Hidden Markov Models in DNA Sequencing

Michael Marino & Eric Marti

Markov Process

- Stochastic process satisfying the Markov Property
- Probability distribution of future states dependent on present and past states depends only on present state

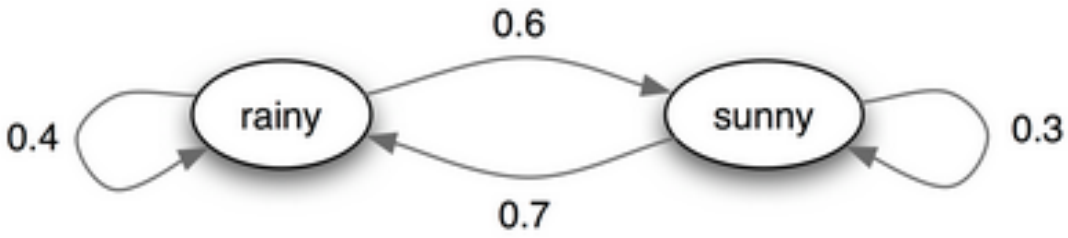


Image source:
http://www.math.cornell.edu/~numb3rs/blanco/Undercurrents.html

Hidden Markov Model

- Assume behavior of unobservable hidden states to be a Markov process
- Requires knowledge regarding emission and transmission probabilities
  - Emission probabilities – probability of observation given the system is in a particular state
  - Transmission probabilities – probability distribution of next state given current state

Example

- Model observed sequence of bases

$$\{Y_k\}_{k \geq 0} \in \{A, C, G, T\}$$

by two-state hidden Markov model with non-observable state binary with one corresponding to coding region and zero corresponding to non-coding region

- Codons composed of three successive symbols , thus, a higher order HMM is appropriate
  - Distribution of Y_k depends on current state X_k as well as index $k \; modulo \; 3$.
  - Alternatively, condition state probability on Y_{k_1} and Y_{k_2} in addition to Y_k

FragGeneScan

- Developed in 2010 by researches at Indiana University School of Informatics and Computing in collaboration with the Center for Genomics and Bioinformatics
- Probablistic model combines sequencing error models and codon usages to improve accuracy in predicting protein-coding regions
- Unique features
  - Finding genes fragmented by boundary of given input sequences
  - Correcting frameshifts caused by indel errors in reads

FragGeneScan Algorithm

- Viterbi algorithm – determines most likely sequence of hidden states
- Conditions
  - (i) Length of genes is no longer than 60 bp
  - (ii) Genes start in a start state or match state
  - (iii) genes end in a stop state
- Predicts complete genes as well as partial gene
- Computational complexity = $O(n)$ where n is the total length of input genomic sequences

