your turn

Which homologs can the MSV filter lose?

E-values again

Same profile, same globins, two databases:

		HBB_MANSP		HBB2_TRICR	
database	sequences	bits	E	bits	E
`globins45.fa`	45	217.7	$1.9 \cdot 10^{-67}$	118.0	10^{-36}
plus the shuffled globins	9045	217.7	$3.9 \cdot 10^{-65}$	118.0	$2 \cdot 10^{-34}$

Your turn

You want to compare hits from two searches against different databases. Which number do you compare?

E-values again

Same profile, same globins, two databases:

		HBB_MANSP		HBB2_TRICR	
database	sequences	bits	E	bits	E
`globins45.fa`	45	217.7	$1.9 \cdot 10^{-67}$	118.0	10^{-36}
plus the shuffled globins	9045	217.7	$3.9 \cdot 10^{-65}$	118.0	$2 \cdot 10^{-34}$

Your turn

You want to compare hits from two searches against different databases. Which number do you compare?

- $E = Z \cdot P$, with Z the number of comparisons: sequences for `hmmsearch`, profiles for `hmmscan`; here Z grew 201-fold (lecture 2: "Same hit, other database") ▸ lecture 2
- the bit score depends only on the profile and the sequence: compare hits by bits, judge each search by E
- two E-values per sequence: all its domains together (full sequence) and its best domain alone (later today)

A profile is only as good as its MSA

- misaligned columns (lecture 3, "Typical artifacts"): a match state mixes two true positions, its emissions blur
- fragments, gappy regions: columns fall below the residue fraction, become inserts
- one unrelated sequence in the MSA: the profile models it too, and finds its relatives
- a few near-identical sequences: the profile is little more than one sequence (EFFN, earlier today)

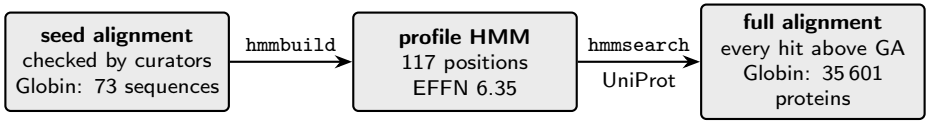
Careful

The E-value assumes that the profile is right. A profile built from a wrong MSA reports its errors with good E-values.

Your turn

`jackhmmer` puts every hit with $E \leq 0.001$ into the next round's profile. What happens after one false positive?

Pfam



- **gathering threshold** (GA): a bit score the curators set per family (Globin: 22 bits); in bits, because bits do not depend on the database (`hmmsearch --cut_ga`)
- **clans**: families with a common ancestor, too far apart for one profile (Globin clan: 10 families)
- Pfam 39.0 (October 2026), tens of thousands of families; Pfam now lives in InterPro

Your turn

Your protein scores 21 bits against the Globin profile; GA is 22 bits. Is it not a globin?

Which search when?

tool	you have	searched against	good for
`blastp`	one protein	sequences	close homologs, quick checks
`phmmer`	one protein	sequences	the same question, scored by forward
`jackhmmer`	one protein	sequences	distant homologs, no family yet
`hmmsearch`	a family (MSA)	sequences	new members of a known family
`hmmscan`	one protein	Pfam profiles	which domains are in it?
`blastn`, `nhmmer`	DNA	DNA	`blastn` faster, `nhmmer` more sensitive

Your turn

Which tool? (a) Which of the 20 000 proteins of a new genome contain a kinase domain? (b) What is this protein? (c) BLAST finds nothing for your protein, and no Pfam family matches.

Which search when?

tool	you have	searched against	good for
`blastp`	one protein	sequences	close homologs, quick checks
`phmmer`	one protein	sequences	the same question, scored by forward
`jackhmmer`	one protein	sequences	distant homologs, no family yet
`hmmsearch`	a family (MSA)	sequences	new members of a known family
`hmmscan`	one protein	Pfam profiles	which domains are in it?
`blastn`, `nhmmer`	DNA	DNA	`blastn` faster, `nhmmer` more sensitive

Your turn

Which tool? (a) Which of the 20 000 proteins of a new genome contain a kinase domain? (b) What is this protein? (c) BLAST finds nothing for your protein, and no Pfam family matches.

