Hidden Markov Models and Gene Prediction

3 February 2021

Overview

- Sequence profiles
- How hidden Markov models work
- Training HMMs
- HMMs for gene prediction

```
K-ELQRAASLTIEV
KDEGQK--SLVIDV
```

If we have an alignment…

…what can we do with it?

For many questions, we would like to know the distribution of residues (and gaps) in a block of sequences

```
CGGCCT
CGAGCT
GATGCA
AAAGCA
ATAGCA
TCTACT
AACATC
TACGCC
AACGAG
AGCTGT
```

Position-specific scoring matrices (PSSM)

PAM, BLOSUM, etc. are position-**independent** scoring matrices

A PSSM is a log-odds matrix of column frequencies

```
CGGCCT
CGAGCT
GATGCA
AAAGCA
ATAGCA
TCTACT
AACATC
TACGCC
AACGAG
AGCTGT
```

➡

Frequency Matrix

	1	2	3	4	5	6
A	0.5	0.5	0.3	0.2	0.1	0.3
C	0.2	0.1	0.4	0.1	0.7	0.2
G	0.1	0.3	0.1	0.5	0.1	0.1
T	0.2	0.1	0.2	0.2	0.1	0.4

Background frequencies:
A = 19/60 = 0.317
C = 17/60 = 0.283
G = 12/60 = 0.2
T = 12/60 = 0.2

Frequency Matrix

	1	2	3	4	5	6
A	0.5	0.5	0.3	0.2	0.1	0.3
C	0.2	0.1	0.4	0.1	0.7	0.2
G	0.1	0.3	0.1	0.5	0.1	0.1
T	0.2	0.1	0.2	0.2	0.1	0.4

Background frequencies:
A = 19/60 = 0.317
C = 17/60 = 0.283
G = 12/60 = 0.2
T = 12/60 = 0.2

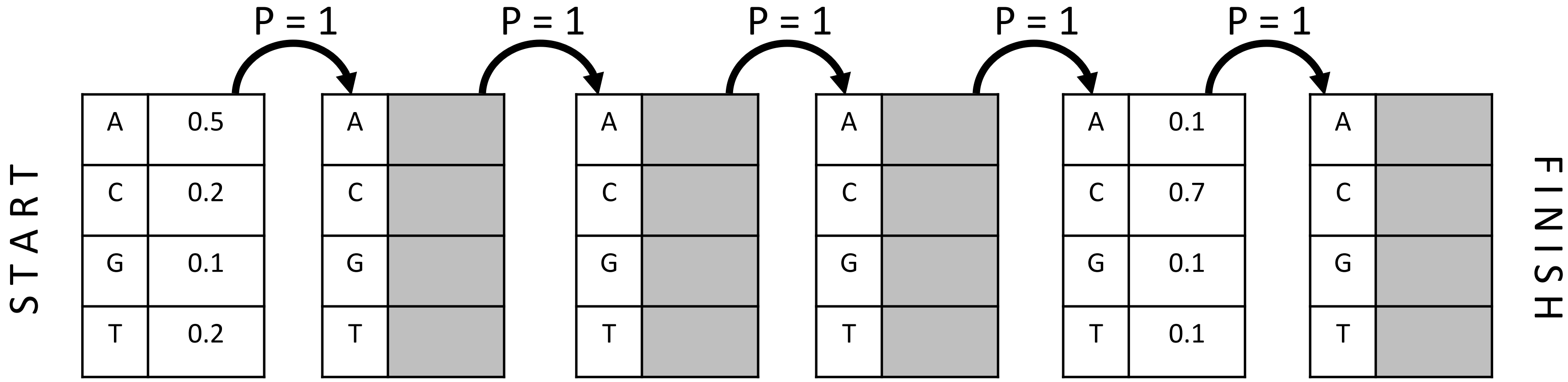
log$_n$-odds matrix (n = *e*)

	1	…	5
A	0.18		-0.5
C	-0.15		0.54
G	-0.3		-0.3
T	0		-0.3

Aligning a sequence against log-odds matrix:
Add scores for residue at each position, then take n^{sum}

How do we represent insertions and deletions?