Viterbi Algorithm

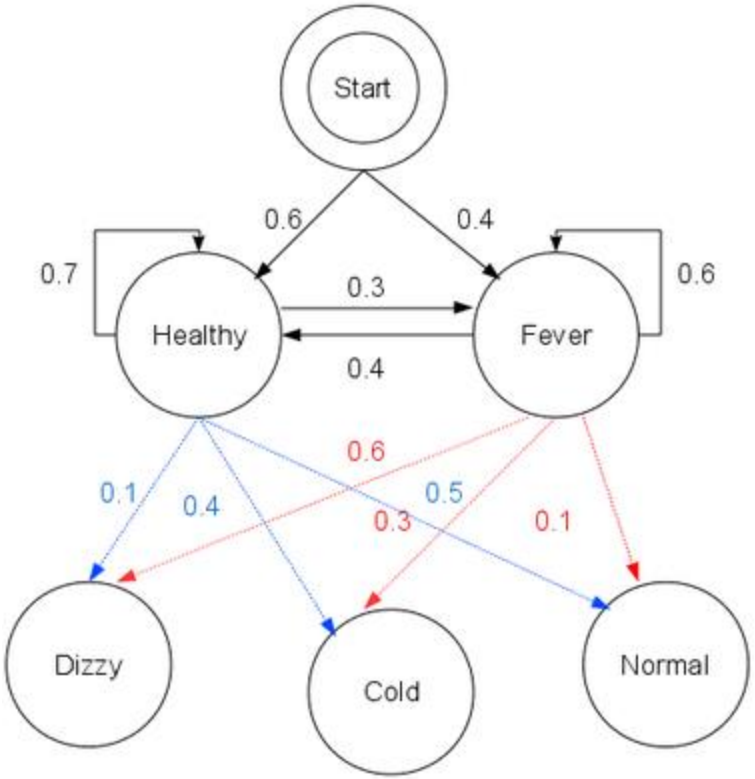
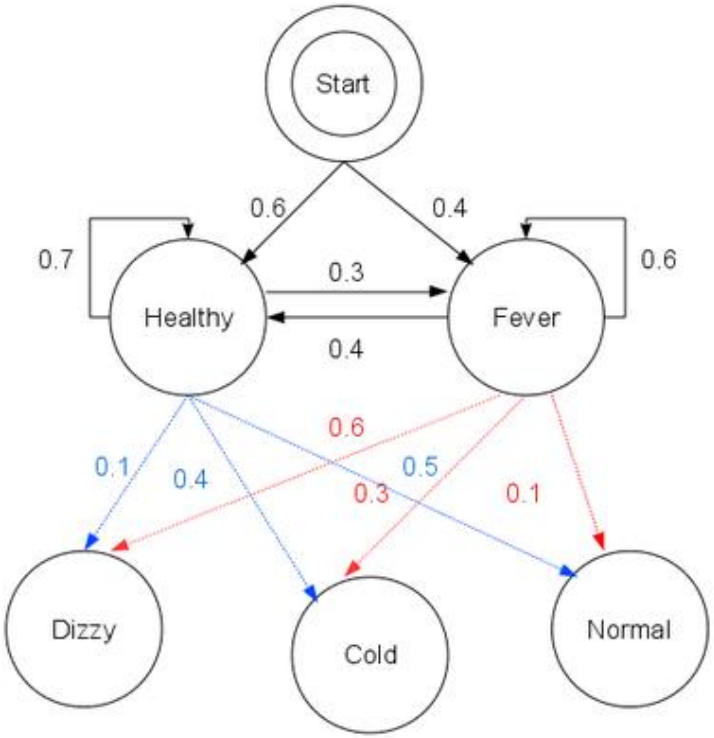


Image source - http://en.wikipedia.org/wiki/Viterbi_algorithm

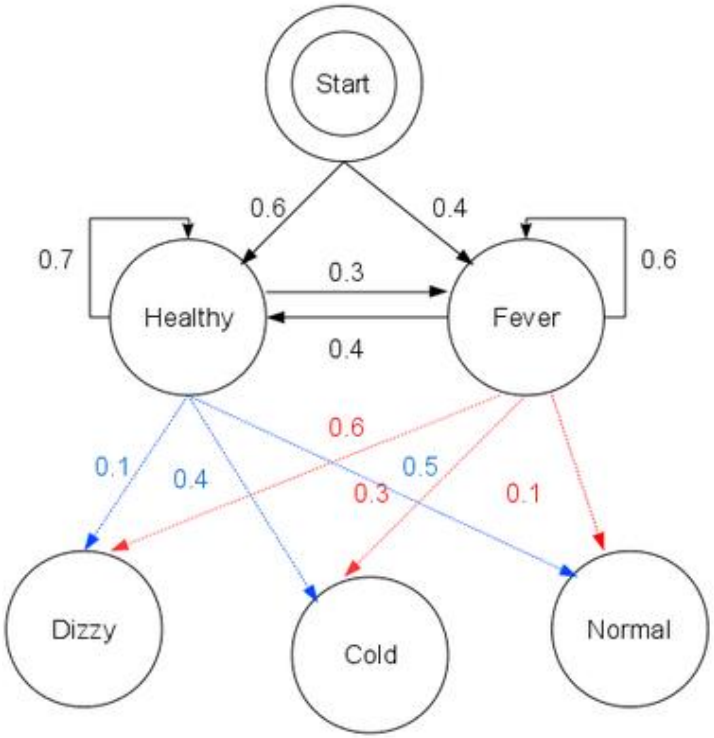
Viterbi Algorithm



- Observation matrix – {normal, cold, dizzy}
- Most likely generated by states - {Healthy, Healthy, Fever}