(a) `hmmsearch` with Pfam's kinase profile, one search; (b) `blastp` or `phmmer` for named relatives, `hmmscan` for its domains; (c) `jackhmmer`, checking the hits of every round.

Globins: one query vs. a profile

Both against `globins45.fa`: `phmmer` with one β chain (`HBB_HUMAN`), `hmmsearch` with `globins4.hmm` (a β and an α chain, a myoglobin, a lamprey globin)

		bits		$E \leq 0.01$ in all of UniProt	
	sequences	`phmmer`	profile	`phmmer`	profile
β chains	19	142.0–314.3	118.0–217.7	19	19
α chains	19	88.5–105.9	141.3–206.0	19	19
myoglobins	7	24.6–44.2	120.2–214.0	6	7

Your turn

`phmmer` scores the β chains higher than the profile does, and the myoglobins far lower. Why?

Globins: one query vs. a profile

Both against `globins45.fa`: `phmmer` with one β chain (`HBB_HUMAN`), `hmmsearch` with `globins4.hmm` (a β and an α chain, a myoglobin, a lamprey globin)

		bits		$E \leq 0.01$ in all of UniProt	
	sequences	`phmmer`	profile	`phmmer`	profile
β chains	19	142.0–314.3	118.0–217.7	19	19
α chains	19	88.5–105.9	141.3–206.0	19	19
myoglobins	7	24.6–44.2	120.2–214.0	6	7

Your turn

`phmmer` scores the β chains higher than the profile does, and the myoglobins far lower. Why?

- one β chain models its close relatives best; the profile spreads its probability over four globins and knows which positions vary (not a blind test: one of the four is a myoglobin)
- a search of UniProt's size ($Z = 147\,277\,710$): `MYG_MUSAN` gets $E = 0.17$ with `phmmer`, missed; $E = 6.8 \cdot 10^{-31}$ with the profile

▸ blastp

Reading hmmsearch output

`hmmsearch globins4.hmm globins45.fa`, the list of hits (top and bottom):

```
Scores for complete sequences (score includes all domains):
   --- full sequence ---    --- best 1 domain ---    -#dom-
    E-value  score  bias    E-value  score  bias    exp  N  Sequence    Description
    ------- ------ -----    ------- ------ -----    ---- --  --------    -----------
    1.9e-67  217.7   0.1    2.1e-67  217.5   0.1    1.0  1  HBB_MANSP
    1.3e-66  215.0   0.2    1.4e-66  214.9   0.2    1.0  1  HBB_CALAR
    2.6e-66  214.0   3.0    2.9e-66  213.9   3.0    1.0  1  MYG_ESCGI
    ...
    2.1e-37  120.2   0.4    2.3e-37  120.1   0.4    1.0  1  MYG_MUSAN
      1e-36  118.0   0.0    1.1e-36  117.9   0.0    1.0  1  HBB2_TRICR
```

Your turn

Two *E*-values, two scores, two domain counts per line: what does each pair tell you?

Reading hmmsearch output

`hmmsearch globins4.hmm globins45.fa`, the list of hits (top and bottom):

```
Scores for complete sequences (score includes all domains):
   --- full sequence ---    --- best 1 domain ---    -#dom-
    E-value  score  bias    E-value  score  bias    exp  N  Sequence    Description
    ------- ------ -----    ------- ------ -----    ---- --  --------    -----------
    1.9e-67  217.7   0.1    2.1e-67  217.5   0.1    1.0  1  HBB_MANSP
    1.3e-66  215.0   0.2    1.4e-66  214.9   0.2    1.0  1  HBB_CALAR
    2.6e-66  214.0   3.0    2.9e-66  213.9   3.0    1.0  1  MYG_ESCGI
    ...
    2.1e-37  120.2   0.4    2.3e-37  120.1   0.4    1.0  1  MYG_MUSAN
      1e-36  118.0   0.0    1.1e-36  117.9   0.0    1.0  1  HBB2_TRICR
```

Your turn

Two E-values, two scores, two domain counts per line: what does each pair tell you?

