CSE 527

Markov Models and Hidden Markov Models

http://upload.wikimedia.org/wikipedia/commons/b/ba/Calico_cat

Dosage Compensation and X-Inactivation

2 copies (mom/dad) of each chromosome 1-23

Mostly, both copies of each gene are expressed

> E.g., A B O blood group defined by 2 alleles of 1 gene

Women (XX) get double dose of X genes (vs XY)?

So, early in embryogenesis:

- One X randomly inactivated in each cell
- Choice maintained in daughter cells

How?

Calico: major coat color gene is on X

Reminder: Proteins "Read" DNA

E.g.:

recognition helix
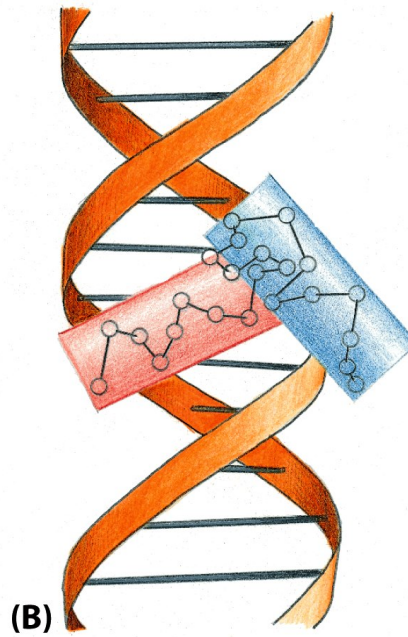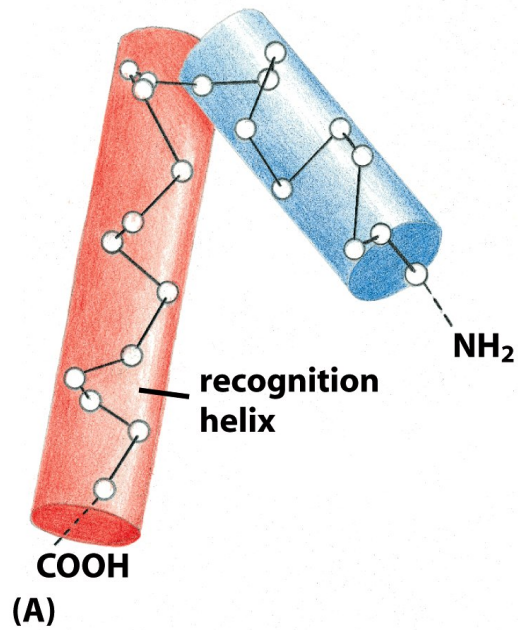
NH₂

COOH

(A)

(B)

Figure 7-10 Molecular Biology of the Cell 5/e (© Garland Science 2008)

Figure 7-19 Molecular Biology of the Cell 5/e (© Garland Science 2008)

Down in the Groove

Different patterns of hydrophobic methyls, potential H bonds, etc. at edges of different base pairs. They're accessible, esp. in major groove
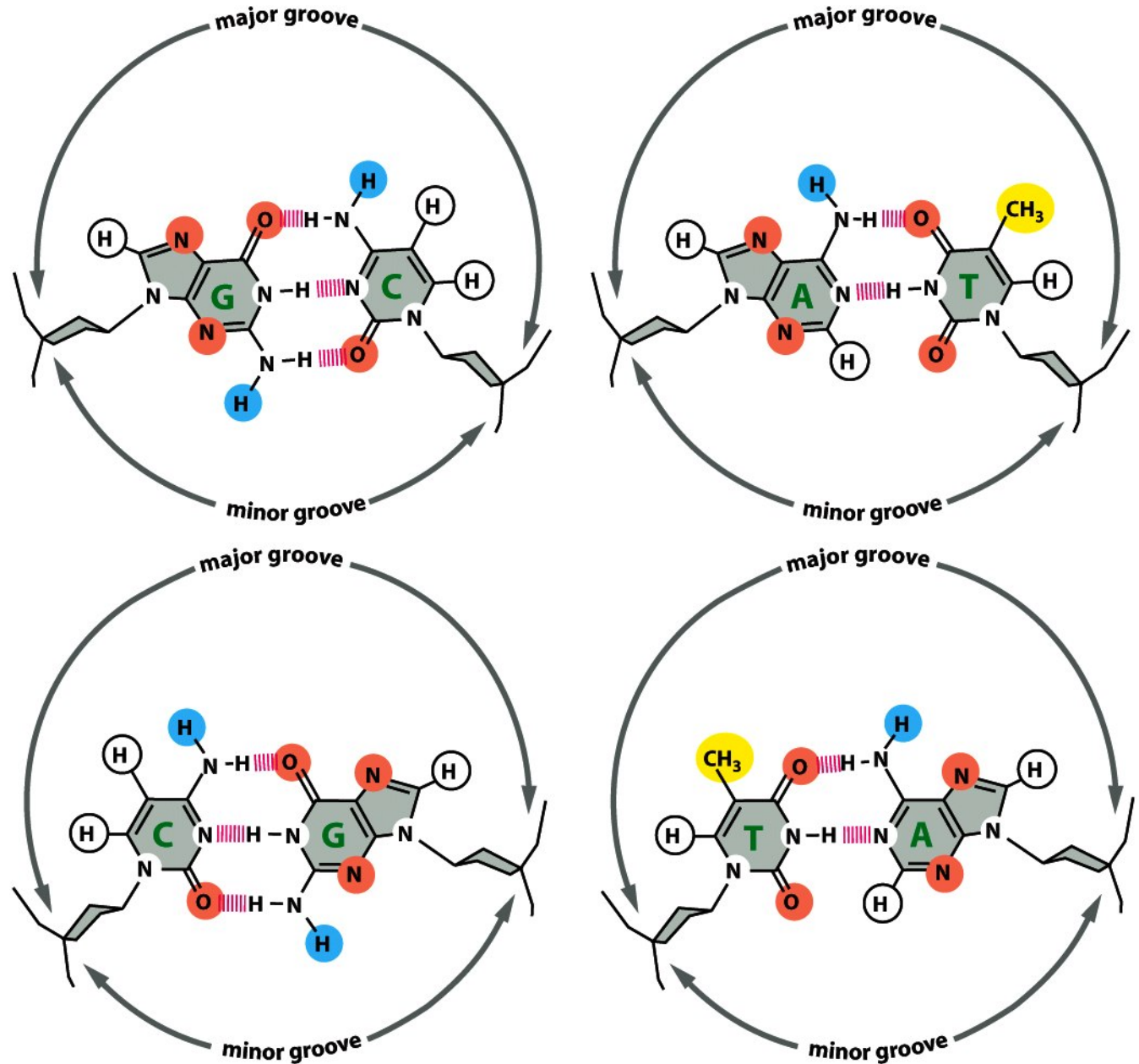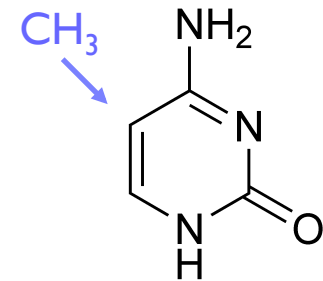

Figure 7-7 Molecular Biology of the Cell 5/e (© Garland Science 2008)

DNA Methylation

CpG - 2 adjacent nts, same strand
(not Watson-Crick pair; "p" mnemonic for the phosphodiester bond of the DNA backbone)

C of CpG is often (70-80%) methylated in mammals i.e., CH_3 group added (both strands)

cytosine

Same Pairing

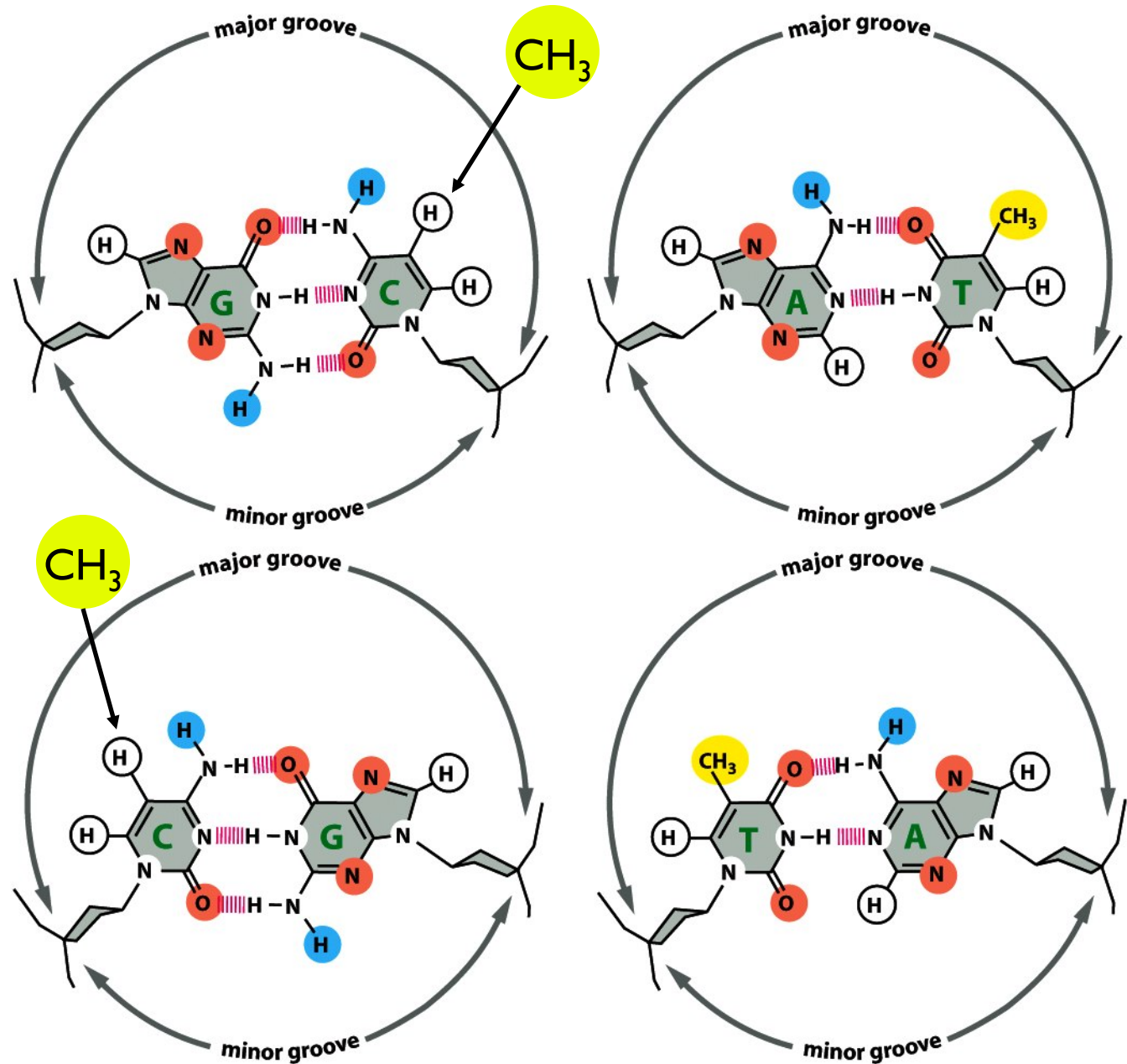Methyl-C alters major groove profile (∴ TF binding), but not base-pairing, transcription or replication

Figure 7-7 Molecular Biology of the Cell 5/e (© Garland Science 2008)

DNA Methylation–Why

In vertebrates, it generally silences transcription

(Epigenetics) X-inactivation, imprinting, repression of mobile elements, cancers, aging, and developmental differentiation

E.g., if a stem cell divides, one daughter fated to be liver, other kidney, need to

(a) turn off liver genes in kidney & vice versa,

(b) remember that through subsequent divisions

How?