Transitions in a Probability Matrix



START

P = 1

A	0.5
C	0.2
G	0.1
T	0.2

P = 1

A	
C	
G	
T	

P = 1

A	
C	
G	
T	

P = 1

A	
C	
G	
T	

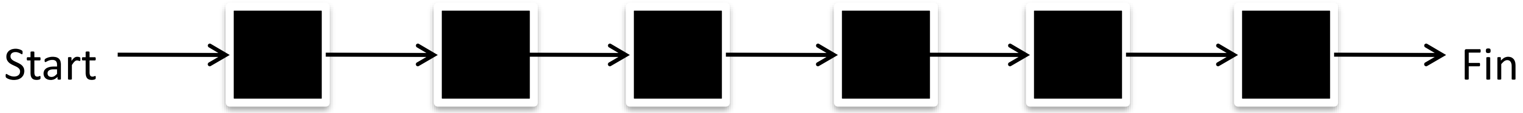
P = 1

A	0.1
C	0.7
G	0.1
T	0.1

A	
C	
G	
T	

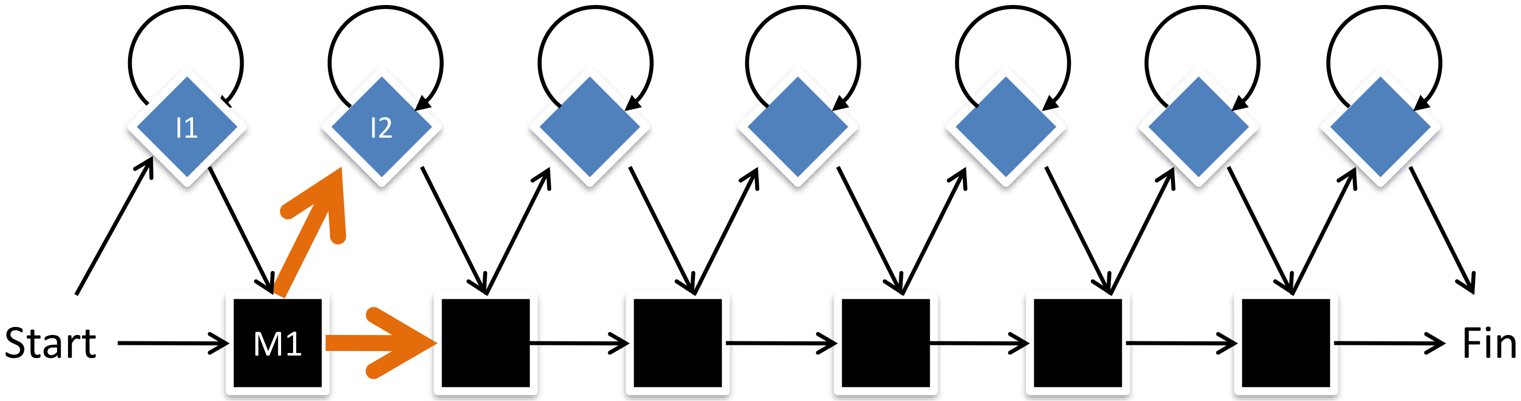
FINISH

Transition from match state *k* to match state *k* + 1 with probability 1.0



Match states

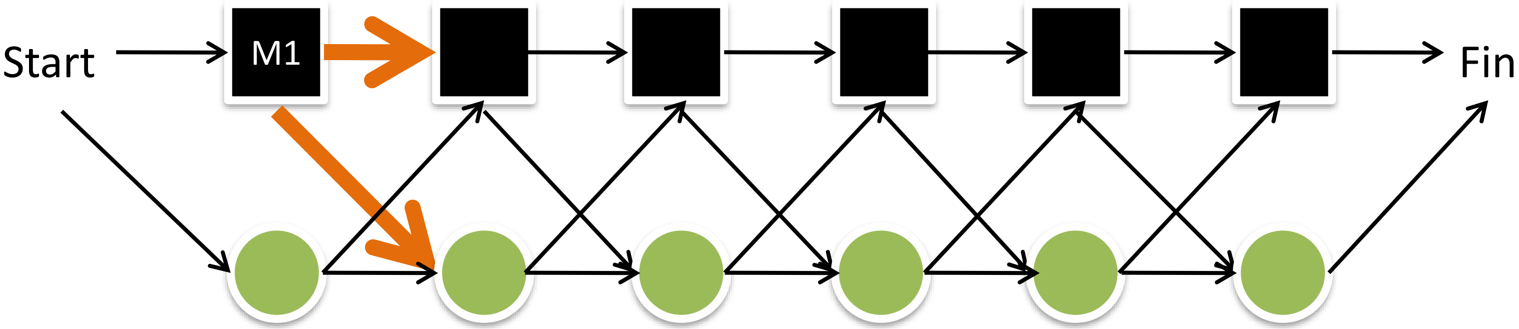
Insertions



Insert states

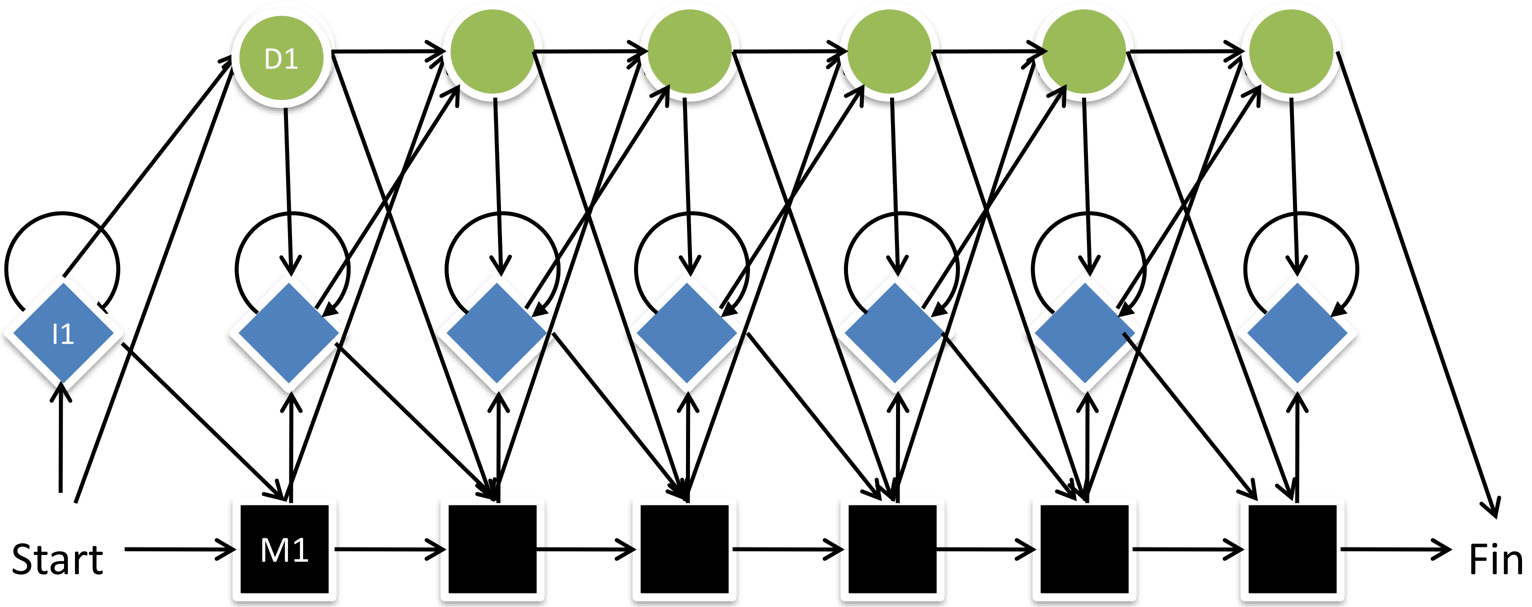
Transition probabilities out of any state must sum to 1.0

Deletions



Delete states

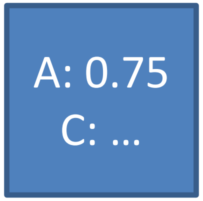
Hidden Markov Model



HIDDEN because we don’t actually know the states of the sequence we’re looking at
MARKOV because the future does not depend on the past
MODEL because, well, it’s a model

Key components of an HMM

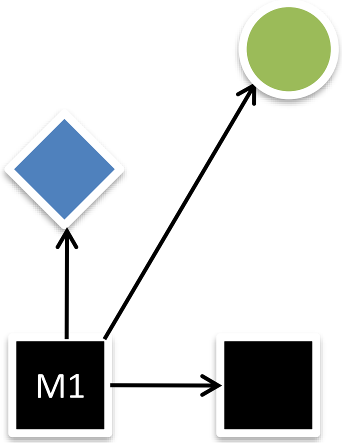
- EMISSIONS: A character (nucleotide or amino acid) produced by a given insertion or match state