Image source - http://en.wikipedia.org/wiki/Viterbi_algorithm

Viterbi Algorithm



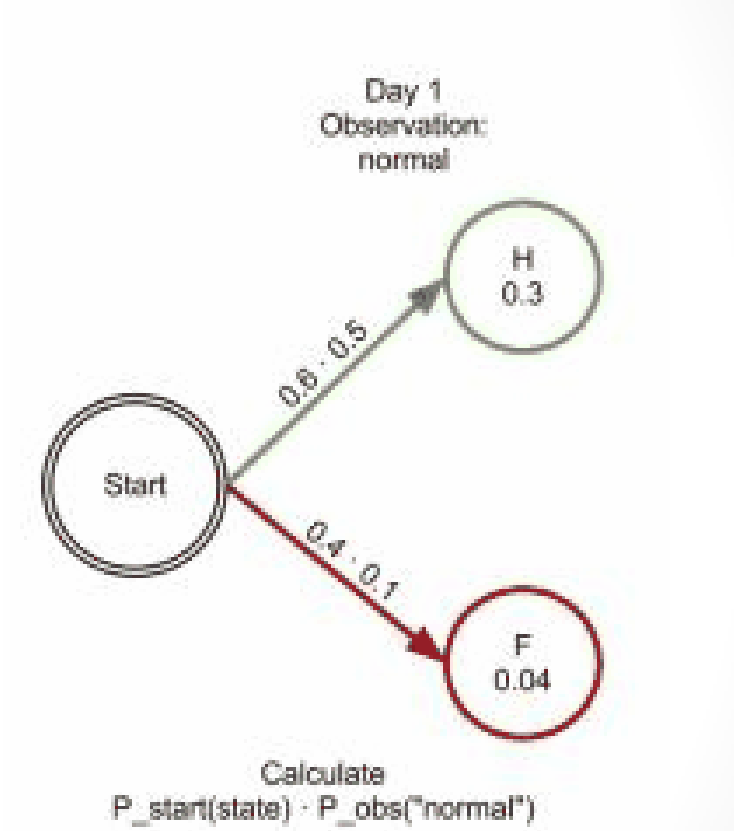
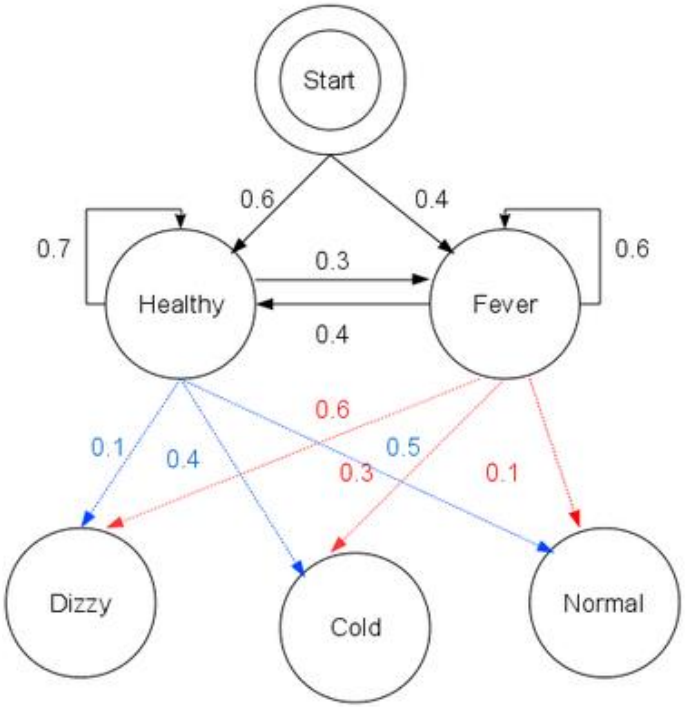