(a) Methylate genes, esp. promoters, to silence them

(b) after ÷, DNA methyltransferases convert hemi- to fully-methylated (& deletion of methyltransferse is embrionic-lethal in mice)

Major exception: promoters of housekeeping genes

CH_3 NH_2
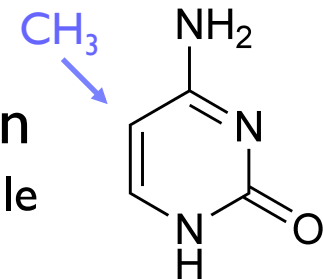
cytosine

"CpG Islands"

Methyl-C mutates to T relatively easily

Net: CpG is less common than expected genome-wide: $f(CpG) < f(C)*f(G)$

BUT in some regions (e.g. active promoters), CpG remain unmethylated, so CpG → TpG less likely there: makes "CpG Islands"; often mark gene-rich regions
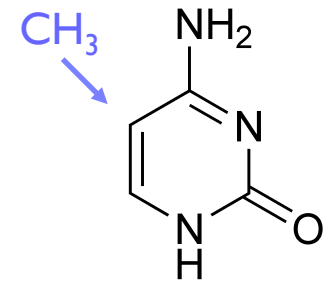
cytosine
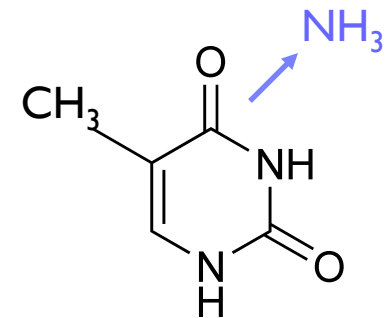
thymine

CpG Islands

CpG Islands

- More CpG than elsewhere (say, CpG/GpC>50%)
- More C & G than elsewhere, too (say, C+G>50%)
- Typical length: few 100 to few 1000 bp

Questions

- Is a short sequence (say, 200 bp) a CpG island or not?
- Given long sequence (say, 10-100kb), find CpG islands?

Markov & Hidden Markov Models

References (see also online reading page):

Eddy, "What is a hidden Markov model?" Nature Biotechnology, 22, #10 (2004) 1315-6.

Durbin, Eddy, Krogh and Mitchison, "Biological Sequence Analysis", Cambridge, 1998 (esp. chs 3, 5)

Rabiner, "A Tutorial on Hidden Markov Models and Selected Application in Speech Recognition," Proceedings of the IEEE, v 77 #2, Feb 1989, 257-286

Independence

A key issue: Previous models we've talked about assume *independence* of nucleotides in different positions - definitely unrealistic.

Markov Chains

A sequence $x_1, x_2, \ldots$ of random variables is a *k-th order Markov chain* if, for all i, i^{th} value is independent of all but the previous k values:

$$P(x_i \mid x_1, x_2, \ldots, x_{i-1}) = P(x_i \mid x_{i-k}, x_{i-k+1}, \ldots, x_{i-1})$$

Example 1: Uniform random ACGT   } 0th order
Example 2: Weight matrix model

Example 3: ACGT, but ↓ Pr(G following C)   } 1st order

A Markov Model (1st order)
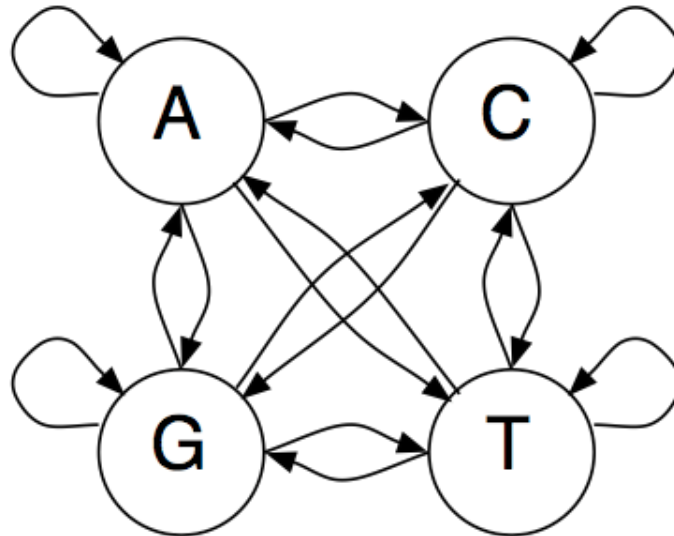
States: A,C,G,T
Emissions: corresponding letter
Transitions: $a_{st} = P(x_i = t \mid x_{i-1} = s)$ ⟵ 1st order

A Markov Model (1st order)
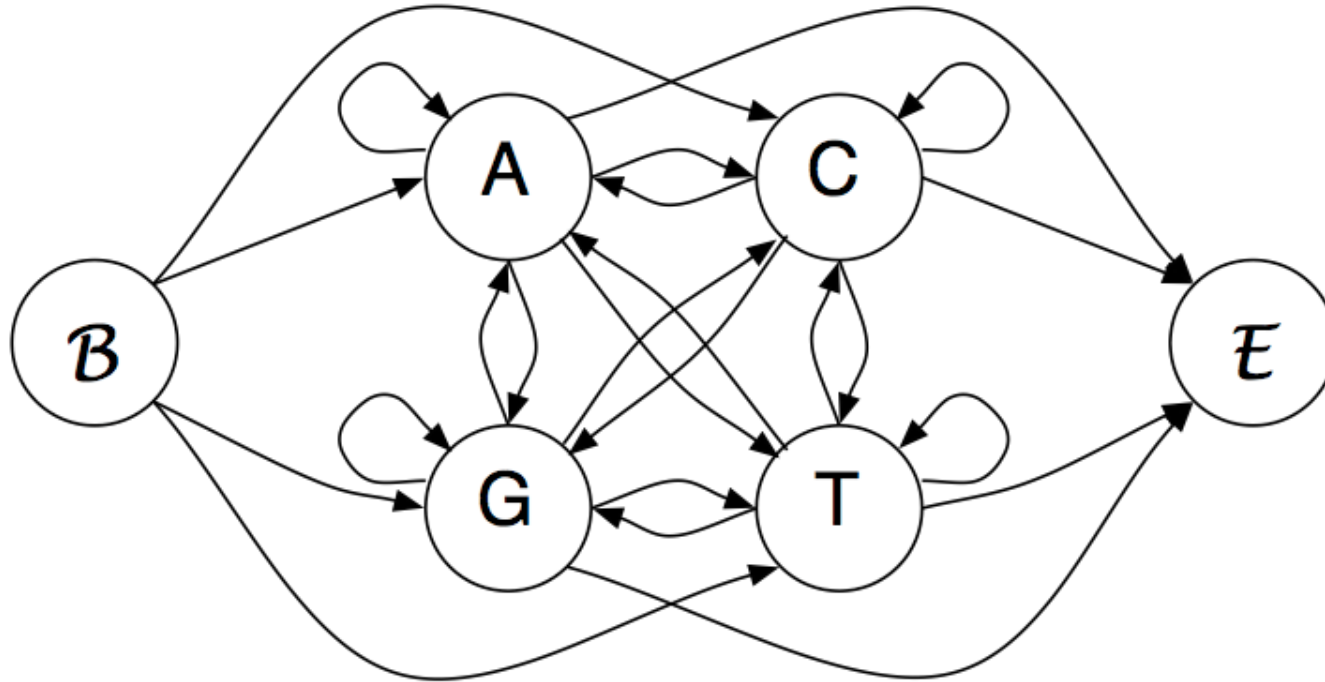
States: A,C,G,T
Emissions: corresponding letter
Transitions: $a_{st} = P(x_i = t \mid x_{i-1} = s)$
$\mathcal{B}$egin/$\mathcal{E}$nd states

Pr of emitting sequence *x*

$$x \quad = \quad x_1 \, x_2 \, \ldots \, x_n$$

$$P(x) \quad = \quad P(x_1, x_2, \ldots, x_n)$$

> laws of probability

$$= \quad P(x_1) \cdot P(x_2 \mid x_1) \cdots P(x_n \mid x_{n-1}, \ldots, x_1)$$

> if 1st order MC

$$= \quad P(x_1) \cdot P(x_2 \mid x_1) \cdots P(x_n \mid x_{n-1})$$

$$= \quad P(x_1) \prod_{i=1}^{n-1} a_{x_i, x_{i+1}}$$

$$= \quad \prod_{i=0}^{n-1} a_{x_i, x_{i+1}} \quad \text{(with Begin state)}$$

Training

Max likelihood estimates for transition probabilities are just the frequencies of transitions when emitting the training sequences

E.g., from 48 CpG islands in 60k bp:

+	A	C	G	T		-	A	C	G	T
A	0.180	0.274	0.426	0.120		A	0.300	0.205	0.285	0.210
C	0.171	0.368	0.274	0.188		C	0.322	0.298	0.078	0.302
G	0.161	0.339	0.375	0.125		G	0.248	0.246	0.298	0.208
T	0.079	0.355	0.384	0.182		T	0.177	0.239	0.292	0.292

From DEKM

Discrimination/Classification

Log likelihood ratio of CpG model vs background model

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

β	A	C	G	T
A	−0.740	0.419	0.580	−0.803
C	−0.913	0.302	1.812	−0.685
G	−0.624	0.461	0.331	−0.730
T	−1.169	0.573	0.393	−0.679

From DEKM

CpG Island Scores
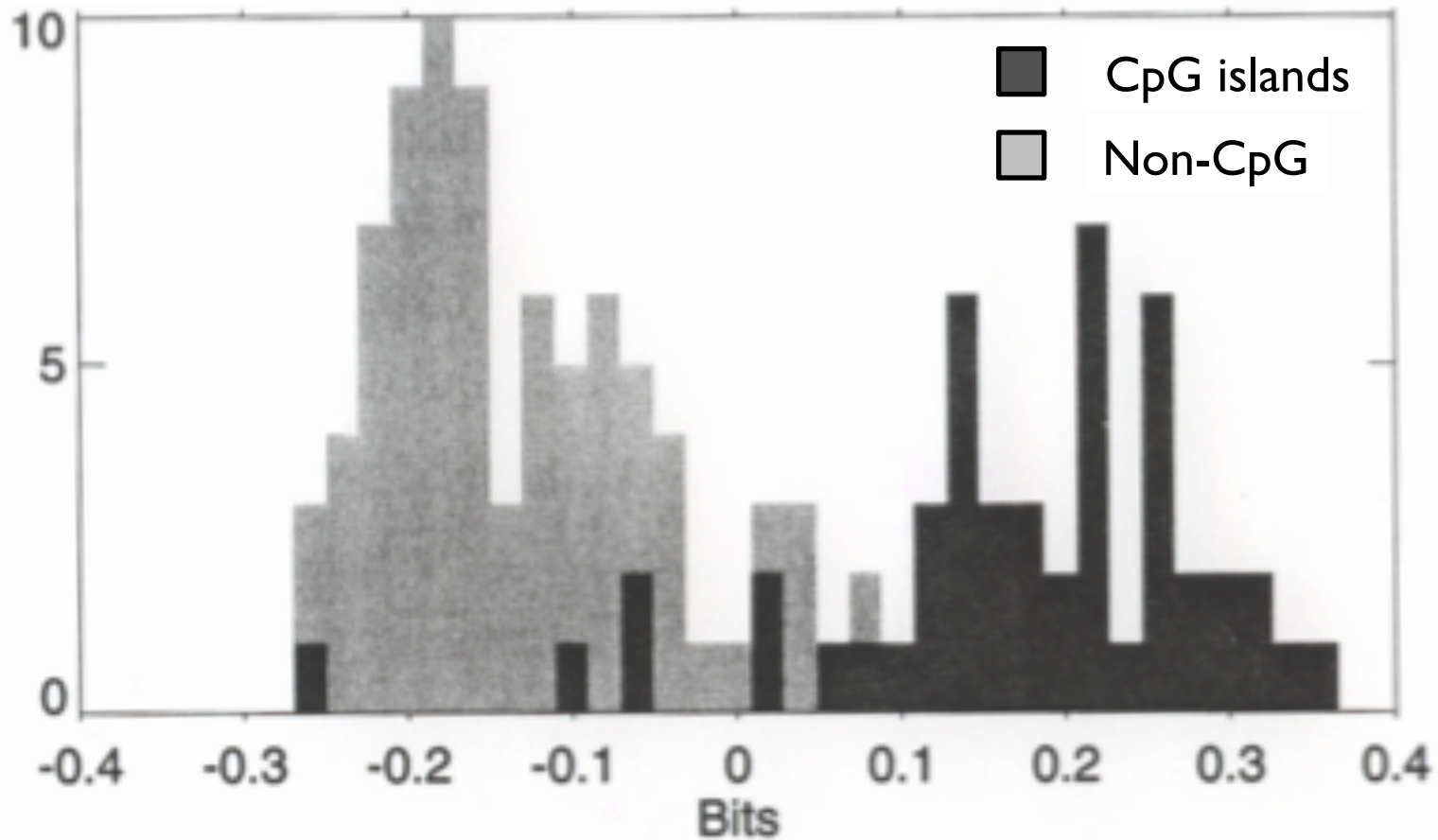
Figure 3.2 *Histogram of length-normalized scores.*

What does a 2nd order
Markov Model look like?