- full sequence: all its domains together; best 1 domain: the best one alone
- `bias`: bits taken off by null2; `exp`: expected number of domains, `N`: domains annotated
- listed: $E \leq 10$; counted as hits (included): $E \leq 0.01$, here all 45

Reading hmmsearch output: the domain table

The domain table of MYG_MUSAN, from the same search:

```
>> MYG_MUSAN
   #    score  bias  c-Evalue  i-Evalue hmmfrom  hmm to    alifrom  ali to    envfrom  env to     acc
 ---   ------ ----- --------- --------- ------- -------    ------- -------    ------- -------    ----
   1 !  120.1   0.4   2.3e-37   2.3e-37       7     149 .]       2     142 ..       1     142 [. 0.98
```

Your turn

The alignment of MYG_MUSAN starts at residue 2, its envelope at residue 1. Which coordinates would you use to annotate the domain?

Reading hmmsearch output: the domain table

The domain table of MYG_MUSAN, from the same search:

```
>> MYG_MUSAN
   #    score  bias  c-Evalue  i-Evalue hmmfrom  hmm to    alifrom  ali to    envfrom  env to     acc
 ---   ------ ----- --------- --------- ------- -------    ------- -------    ------- -------    ----
   1 !  120.1   0.4   2.3e-37   2.3e-37       7     149 .]       2     142 ..       1     142 [. 0.98
```

Your turn

The alignment of MYG_MUSAN starts at residue 2, its envelope at residue 1. Which coordinates would you use to annotate the domain?

- `!`: passes the inclusion thresholds; `?`: only reported
- `i-Evalue`: this domain alone, in the whole search (= best 1 domain); `c-Evalue`: a further domain in a sequence already accepted, among the reported sequences only
- `hmm/ali from/to`: where the alignment starts and ends in profile and sequence; `[ ]`: flush with the end, `.`: inside
- `env`: where the domain lies, from the posteriors; use it for annotation. `acc`: mean posterior probability of the aligned residues (next frame)

The PP line

MYG_MUSAN, residues 6–60, aligned to the profile (hmmsearch, 120.2 bits, acc 0.98)

```
  globins4 11 vkavWakveadveevGadiLvrlfvsyPetqefFekFkdLstedelkgsadvkkHg 66
              v++vW+ ve+d    G++iL rlf  yPe+q++F+kFk+ s   elk  ad+k+ +
MYG_MUSAN  6 VNSVWSAVESDLTAIGQNILLRLFEQYPESQNHFPKFKNKS-LGELKDTADIKAQA 60
              **********************************9877.99*********** PP
```

and to one β chain (phmmer, 24.6 bits, acc 0.87):

```
HBB_HUMAN 11 vtalwgkvnvd..evggealgrllvvypwtqrffesfgdlstpdavmgnpkvkahg 64
             v ++w+ v  d   +g   l rl+  yp +q+ f  f + s  + +     +ka++
MYG_MUSAN  6 VNSVWSAVESDltAIGQNILLRLFEQYPESQNHFPKFKNKSLGE-LKDTADIKAQA 60
             778999887663378999*********************99876.678999***** PP
```

Your turn

Where do you trust the phmmer alignment, and where not?

The PP line

MYG_MUSAN, residues 6–60, aligned to the profile (hmmsearch, 120.2 bits, acc 0.98)

```
  globins4 11 vkavWakveadveevGadiLvrlfvsyPetqefFekFkdLstedelkgsadvkkHg 66
              v++vW+ ve+d    G++iL rlf  yPe+q++F+kFk+ s   elk  ad+k+ +
MYG_MUSAN  6 VNSVWSAVESDLTAIGQNILLRLFEQYPESQNHFPKFKNKS-LGELKDTADIKAQA 60
              **********************************9877.99*********** PP
```

and to one β chain (phmmer, 24.6 bits, acc 0.87):

```
HBB_HUMAN 11 vtalwgkvnvd..evggealgrllvvypwtqrffesfgdlstpdavmgnpkvkahg 64
             v ++w+ v  d   +g   l rl+  yp +q+ f  f + s  + +      +ka++
MYG_MUSAN  6 VNSVWSAVESDltAIGQNILLRLFEQYPESQNHFPKFKNKSLGE-LKDTADIKAQA 60
             778999887663378999**********************99876.678999***** PP
```

Your turn

Where do you trust the phmmer alignment, and where not?