Image source -
http://en.wikipedia.org/wiki/Viterbi_algorithm

Viterbi Algorithm



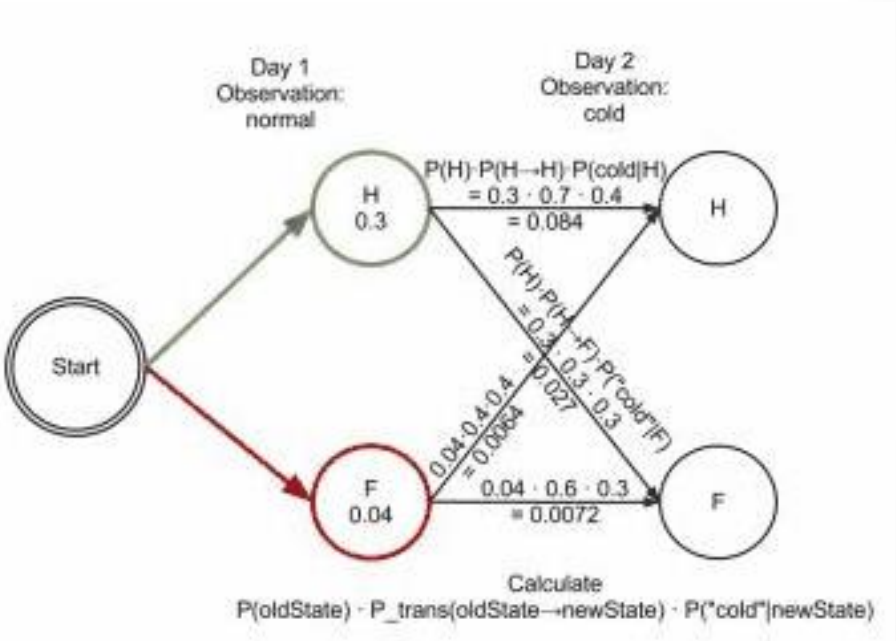
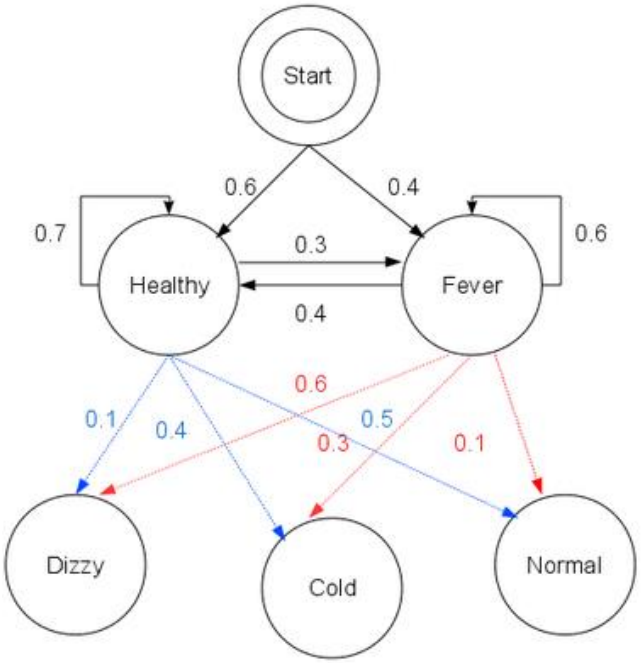


Image source -
http://en.wikipedia.org/wiki/Viterbi_algorithm

Viterbi Algorithm



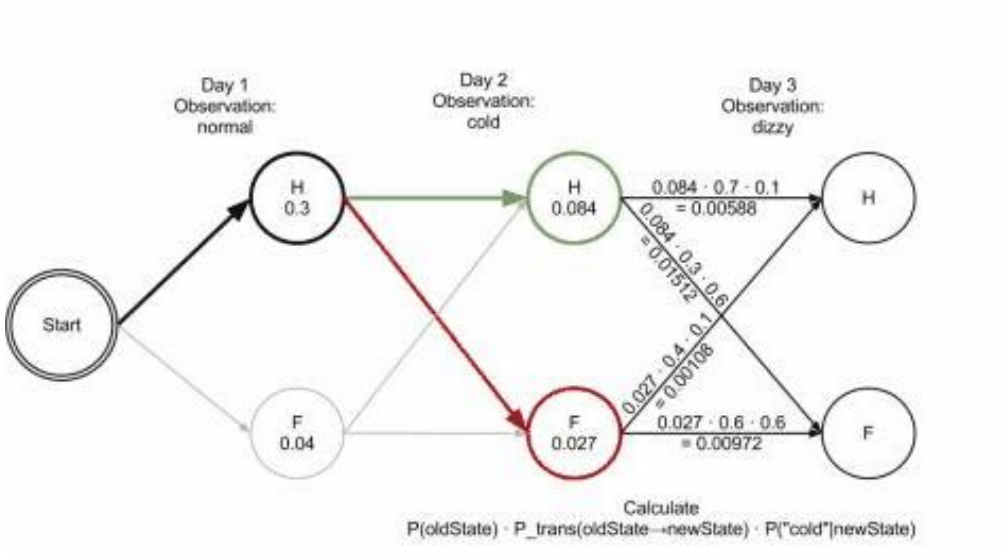
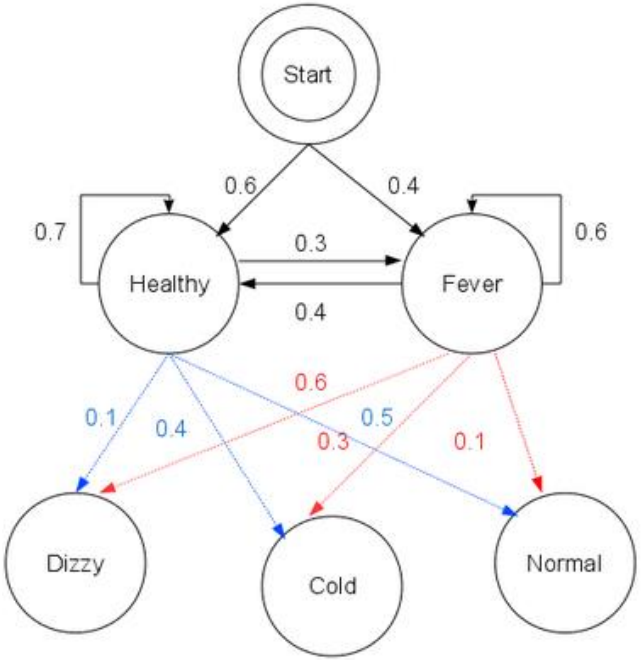