3rd order?

Questions

Q1: Given a *short* sequence, is it more likely from feature model or background model? Above

Q2: Given a *long* sequence, where are the features in it (if any)

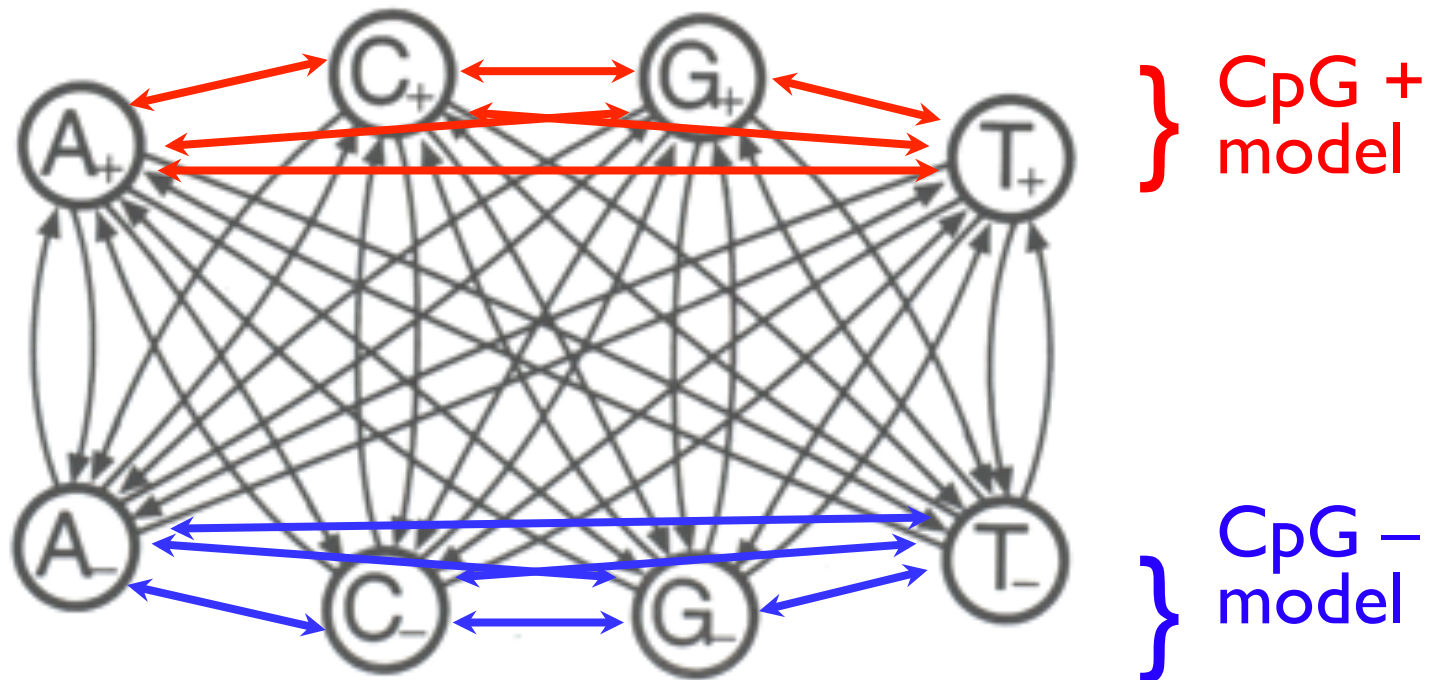 Approach 1: score 100 bp (e.g.) windows

    Pro: simple

    Con: arbitrary, fixed length, inflexible

  Approach 2: *combine* +/- models.

Combined Model

CpG + model

CpG – model

Emphasis is "Which (hidden) state?" not "Which model?"

Hidden Markov Models
(HMMs; Claude Shannon, 1948)

States:	$1, 2, 3, \ldots$
Paths:	sequences of states $\pi = (\pi_1, \pi_2, \ldots)$
Transitions:	$a_{k,l} = P(\pi_i = l \mid \pi_{i-1} = k)$
Emissions:	$e_k(b) = P(x_i = b \mid \pi_i = k)$
Observed data:	emission sequence
Hidden data:	state/transition sequence

The Occasionally Dishonest Casino

1 fair die, 1 "loaded" die, occasionally swapped

```
Rolls     315116246446644245311321631164152133625145436316566256666
Die       FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLL

Rolls     651166453132651245636664631636663162326455236266666625151631
Die       LLLLLLFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLLFFFFFFFFF
Viterbi   LLLLLLFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFF

Rolls     222555441666566563564324364131513465146353411126414626253356
Die       FFFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls     366163666466232534413661661163252562462255265252266435353336
Die       LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi   LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls     233121625364414432335163243633665562466626326666612355245242
Die       FFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFFFF
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
```

Figure 3.5

Rolls: Visible data–300 rolls of a die as described above.
Die: Hidden data–which die was actually used for that roll (F = fair, L = loaded).
Viterbi: the prediction by the Viterbi algorithm is shown.

Inferring hidden stuff

Joint probability of a given path π & emission sequence *x:*

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

But π is hidden; what to do? Some alternatives:

Most probable single path

$$\pi^* = \arg\max_{\pi} P(x, \pi)$$

Sequence of most probable states

$$\hat{\pi}_i = \arg\max_{k} P(\pi_i = k \mid x)$$

The Viterbi Algorithm: The most probable path

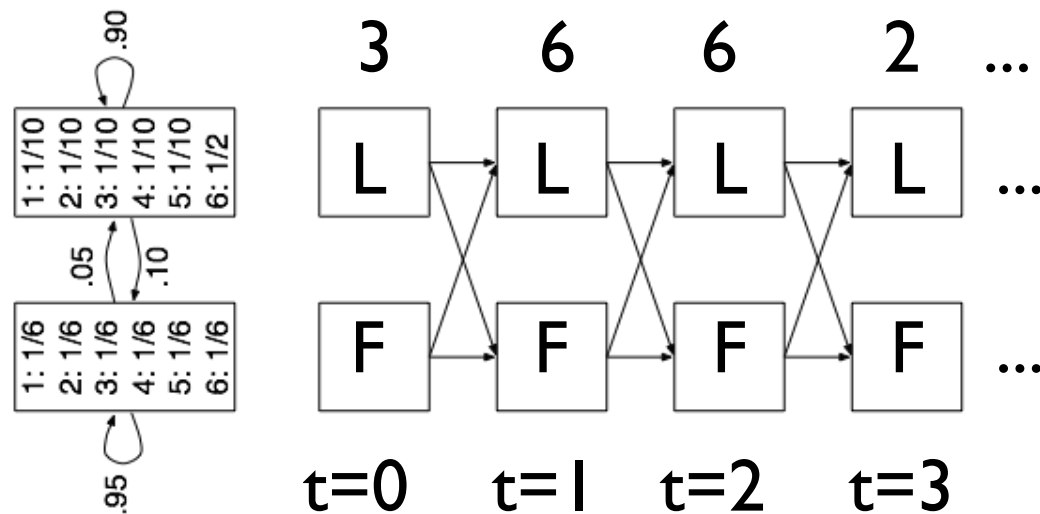Viterbi finds: $\qquad \pi^* = \arg \max_{\pi} P(x, \pi)$

Possibly there are 10^{99} paths of prob 10^{-99}

More commonly, one path (+ slight variants)
dominate others.
(If not, other approaches may be preferable.)

Key problem: exponentially many paths π

Unrolling an HMM

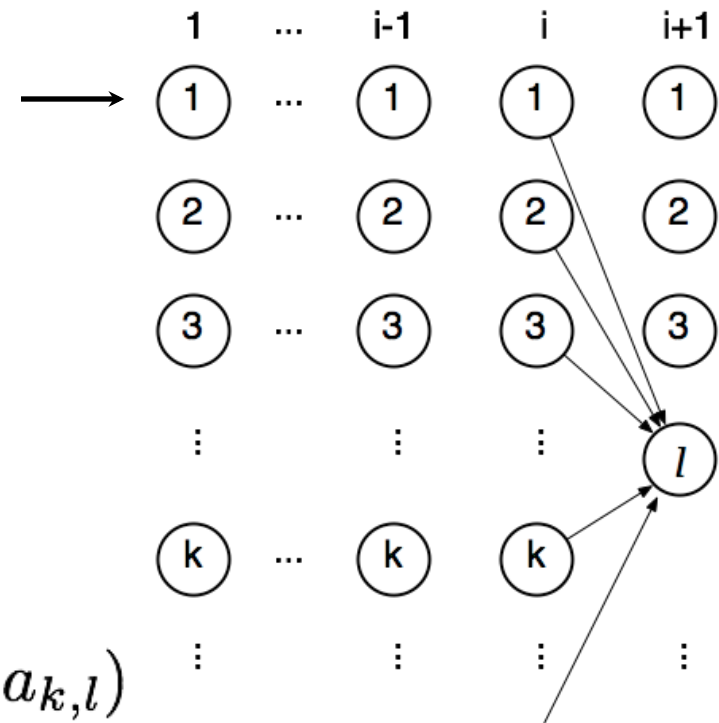

Conceptually, sometimes convenient

Note exponentially many paths

Viterbi

$v_l(i)$ = probability of the most probable path emitting $x_1, x_2, \ldots, x_i$ and ending in state l

Initialize:

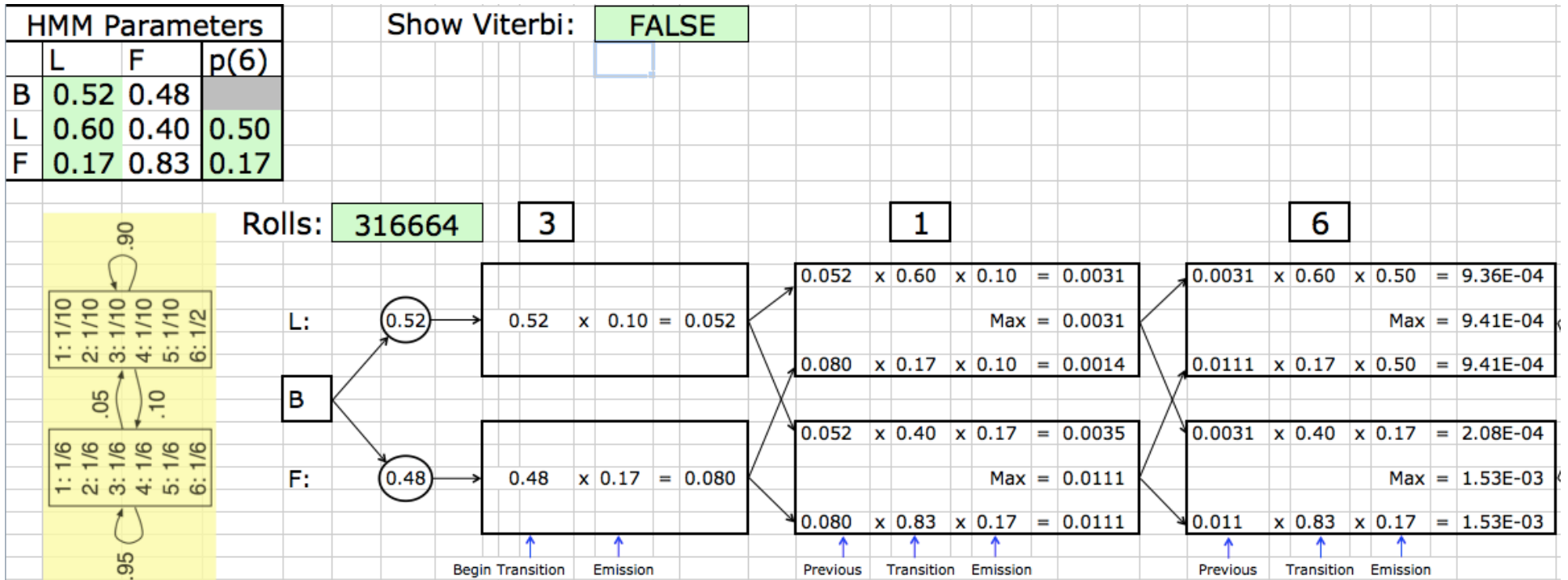$$v_l(0) = \begin{cases} 1 & \text{if } l = Begin \text{ state} \\ 0 & \text{otherwise} \end{cases}$$

General case:

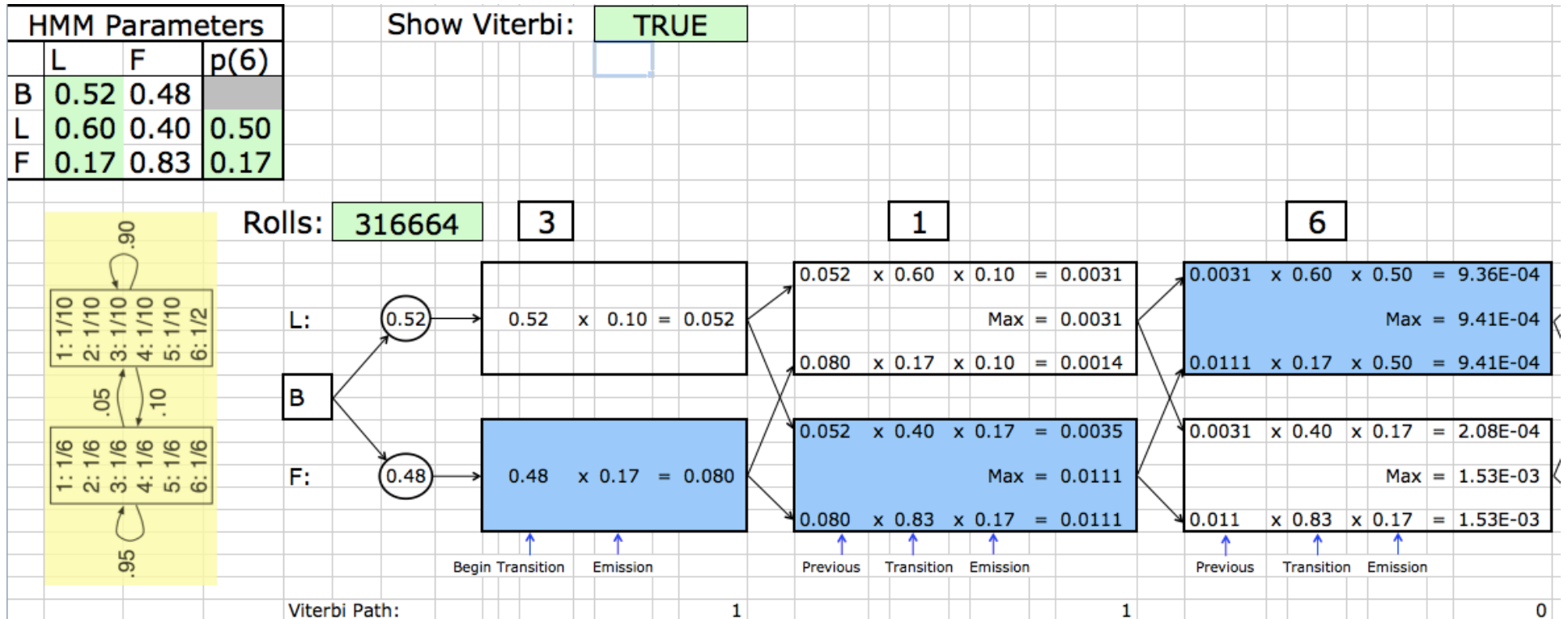$$v_l(i+1) = e_l(x_{i+1}) \cdot \max_k(v_k(i)\, a_{k,l})$$

HMM Casino Example

(Excel spreadsheet on web; download & play…)

HMM Casino Example

HMM Parameters		
	L	F	p(6)
B	0.52	0.48	
L	0.60	0.40	0.50
F	0.17	0.83	0.17

Show Viterbi: **TRUE**

Rolls: **316664** | 3 | | 1 | | 6 |

L: (0.52) → 0.52 x 0.10 = 0.052

0.052 x 0.60 x 0.10 = 0.0031
Max = 0.0031
0.080 x 0.17 x 0.10 = 0.0014

0.0031 x 0.60 x 0.50 = 9.36E-04
Max = 9.41E-04
0.0111 x 0.17 x 0.50 = 9.41E-04

B

F: (0.48) → 0.48 x 0.17 = 0.080

0.052 x 0.40 x 0.17 = 0.0035
Max = 0.0111
0.080 x 0.83 x 0.17 = 0.0111

0.0031 x 0.40 x 0.17 = 2.08E-04
Max = 1.53E-03
0.011 x 0.83 x 0.17 = 1.53E-03

Begin Transition   Emission   Previous   Transition   Emission   Previous   Transition   Emission

Viterbi Path: 1   1   0

.90
1: 1/10 2: 1/10 3: 1/10 4: 1/10 5: 1/10 6: 1/2
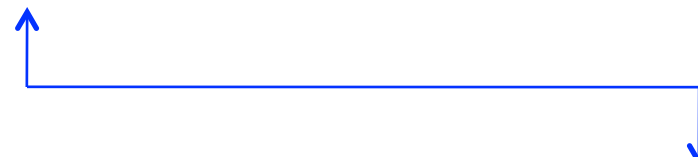.05   .10
1: 1/6 2: 1/6 3: 1/6 4: 1/6 5: 1/6 6: 1/6
.95

(Excel spreadsheet on web; download & play…)

Viterbi Traceback

Above finds *probability* of best path

To find the path itself, trace *backward* to the state *k* attaining the max at each stage

$$v_l(i+1) = e_l(x_{i+1}) \cdot \max_k (v_k(i)\, a_{k,l})$$

```
Rolls      315116246446644245311321631164152133625144543631656626566666
Die        FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLL

Rolls      651166453132651245636664631636663162326455236266666625151631
Die        LLLLLLFFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLLFFFFFFFFF
Viterbi    LLLLLLFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFF

Rolls      222555441666566563564324364131513465146353411126414626253356
Die        FFFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls      366163666466232534413661661163252562462255265252266435353336
Die        LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi    LLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls      233121625364414432335163243633665562466662632666612355245242
Die        FFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
Viterbi    FFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
```

Figure 3.5

Rolls: Visible data–300 rolls of a die as described above.

Die: Hidden data–which die was actually used for that roll (F = fair, L = loaded).

Viterbi: the prediction by the Viterbi algorithm is shown.

Most probable path ≠ Sequence of most probable states
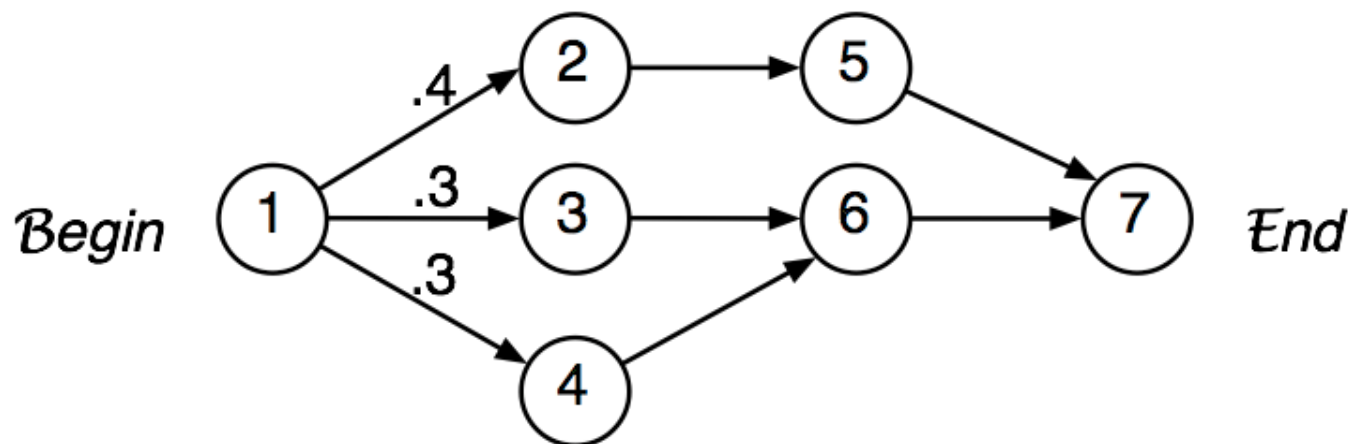
Another example, based on casino dice again