- query consensus (capitals: conserved); letter: identical, +: positive score; lower case: insert
- PP: posterior probability that the residue sits right (earlier today), 9: 85–95 %, ∗: $\geq$ 95 %
- uncertainty sits around the gaps and at the ends

Annotating new genomes

A new genome is assembled. Two questions, both answered with profile HMMs:

- **What do its proteins do?** Predict the genes, then `hmmscan` every protein against Pfam (InterProScan runs this and more): domain annotation, "what is this protein?" from "Which search when?"
- **Is the assembly complete?** BUSCO: for a lineage, the genes found *once* in almost every genome of that lineage (from OrthoDB), one profile HMM each. Search the new genome; each gene comes out complete (single or duplicated), fragmented or missing.

Your turn

Why single-copy genes? What could many *duplicated* BUSCOs mean for an assembly?

Annotating new genomes

A new genome is assembled. Two questions, both answered with profile HMMs:

- **What do its proteins do?** Predict the genes, then `hmmscan` every protein against Pfam (InterProScan runs this and more): domain annotation, "what is this protein?" from "Which search when?"
- **Is the assembly complete?** BUSCO: for a lineage, the genes found *once* in almost every genome of that lineage (from OrthoDB), one profile HMM each. Search the new genome; each gene comes out complete (single or duplicated), fragmented or missing.

Your turn

Why single-copy genes? What could many *duplicated* BUSCOs mean for an assembly?

Single copy: a hit is expected exactly once, so 0 or 2 is informative. Many duplicates: both haplotypes assembled separately, contamination, or a real genome duplication. Lecture 7: assembly quality.

Take-home

1. Pair-HMM: Gotoh's three matrices are three states; Viterbi gives the best alignment, its scores are log-odds against a random model
2. Forward sums over all alignments: "are x and y related?"; the best alignment may carry only a small share
3. Posterior match probabilities show which parts of an alignment to trust
4. Profile HMM: an MSA as a model with position-specific emissions and gap costs, built by counting with pseudocounts and weights
5. HMMER scores with forward against a null model; compare hits by bits, judge a search by E; a profile is only as good as its MSA
6. A profile finds more distant homologs than one query; choose the search by the question

Could you answer these?

- Which pair-HMM state belongs to which Gotoh matrix? Which step does neither allow?
- Where do d and e come from in the pair-HMM, and why does the random model need η?
- Why can the best alignment carry only a small part of $P(x, y)$, and what do you use instead?
- Which MSA columns become match states, and where do the other residues go?
- Why does a profile HMM need pseudocounts and sequence weights?
- Two searches against different databases: do you compare bit scores or E-values?
- A sequence scores just above Pfam's gathering threshold. What does that tell you, and what not?
- When do you use `hmmscan` rather than `hmmsearch`, and `jackhmmer` rather than `phmmer`?

Backup: the Gotoh recursion (lecture 2)

$$M(i,j) = \max \begin{cases} M(i-1,j-1) + s(x_i, y_j) \\ I_x(i-1,j-1) + s(x_i, y_j) \\ I_y(i-1,j-1) + s(x_i, y_j) \end{cases}$$

$$I_x(i,j) = \max \begin{cases} M(i-1,j) - d \\ I_x(i-1,j) - e \end{cases} \qquad I_y(i,j) = \max \begin{cases} M(i,j-1) - d \\ I_y(i,j-1) - e \end{cases}$$

$$M(0,0) = 0 \qquad I_x(i,0) = -d - (i-1)\,e \qquad I_y(0,j) = -d - (j-1)\,e$$
$$M(i,0) = M(0,j) = -\infty \qquad I_x(0,j) = -\infty \qquad I_y(i,0) = -\infty$$

No step $I_x \to I_y$ or $I_y \to I_x$; score: the best of the three at (n, m).