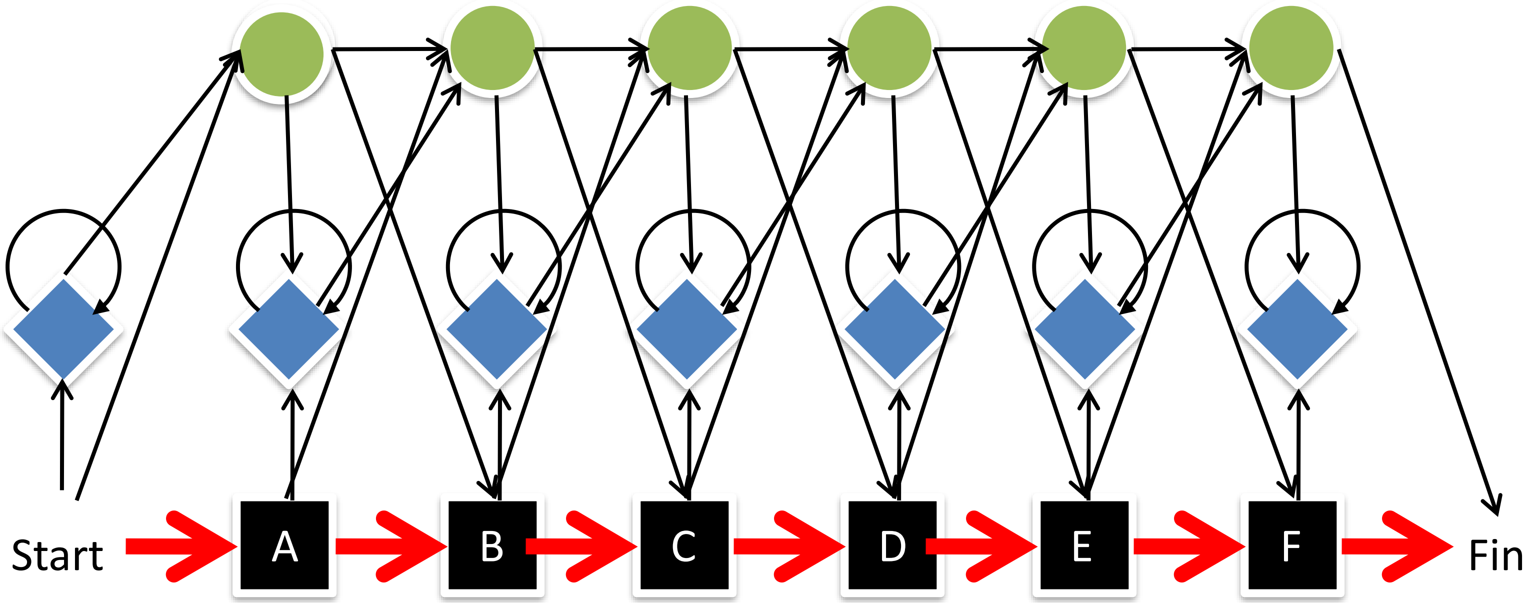


A: 0.75
C: …

emission probability

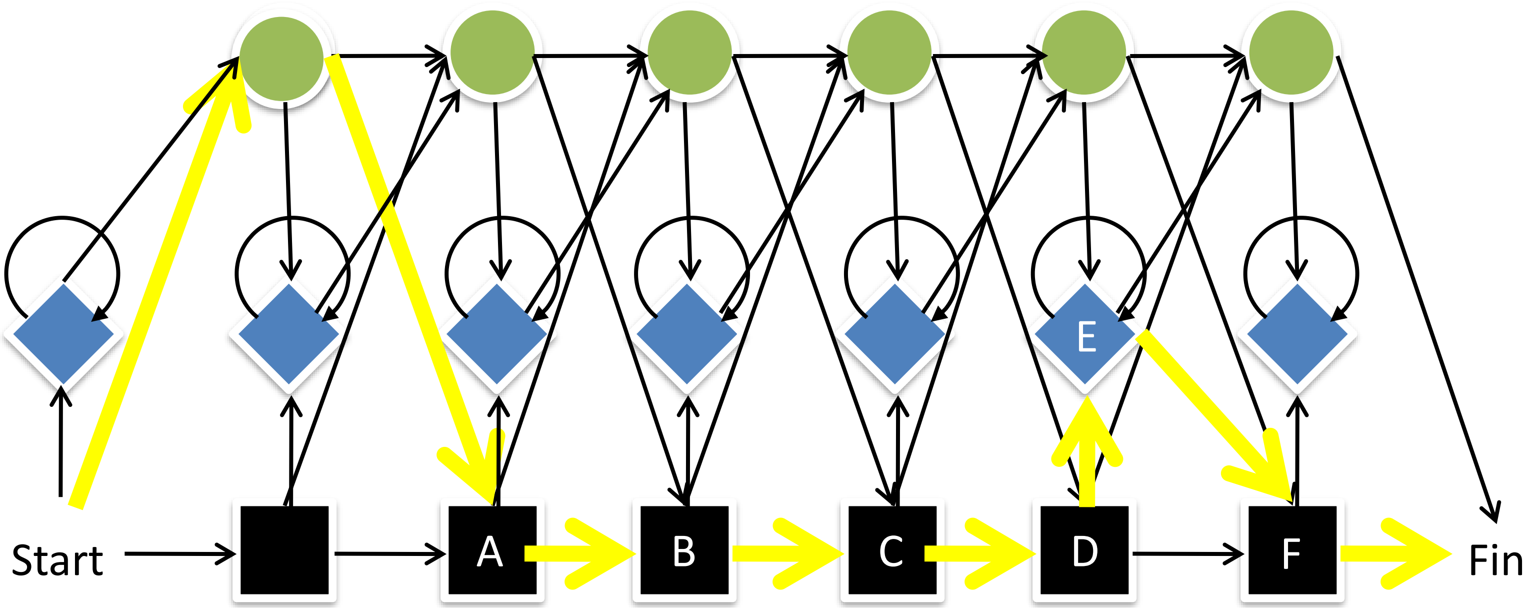
- TRANSITIONS: The probability of going from state *i* to state *j* (sum of all transitions from a given state = 1)