Suppose p(fair↔loaded) transitions are 10^{-99} and roll sequence is 11111…66666; then fair state is more likely all through 1's & well into the run of 6's, but eventually loaded wins, and the improbable F→L transitions make Viterbi = *all* L.
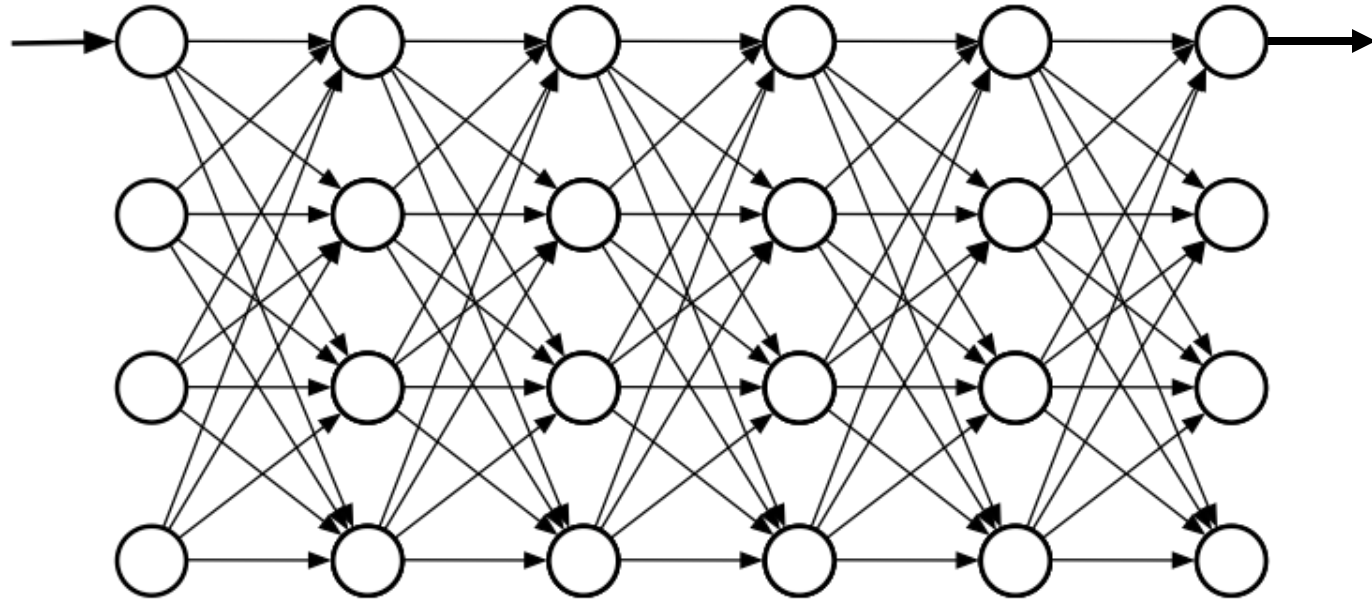

* = max prob
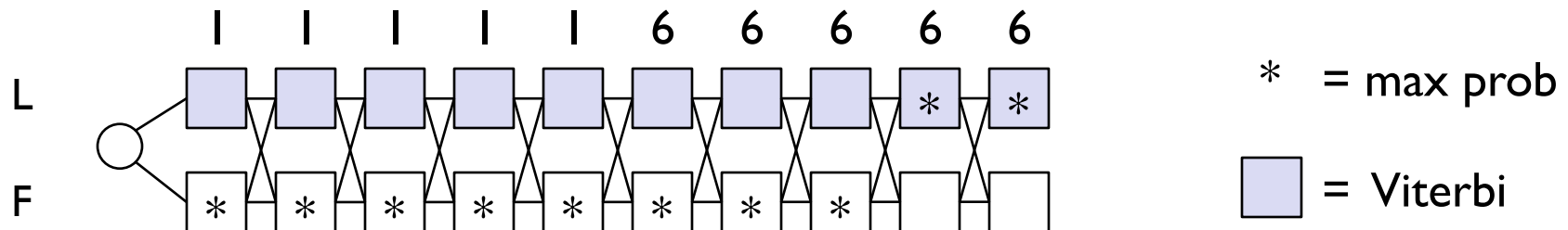
☐ = Viterbi

Is Viterbi "best"?

Viterbi finds $\pi^* = \arg\max_{\pi} P(x, \pi)$

Most probable (Viterbi) *path* goes through 5, but most probable *state* at 2nd step is 6
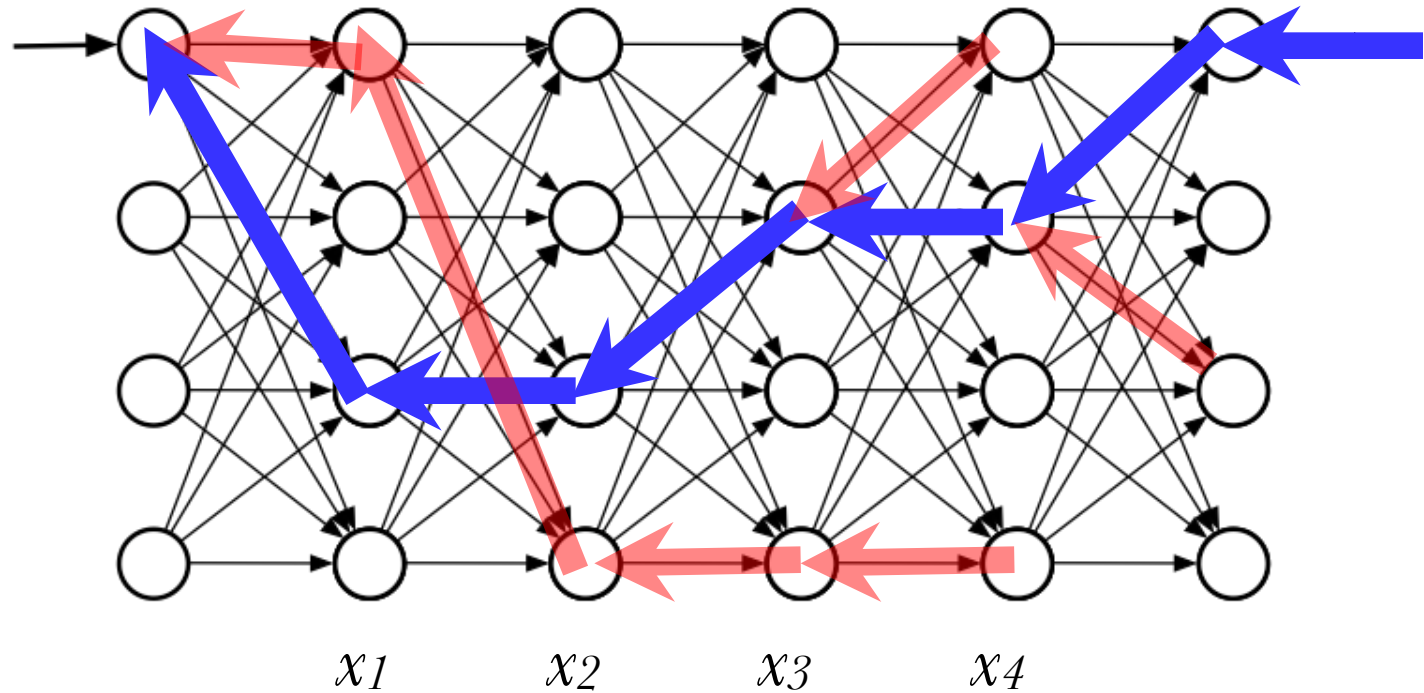(I.e., Viterbi is not the only interesting answer.)

An HMM (unrolled)

States

x_1 x_2 x_3 x_4

Emissions/sequence positions ⟶

Viterbi: best path to each state

$$x_1 \qquad x_2 \qquad x_3 \qquad x_4$$

Viterbi *score:*
$$v_l(i+1) = e_l(x_{i+1}) \cdot \max_k (v_k(i)\, a_{k,l})$$

Viterbi *path^R:*
$$back_l(i+1) = \arg\max_k (v_k(i)\, a_{k,l})$$

The Forward Algorithm

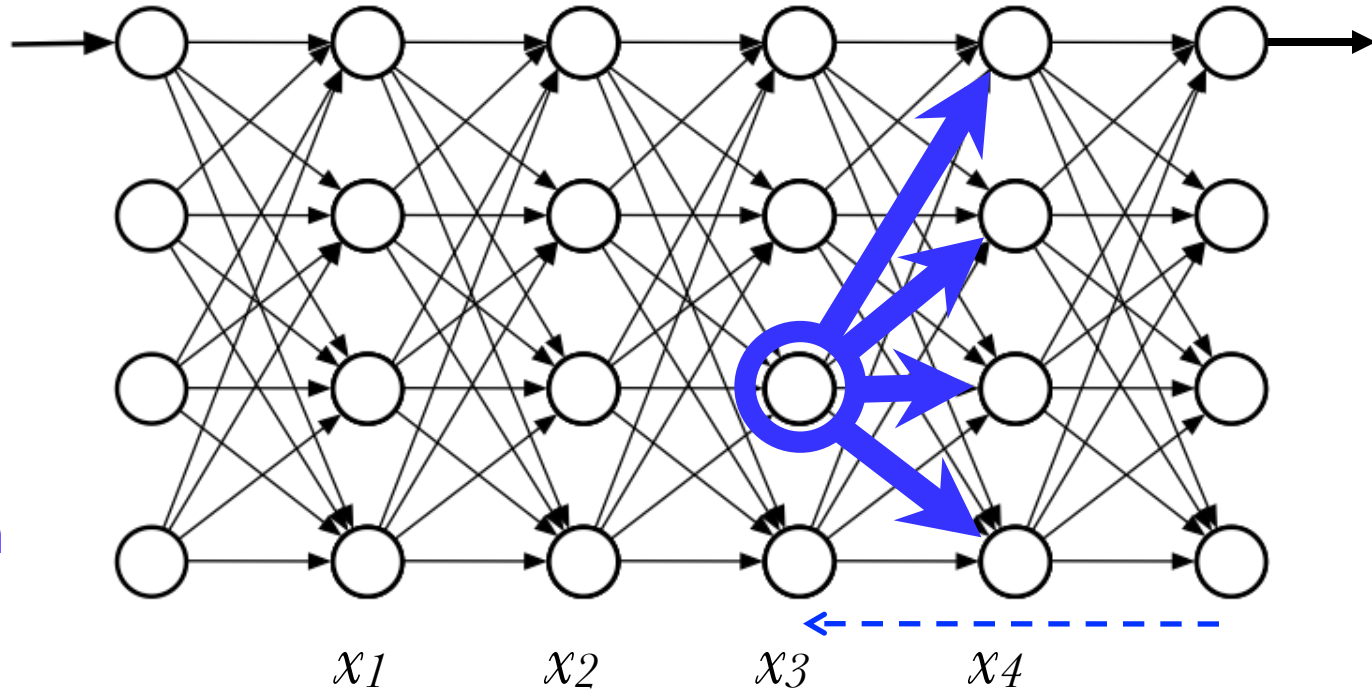For each state/time, want *total* probability of all paths leading *to* it, with given emissions

x_1 x_2 x_3 x_4
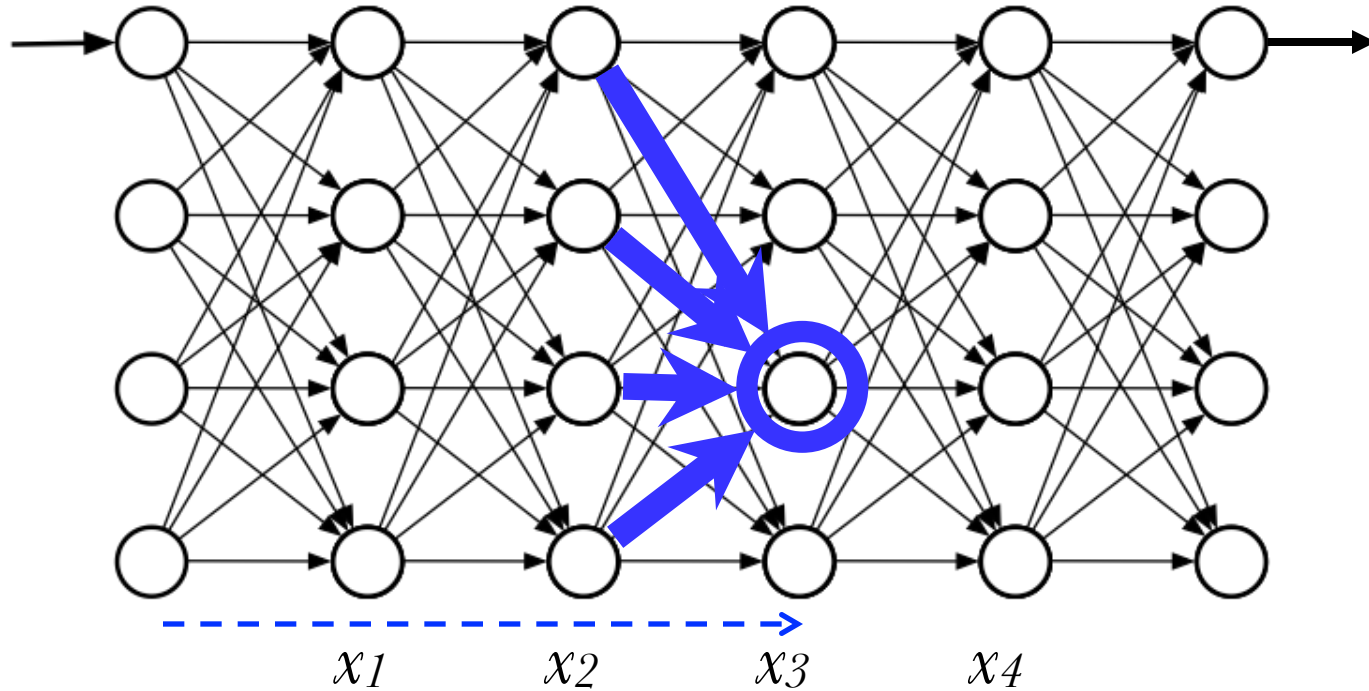
$$f_k(i) \quad \stackrel{\triangle}{=} \quad P(x_1 \ldots x_i, \ \pi_i = k)$$

$$f_l(i+1) \quad = \quad e_l(x_{i+1}) \sum_k f_k(i) a_{k,l}$$

$$P(x) \quad = \quad \sum_\pi P(x, \pi) \ = \ \sum_k f_k(n) a_{k,0}$$

The Backward Algorithm

Similar: for each state/time, want total probability of all paths *from* it, with given emissions, *conditional* on that state.