Image source - http://en.wikipedia.org/wiki/Viterbi_algorithm

Viterbi Algorithm



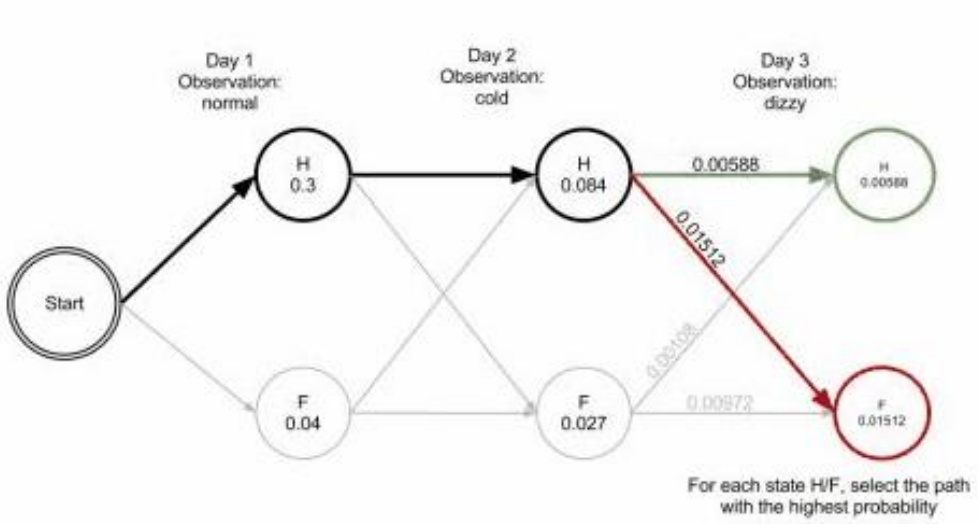


Image source -
http://en.wikipedia.org/wiki/Viterbi_algorithm

FragGeneScan Algorithm

- Start codons – ATG, GTG, TTG
- Stop codons – TAA, TAG, TGA
- Training set determines probability distribution
  - $P(TAG|stop) = 0.54, P(TAA|stop) = 0.30, P(TGA|stop) = 0.16$ [1]
- Start states modeled by positional weight matrix over 63 nucleotides centered on a putative start codon ATG, GTG, or TTG
  - $score = \sum_{i=1}^{61} \log P(trinucleotide_i|PWM)$
  - $P(trinucleotidei|PWM)$ is the probability of observing trinucleotide at position i, given the PWM of triplet frequencies (from training set)

Table 3. Gene prediction performance in short reads simulated from complete genomic sequences