◂ Viterbi = Gotoh

Backup: aligning alignments (lecture 3)

Alignments P (rows k) and Q (rows l), columns P_i and Q_j; NW over columns:

$$F(i,j) = \max \begin{cases} F(i-1,j-1) + S(P_i, Q_j) \\ F(i-1,j) + S(P_i, -) \\ F(i,j-1) + S(-, Q_j) \end{cases} \qquad S(P_i, Q_j) = \sum_{k \in P} \sum_{l \in Q} s(P_i^k, Q_j^l)$$

$-$: a column of gaps, inserted into all rows of one alignment

A sequence against a profile HMM is the same DP with one side replaced by the model:

- $S(P_i, Q_j)$ becomes the match emission $\log \frac{E(M_j, x_i)}{q_{x_i}}$, one score table per column
- the gap costs become the transitions of I_j and D_j, different at every position

◂ Aligning a sequence to the profile

Backup: E-values (lecture 2)

Definition: E-value

expected number of hits with score $\geq S$ in a search of random sequences of this size:

$$E = K\,m\,n\,e^{-\lambda S} \qquad \text{in bits } S' = \frac{\lambda S - \ln K}{\ln 2}: \qquad E = m\,n\,2^{-S'}$$

- one bit more halves E; the bit score compares hits across scoring schemes
- $E \propto n$: the same hit in a 200 times larger database, same bits, 200 times larger E
- at least one chance hit: $P = 1 - e^{-E}$; for $E < 0.01$, $E \approx P$

What does the score mean?

R ignores repeats, low complexity and composition bias: in real data, E is too optimistic. $E = 3$ does not mean "not homologous"; $E = 10^{-30}$ does not mean "same function".

◂ E-values again

Backup: notation in the books

	this course	**Durbin et al.**	**HMMER user guide / file**
pair-HMM states	M, I_x, I_y	M, X, Y	–
gap open, extend, end	δ, ε, τ	δ, ε, τ	–
emissions (pair)	$E(M, a, b) = p_{ab}$, q_a	$p_{x_i y_j}$, q_{x_i}	–
profile states	M_j, I_j, D_j	M_j, I_j, D_j	match, insert, delete
profile emissions	$E(M_j, b)$, $E(I_j, b)$	$e_{M_j}(b)$, $e_{I_j}(b)$	match / insert emissions, $-\ln p$
transitions	$T(M_j, M_{j+1})$	$a_{M_j M_{j+1}}$	`m->m`, `m->i`, ..., `d->d`
background	q_b	q_a	null model (null1; null2 for bias)
tables	V, α, β	v, f, b	–

Backup: BLAST on the globins

`blastp` (BLAST+ 2.17.0) with `HBB_HUMAN` against `globins45.fa`, next to `phmmer`: all 38 α and β chains are found by every run; the myoglobins, *E*-values:

	`blastp`	`blastp -word_size 2`	`phmmer`
`MYG_LYCPI`	$1.4 \cdot 10^{-14}$	$1.4 \cdot 10^{-14}$	$4.8 \cdot 10^{-14}$
`MYG_SAISC`	$2.8 \cdot 10^{-12}$	$2.8 \cdot 10^{-12}$	$7 \cdot 10^{-12}$
`MYG_PROGU`	—	$1.5 \cdot 10^{-11}$	$1.3 \cdot 10^{-11}$
`MYG_MOUSE`	—	$1.5 \cdot 10^{-11}$	$2.6 \cdot 10^{-11}$
`MYG_HORSE`	—	$7.9 \cdot 10^{-11}$	$8.2 \cdot 10^{-11}$
`MYG_ESCGI`	$5.3 \cdot 10^{-10}$	$5.3 \cdot 10^{-10}$	$3.5 \cdot 10^{-10}$
`MYG_MUSAN`	—	$1.4 \cdot 10^{-8}$	$5.1 \cdot 10^{-8}$
globins found	41 of 45	45 of 45	45 of 45

Your turn

Where both find a myoglobin, their *E*-values agree. So why does `blastp` miss four of them?