$$x_1 \qquad x_2 \qquad x_3 \qquad x_4$$

$$b_k(i) \quad \overset{\triangle}{=} \quad P(x_{i+1} \cdots x_n \mid \pi_i = k)$$

$$b_k(i) \quad = \quad \sum_l a_{k,l} \, e_l(x_{i+1}) \, b_l(i+1)$$

$$b_k(n) \quad = \quad a_{k,0}$$

In state *k* at step *i* ?

$$P(x, \pi_i = k)$$

$$= P(x_1, \ldots, x_i, \pi_i = k) \cdot P(x_{i+1}, \ldots, x_n \mid x_1, \ldots, x_i, \pi_i = k)$$

$$= P(x_1, \ldots, x_i, \pi_i = k) \cdot P(x_{i+1}, \ldots, x_n \mid \pi_i = k)$$
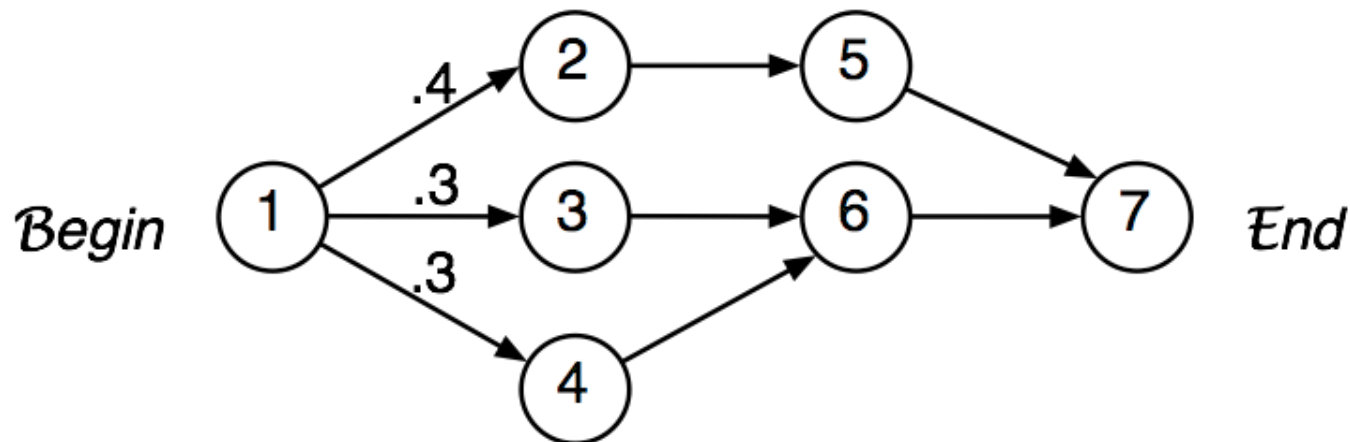
$$= f_k(i) \cdot b_k(i)$$

$$P(\pi_i = k \mid x) = \frac{P(x, \pi_i = k)}{P(x)} = \frac{f_k(i) \cdot b_k(i)}{P(x)}$$

Posterior Decoding, I
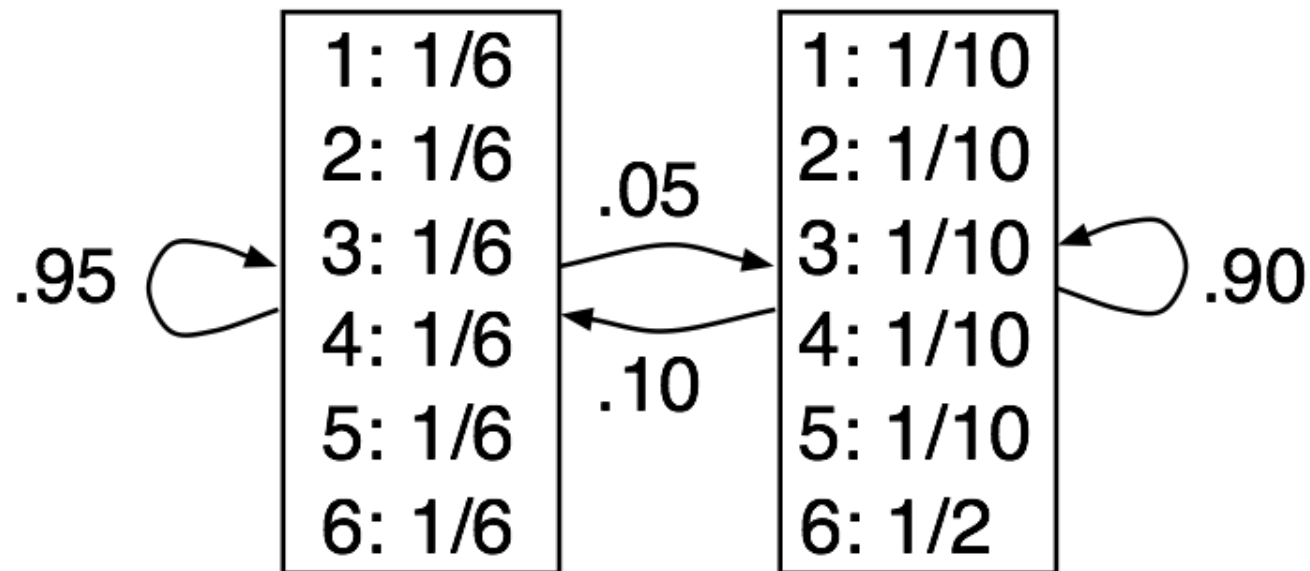
Alternative 1: what's the most likely state at step *i*?

$$\hat{\pi}_i = \arg\max_k P(\pi_i = k \mid x)$$

Note: the sequence of most likely states ≠ the most likely sequence of states. May not even be legal!

The Occasionally Dishonest Casino

1 fair die, 1 "loaded" die, occasionally swapped

```
Rolls     315116246446644245311321631164152133625145436316566256666
Die       FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLL

Rolls     651166453132651245636664631636663162326455236266666625151631
Die       LLLLLLFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLLFFFFFFFFF
Viterbi   LLLLLLFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFF

Rolls     222555441666566563564324364131513465146353411126414626253356
Die       FFFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls     366163666466232534413661661163252562462255265252266435353336
Die       LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi   LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls     233121625364414432335163243633665562466662632666612355245242
Die       FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
Viterbi   FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
```
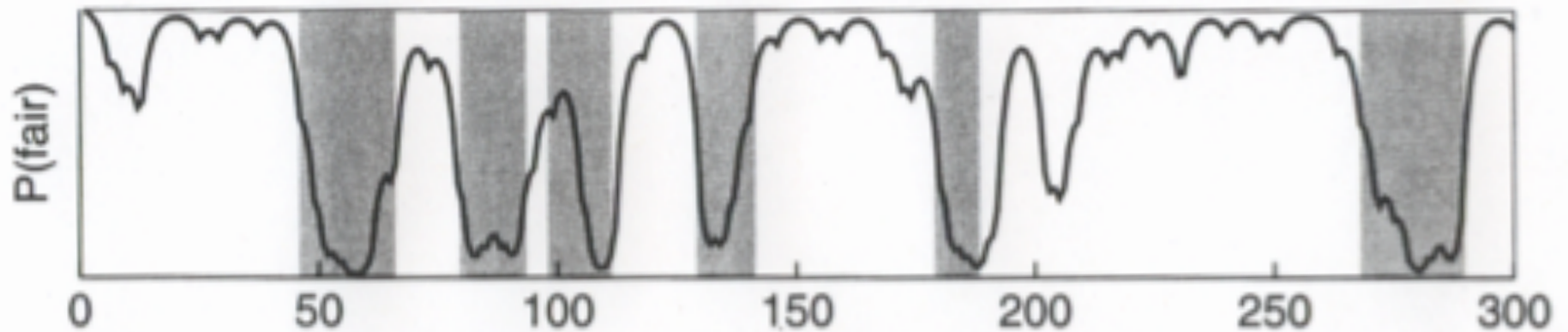
Figure 3.5

Rolls: Visible data–300 rolls of a die as described above.

Die: Hidden data–which die was actually used for that roll (F = fair, L = loaded).

Viterbi: the prediction by the Viterbi algorithm is shown.

Posterior Decoding

Figure 3.6 *The posterior probability of being in the state corresponding to the fair die in the casino example. The x axis shows the number of the roll. The shaded areas show when the roll was generated by the loaded die.*

From DEKM

Posterior Decoding, II

Alternative 1: what's most likely state at step *i* ?

$$\hat{\pi}_i = \arg\max_k P(\pi_i = k \mid x)$$

Alternative 2: given some function *g(k)* on states, what's its expectation. E.g., what's probability of "+" model in CpG HMM *(g(k)=1* iff *k* is "+" state)?

$$G(i \mid x) = \sum_k P(\pi_i = k \mid x) \cdot g(k)$$

CpG Islands again

Data: 41 human sequences, totaling 60kbp,
   including 48 CpG islands of about 1kbp each

Viterbi: Post-process:
   Found 46 of 48   46/48
   plus 121 "false positives"   67 false pos

Posterior Decoding:
   same 2 false negatives   46/48
   plus 236 false positives   83 false pos

   Post-process: merge within
      500; discard < 500

Training

Given model topology & training sequences,
  learn transition and emission probabilities

If π known, then MLE is just frequency observed
  in training data

$$a_{k,l} = \frac{\text{count of } k \to l \text{ transitions}}{\text{count of } k \to \text{anywhere transitions}}$$

$$e_k(b) = \ldots$$

$\leftarrow$ + pseudocounts?

If π hidden, then use EM:
  given π, estimate θ; given θ estimate π. $\Big\}$ 2 ways

Viterbi Training
given π, estimate θ; given θ estimate π

Make initial estimates of parameters θ

Find Viterbi path π for each training sequence

Count transitions/emissions on those paths,
  getting new θ

Repeat

Not rigorously optimizing desired likelihood, but
  still useful & commonly used.
  (Arguably good if you're doing Viterbi decoding.)