Organisms	Read length (bp)[a]	FragGeneScan			MetaGene		
		Sensitivity	Specificity	Accuracy	Sensitivity	Specificity	Accuracy
B. aphidicola	100	79.16	80.12	79.64	49.59	55.24	52.41
	200	83.56	84.20	83.88	31.32	28.92	30.12
	400	84.75	81.58	83.16	17.63	13.73	15.68
	700	89.92	74.64	82.28	45.89	32.42	39.16
B. pseudomallei	100	75.79	64.78	70.28	18.64	49.63	34.14
	200	86.56	78.01	82.29	46.97	43.86	45.41
	400	90.40	82.57	86.48	31.03	25.91	28.47
	700	91.57	82.50	87.04	54.42	42.10	48.26
B. subtilis	100	72.36	65.96	69.16	31.21	55.81	43.51
	200	83.39	79.06	81.22	34.03	36.18	35.10
	400	88.24	83.51	85.88	19.83	19.25	19.54
	700	92.17	84.37	88.27	47.93	39.67	43.80
C. jeikeium	100	75.46	71.04	73.25	33.30	60.11	46.71
	200	83.75	80.93	82.34	39.65	39.27	39.46
	400	86.94	84.44	85.69	24.65	22.06	23.35
	700	90.21	85.72	87.97	49.81	39.14	44.47
C. tepidum	100	73.45	65.20	69.33	28.90	58.64	43.77
	200	81.54	77.22	79.38	40.41	40.71	40.56
	400	84.37	83.02	83.70	24.42	22.73	23.58
	700	86.51	85.86	86.19	49.33	42.55	45.94
E. coli	100	75.24	65.99	70.62	31.33	57.64	44.48
	200	85.78	78.52	82.15	39.78	37.85	38.81
	400	89.19	82.76	85.98	23.54	19.57	21.56
	700	92.86	84.19	88.53	50.97	38.26	44.62
H. pylori	100	72.69	71.69	72.19	41.94	54.58	48.26
	200	82.81	81.39	82.10	30.28	29.83	30.05
	400	84.34	78.25	81.29	17.68	15.64	16.66
	700	88.63	81.79	85.21	45.79	34.87	40.33
P. marinus	100	73.30	75.05	74.16	45.45	57.01	51.23
	200	80.00	81.39	80.69	32.04	31.01	31.52
	400	80.02	77.85	78.94	18.89	16.63	17.76
	700	86.63	82.35	84.49	47.27	36.51	41.89
W. endosymbiont	100	70.71	55.90	63.30	38.83	45.39	42.11
	200	77.56	60.10	68.83	33.23	26.81	30.02
	400	80.43	61.78	71.10	18.05	13.57	15.81
	700	86.66	61.16	73.91	47.90	31.11	39.51

- Sensitivity – ratio of true positives to all annotated genes
- Specificity – ratio of true positives to all predicted genes

Image from [1]

FragGeneScan

- Accuracy of FragGeneScan for 100 bp only 5% lower than for longer reads
- MetaGene shows 22% decrease in accuracy for shorter reads
- FragGeneScan shows consistently better performance by up to 65%
- Increased accuracy comes at expense of increased computation time

HMMER

- Project of Howard Hughes Medical Institute
- Identify homologous protein and nucleotide sequences
- Core utility for protein family databases such as Pfam and InterPro
- HMMER3 is complete rewrite of HMMER2 optimized for speed

HMMER3

- Implements a heuristic acceleration algorithm in order to optimize for speed (not for all Pfam models)
  - Limits search model with heuristic filter
  - Use of vector instructions
- Protein queries – approximately as fast as BLAST
- DNA queries – less than 10x slower than BLAST
- Utilizes Smith-Waterman algorithm (similar to Viterbi)

Smith-Watermann Algorithm

- $H(i,j) =$

$$max \begin{cases} 0 \\ H(i-1,j-1) + s(a_i, bj) \;\; Match/Mismatch \\ \max_{k \geq 1}\{H(i-k,j) + W_k\} \; Deletion \\ \max_{l \geq 1}\{(H(i,j-l) + W_l\} \; Insertion \end{cases}$$

- W is the gap-scoring scheme
- H(i,j) is the maximum similarity score
- s(a,b) is similarity function