Baum-Welch Training

EM: given θ, estimate π ensemble; then re-estimate θ

$$P(\pi_i = k,\ \pi_{i+1} = l \mid x, \theta)$$

$$= \frac{f_k(i \mid \theta)\ a_{k,l}\ e_l(x_{i+1})\ b_l(i+1 \mid \theta)}{P(x \mid \theta)}$$

Estimated # of $k \to l$ transitions $\hat{A}_{k,l}$

$$= \sum_{\text{training seqs } x^j} \sum_i P(\pi_i = k,\ \pi_{i+1} = l \mid x^j, \theta)$$

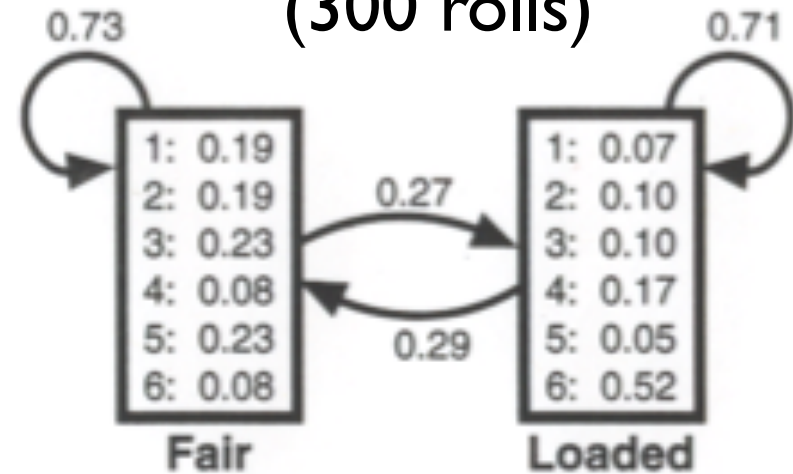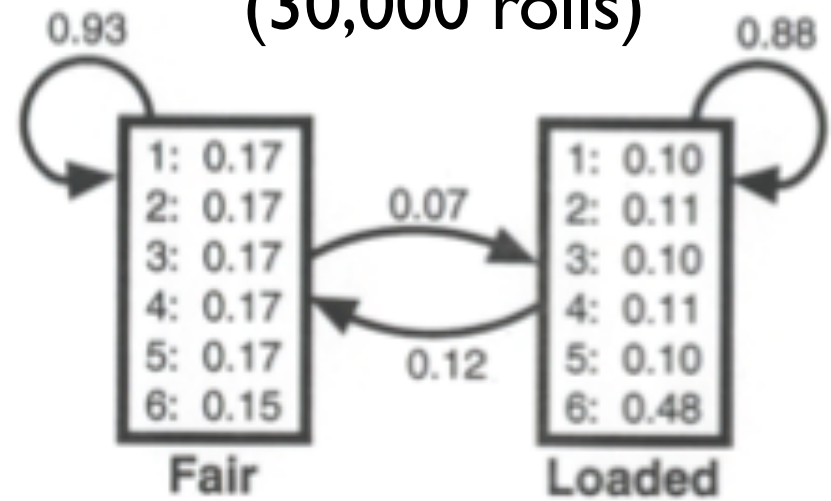New estimate $\hat{a}_{k,l} = \dfrac{\hat{A}_{k,l}}{\sum_l \hat{A}_{k,l}}$

Emissions: similar

True Model
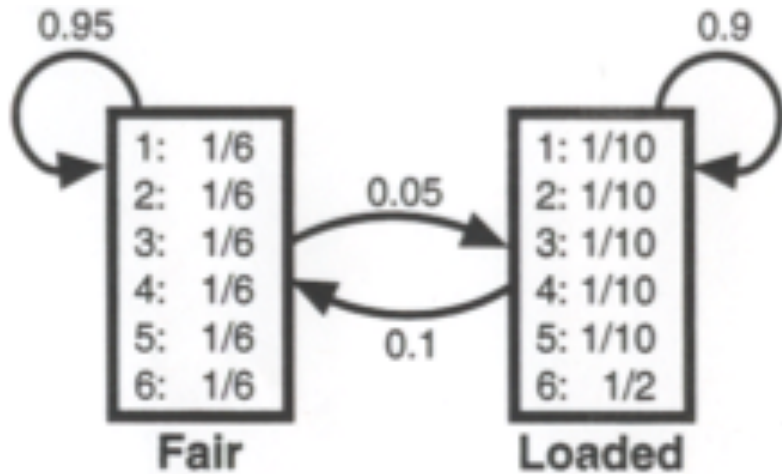
B-W Learned Model
(300 rolls)

B-W Learned Model
(30,000 rolls)

Log-odds (vs all F) per roll

True model   0.101 bits

300-roll est.   0.097 bits

30k-roll est.   0.100 bits

(NB: overestimated)

From DEKM

HMMs in Action: Pfam
http://pfam.sanger.ac.uk/

Proteins fall into families, both across & within species

  Ex: Globins, GPCRs, Zinc fingers, Leucine zippers,...

Identifying family very useful: suggests function, etc.

So, search & alignment are both important

One very successful approach: profile HMMs

```
Helix                    AAAAAAAAAAAAAAAA   BBBBBBBBBBBBBBBBCCCCCCCCCCCC
HBA_HUMAN    ---------VLSPADKTNVKAAWGKVGA--HAGEYGAEALERMFLSFPTTKTYFPHF
HBB_HUMAN    --------VHLTPEEKSAVTALWGKV----NVDEVGGEALGRLLVVYPWTQRFFESF
MYG_PHYCA    ---------VLSEGEWQLVLHVWAKVEA--DVAGHGQDILIRLFKSHPETLEKFDRF
GLB3_CHITP   ----------LSADQISTVQASFDKVKG------DPVGILYAVFKADPSIMAKFTQF
GLB5_PETMA   PIVDTGSVAPLSAAEKTKIRSAWAPVYS--TYETSGVDILVKFFTSTPAAQEFFPKF
LGB2_LUPLU   --------GALTESQAALVKSSWEEFNA--NIPKHTHRFFILVLEIAPAAKDLFS-F
GLB1_GLYDI   --------GLSAAQRQVIAATWKDIAGADNGAGVGKDCLIKFLSAHPQMAAVFG-F
Consensus          Ls....  v aW kv .  .      g . L.. f . P .    F  F

Helix            DDDDDDDEEEEEEEEEEEEEEEEEEEEEE              FFFFFFFFFFFFF
HBA_HUMAN    -DLS-----HGSAQVKGHGKKVADALTNAVAHV---D--DMPNALSALSDLHAHKL-
HBB_HUMAN    .GDLSTPDAVMGNPKVKAHGKKVLGAFSDGLAHL---D--NLKGTFATLSELHCDKL-
MYG_PHYCA    KHLKTEAEMKASEDLKKHGVTVLTALGAILKK----K-GHHEAELKPLAQSHATKH-
GLB3_CHITP   AG-KDLESIKGTAPFETHANRIVGFFSKIIGEL--P---NIEADVNTFVASHKPRG-
GLB5_PETMA   ǨGLTTADQLKKSADVRWHAERIINAVNDAVASM--DDTEKMSMKLRDLSGKHAKSF-
LGB2_LUPLU   LK-GTSEVPQNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG-
GLB1_GLYDI   SG----AS---DPGVAALGAKVLAQIGVAVSHL--GDEGKMVAQMKAVGVRHKGYGN
Consensus     .  t      .. . v..Hg kv. a    a...l   d   . a l. l   H  .

Helix            FFGGGGGGGGGGGGGGGGGGGG   HHHHHHHHHHHHHHHHHHHHHHHHH
HBA_HUMAN    -RVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR------
HBB_HUMAN    -HVDPENFRLLGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH------
MYG_PHYCA    -KIPIKYLEFISEAIIHVLHSRHPGDFGADAQGAMNKALELFRKDIAAKYKELGYQG
GLB3_CHITP   --VTHDQLNNFRAGFVSYMKAHT--DFA-GAEAAWGATLDTFFGMIFSKM-------
GLB5_PETMA   -QVDPQYFKVLAAVIADTVAAG---------DAGFEKLMSMICILLRSAY-------
LGB2_LUPLU   --VADAHFPVVKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMNDAA---
GLB1_GLYDI   KHIKAQYFEPLGASLLSAMEHRIGGKMNAAAKDAWAAAYADISGALISGLQS-----
Consensus       v.   f  l . ..  ....     f   . aa. k. .      l sky
```
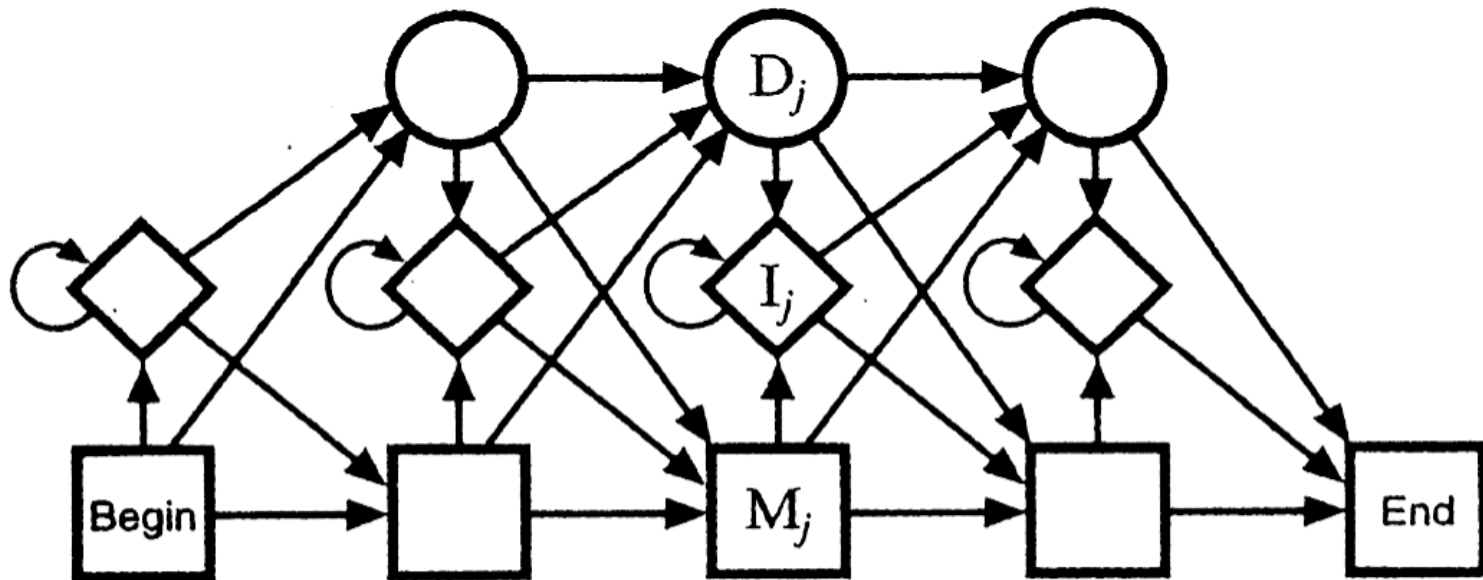
Alignment of 7 globins. A-H mark 8 alpha helices.
Consensus line: upper case = 6/7, lower = 4/7, dot=3/7.
Could we have a profile (aka weight matrix) w/ indels?