Smith-Watermann Algorithm

- Sequence 1 = ACACACTA; sequence 2 = AGCACACA
- s(a,b) = +2 for $a = b$ and -1 for $a \neq b$

$$H = \begin{pmatrix} & - & A & C & A & C & A & C & T & A \\ - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A & 0 & 2 & 1 & 2 & 1 & 2 & 1 & 0 & 2 \\ G & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ C & 0 & 0 & 3 & 2 & 3 & 2 & 3 & 2 & 1 \\ A & 0 & 2 & 2 & 5 & 4 & 5 & 4 & 3 & 4 \\ C & 0 & 1 & 4 & 4 & 7 & 6 & 7 & 6 & 5 \\ A & 0 & 2 & 3 & 6 & 6 & 9 & 8 & 7 & 8 \\ C & 0 & 1 & 4 & 5 & 8 & 8 & 11 & 10 & 9 \\ A & 0 & 2 & 3 & 6 & 7 & 10 & 10 & 10 & 12 \end{pmatrix}$$

$$T = \begin{pmatrix} & - & A & C & A & C & A & C & T & A \\ - & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ A & 0 & \nwarrow & \leftarrow & \nwarrow & \leftarrow & \nwarrow & \leftarrow & \leftarrow & \nwarrow \\ G & 0 & \uparrow & \nwarrow & \uparrow & \nwarrow & \uparrow & \nwarrow & \nwarrow & \uparrow \\ C & 0 & \uparrow & \nwarrow & \leftarrow & \nwarrow & \leftarrow & \nwarrow & \leftarrow & \leftarrow \\ A & 0 & \nwarrow & \uparrow & \nwarrow & \leftarrow & \nwarrow & \leftarrow & \leftarrow & \nwarrow \\ C & 0 & \uparrow & \nwarrow & \uparrow & \nwarrow & \leftarrow & \nwarrow & \leftarrow & \leftarrow \\ A & 0 & \nwarrow & \uparrow & \nwarrow & \uparrow & \nwarrow & \leftarrow & \leftarrow & \nwarrow \\ C & 0 & \uparrow & \nwarrow & \uparrow & \nwarrow & \uparrow & \nwarrow & \leftarrow & \leftarrow \\ A & 0 & \nwarrow & \uparrow & \nwarrow & \uparrow & \nwarrow & \uparrow & \nwarrow & \nwarrow \end{pmatrix}$$

- Result – sequence 1 = A-CACACTA; sequence 2 = AGCACAC-A

Image source - http://en.wikipedia.org/wiki/Smith%E2%80%93Waterman_algorithm

HMMer Algorithm[2]

- Match and insert states emission probabilities learned during model estimation
- Insertions and deletions modeled by transition probabilities to them
- New algorithm limits transition possibilities to optimize for speed (from nine to seven)
- Uses Dirichlet mixture model
  - L-parameter family of probability densities over (L-1)-dimensional space
  - Mathematically convenient for multinomial space to assume prior is a Dirichlet distribution
  - Components weighted probabilistically for each column given the amino acid frequency and combined with observed frequencies

HMMER Algorithm [2]

- Small group of sequences in training sequence that are highly similar may lead to overspecialization
- Sequence weighting techniques designed to overcome this
- Sequence weighting gives outlier sequence additional importance in calculating model parameters

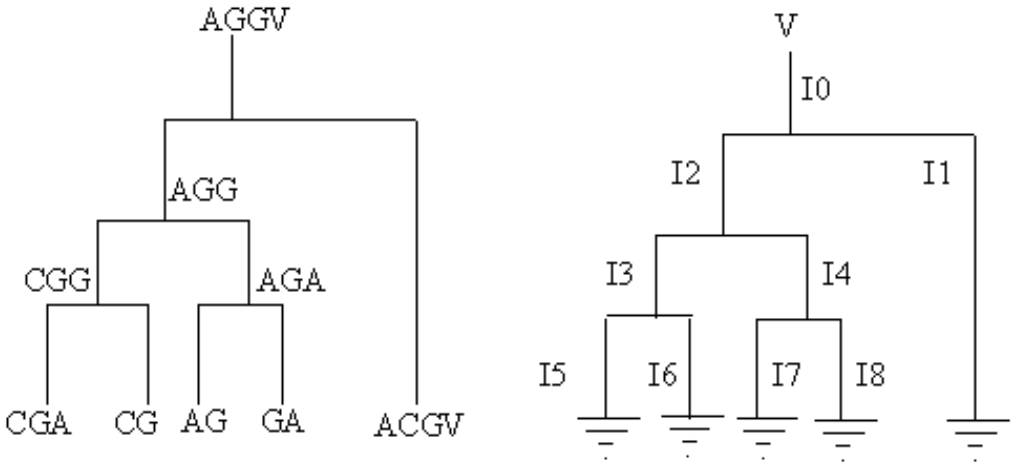


Image source - http://compbio.soe.ucsc.edu/ismb99.handouts/KK185FP.html#seq_weight