Profile Hmm Structure

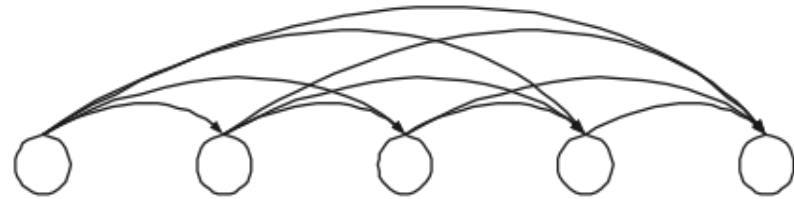**Figure 5.2** *The transition structure of a profile HMM.*

M_j: Match states (20 emission probabilities)

I_j: Insert states (Background emission probabilities)

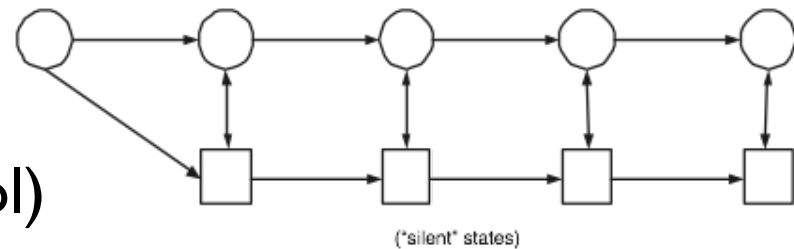D_j: Delete states (silent - no emission)

Silent States

Example: chain of
   states, can skip
   some

Problem: many parameters.

A solution: chain
   of "silent" states;
   fewer parameters
   (but less detailed control)

Algorithms: basically the same.

("silent" states)

Using Profile HMM's

Search

    Forward or Viterbi

    Scoring

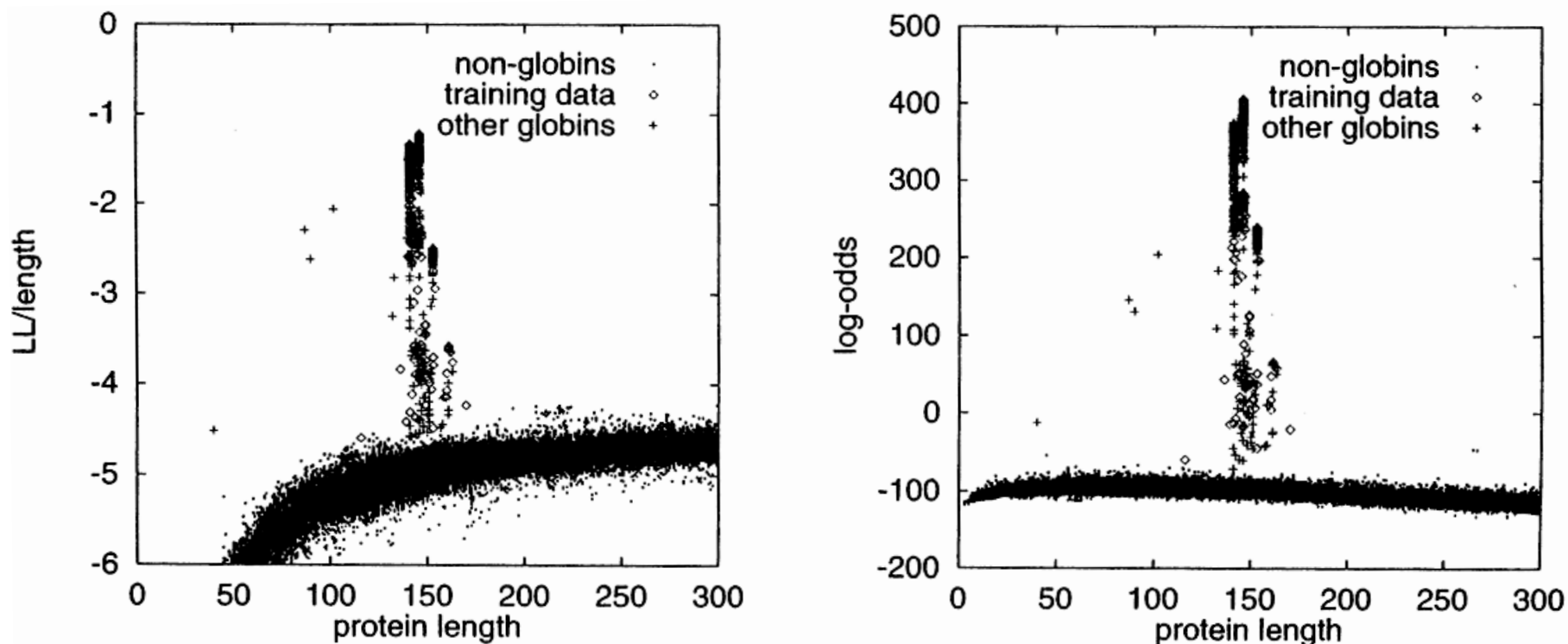   Log likelihood (length adjusted)

        Log odds vs background   } next slides

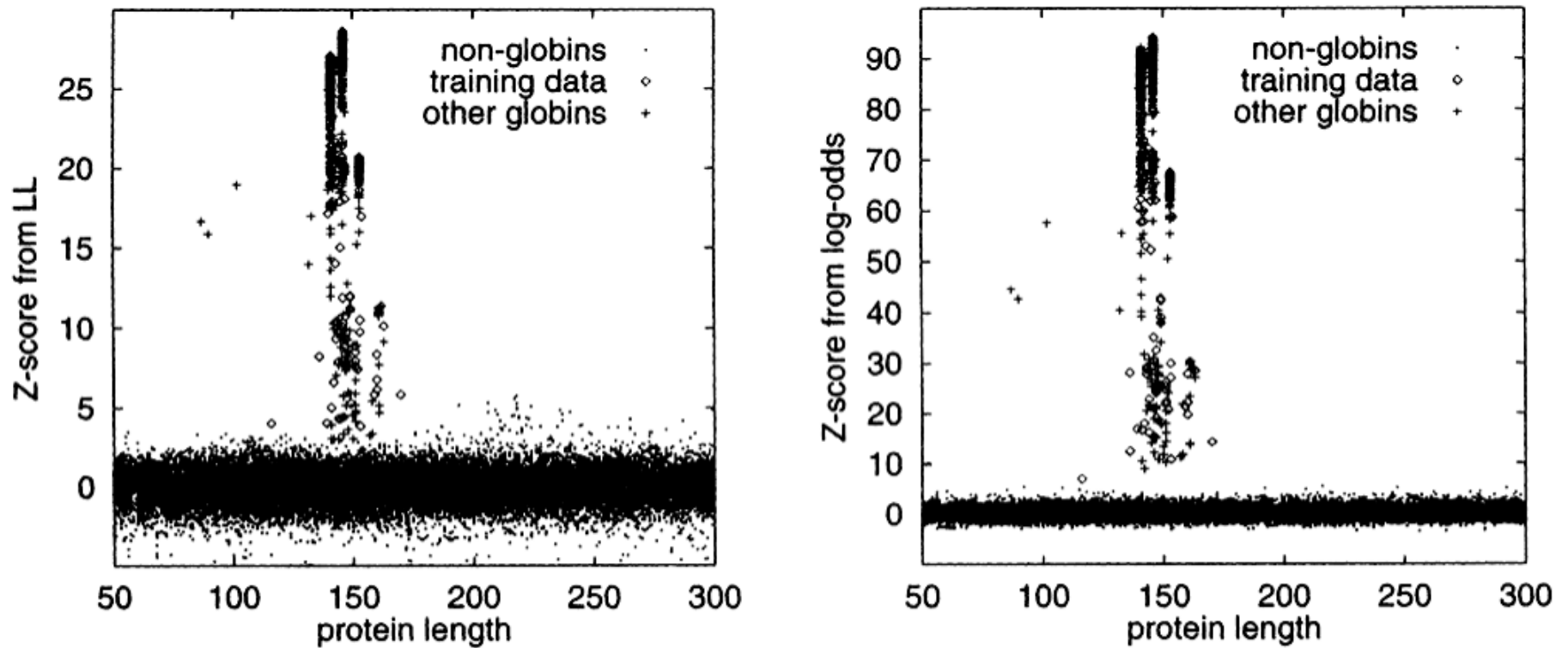   Z scores from either

Alignment

    Viterbi

Likelihood vs Odds Scores

Figure 5.5 *To the left the length-normalized LL score is shown as a function of sequence length. The right plot shows the same for the log-odds score.*

Z-Scores

Figure 5.6 *The Z-score calculated from the LL scores (left) and the log-odds (right).*

From DEKM

Pfam Model Building

Hand-curated "seed" multiple alignments

Train profile HMM from seed alignment

Hand-chosen score threshold(s)

Automatic classification/alignment of all other protein sequences

11912 families in Rfam 24.0, 10/200 (covers ~75% of proteins)

Model-building refinements

Pseudocounts (count = 0 common when training with 20 aa's)

$$e_i(a) = \frac{C_{i,a} + A \cdot q_a}{\sum_a C_{i,a} + A}, \quad A \sim 20, \ q_a = \text{ background}$$

(~50 training sequences)

Pseudocount "mixtures", e.g. separate pseudocount vectors for various contexts (hydrophobic regions, buried regions,...)

(~10-20 training sequences)

More refinements

Weighting: may need to down weight highly similar sequences to reflect phylogenetic or sampling biases, etc.

Match/insert assignment: Simple threshold, e.g. "> 50% gap $\Rightarrow$ insert", may be suboptimal. Can use forward-algorithm-like dynamic programming to compute max *a posteriori* assignment.

Numerical Issues

Products of many probabilities → 0

For Viterbi: just add logs

For forward/backward: also work with logs, but
  you need sums of products, so need
  "log-of-sum-of-product-of-exp-of-logs",
  e.g., by table/interpolation

Keep high precision and perhaps scale factor
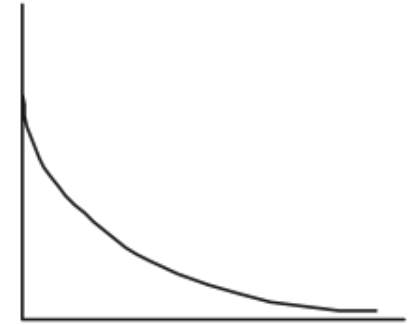
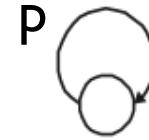Working with log-odds also helps.
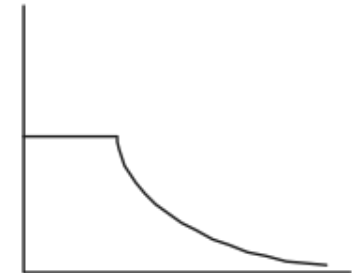
Model structure

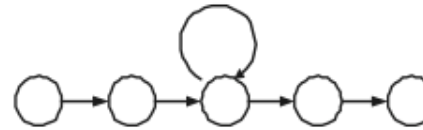Define it as well as you can.

In principle, you can allow all transitions and hope to learn their probabilities from data, but it usually works poorly – too many local optima
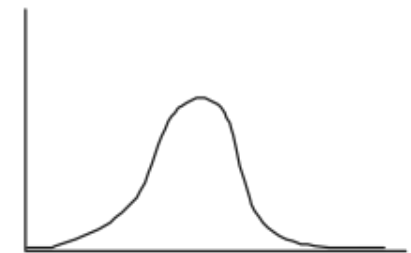
Duration Modeling

Self-loop duration:
  geometric $p^n(1-p)$

min, then geometric

"negative binomial"

More general: possible (but slower)

HMM Summary

Inference
    Viterbi – best single path   (max of products)
    Forward – sum over all paths   (sum of products)
    Backward – similar
    Posterior decoding

joint vs conditional probs

Model building
    Semi-supervised – typically fix architecture (e.g. profile HMM), then learn parameters
    Baum-Welch – training via EM and forward/backward (aka the forward/backward algorithm)
    Viterbi training – also "EM", but Viterbi-based

HMM Summary (cont.)

Search:

Viterbi or forward

Scoring:

Odds ratio to background

Z-score

E-values, etc., too

Excellent tools available (SAM, HMMer, Pfam, …)

A very widely used tool for biosequence analysis