Sequence Weighting

- Weight assigned to a sequence determines its influence on final HMM
- Relative weights determined and then scaled to sum to the total weight
- HMMer groups sequences by single-linkage clustering and counts number of clusters above a specified level of identity
- Sequence can be scored locally to entire profile (global/local) or part (local/local)
  - local/local can result in multiple hits per sequence

Programs in HMMER

- Hmmalign – align sequences to existing model
- Hmmbuild – build a model from a multiple sequence alignment
- Hmmconvert – convert a model file into different formats
- Hmmemit – emit sequences probabilistically from a profile hmm
- Hmmfetch – get a single model from an HMM database
- Hmmpress – format an HMM database into a binary format for hmmscan
- Hmmscan – search a sequence against a profile HMM database

Programs in HMMer

- Hmmsim – collect score distributions on random sequences
- Hmmstat – show summary statistics for each profile in a HMM database
- Phmmer – search a sequence against a sequence database (similar to BLAST)
- Hmmsearch – search a sequence database for matches to an HMM
- Jackhmmer – iteratively search a sequence against a database

Output

Significant Query Matches (9)				Customize
	Target	Description	Species	E-value
>	899452	beta2-chimaerin	Homo sapiens	3.7e-08
>	261861448	chimerin (chimaerin) 2	synthetic construct	3.7e-08
>	332864983	PREDICTED: beta-chimaerin isoform 1	Pan troglodytes	3.7e-08
>	296209338	PREDICTED: beta-chimaerin	Callithrix jacchus	3.7e-08
>	297680753	PREDICTED: beta-chimaerin-like isoform 2	Pongo abelii	3.7e-08
>	18376256	conserved hypothetical protein	Neurospora crassa	3.5e-06
>	38258908	RecName: Full=Myosin ID heavy chain	Dictyostelium discoideum	1.2e-05
>	51094646	growth factor receptor-bound protein 10	Homo sapiens	0.00022
>	13925747	MAP kinase pathway-interacting Ubc2	Ustilago maydis	0.00064
>	7597003	cell division cycle protein	Candida albicans	0.017
>	74876138	RecName: Full=SH3 and FCH domain-containing protein DDB_G0271676	Dictyostelium discoideum	0.083

(show all) alignments

Your search took:0.95 secs

showing rows 1 - 11 of 11

Output

Query		Target Envelope		Target Alignment		Bias	Accuracy	% Identity (count)	% Similarity (count)	Bit Score	E-value	
start	end	start	end	start	end						Ind.	Cond.
73	156	40	138	60	133	0.02	0.87	41.9 (31)	68.9 (51)	34.8	1.3e-06	1.6e-11

- Query start/end – start/end of the maximum expected accuracy (MEA) alignment with respect to the profile HMM
- Target Envelope – defines a subsequence for which there is substantial probability supporting a homologous domain/hit
- Target Alignment – start/end of MEA alignment of this domain with respect to the target sequence
- Bias – bias composition correction is bit score difference contributed by null2 model; high bias scores represent potential false positives
- Accuracy – measure of the reliability of the overall alignment
- % Identity – precentage of identical residues between query and target
- % similarity – similar to identity but using sum of identical and similar residues

References

- [1] Rho, Mina, Haixu Tang, and Yuzhen Ye. "FragGeneScan: predicting genes in short and error-prone reads." *Nucleic acids research* 38.20 (2010): e191-e191.
- [2] Wistrand, Markus, and Erik LL Sonnhammer. "Improved profile HMM performance by assessment of critical algorithmic features in SAM and HMMER." *BMC bioinformatics* 6.1 (2005): 99.
- [3] Sinha, Swati, and Andrew Michael Lynn. "HMM-ModE: implementation, benchmarking and validation with HMMER3." *BMC research notes* 7.1 (2014): 483.
- [4] http://hmmer.janelia.org/help

For further information see

- https://github.com/ericJmarti/ECES490-Tutorial-9/blob/master/README.md