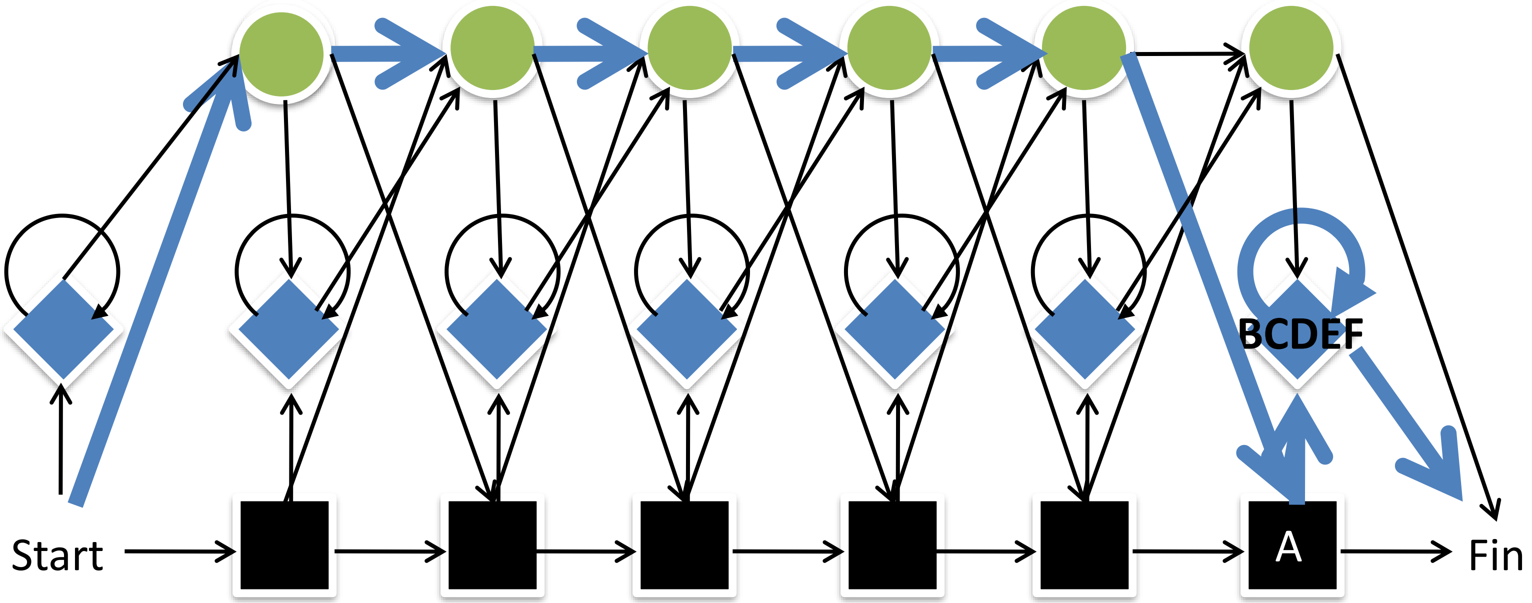
Let’s run a sequence through the HMM!

ABCDEF



Let’s run a sequence through the HMM!

`ABCDEF`



Let’s run a sequence through the HMM!

ABCDEF

The product of the EMISSION PROBABILITIES *e* and the TRANSITION PROBABILITIES *a* through the model

=

The **joint probability** of the *sequence x* and the *path* π

The product of the EMISSION PROBABILITIES *e* and the TRA S *a* through the model

Or sum of logs

=

The **joint probability** of the *sequence x* and the *path π*

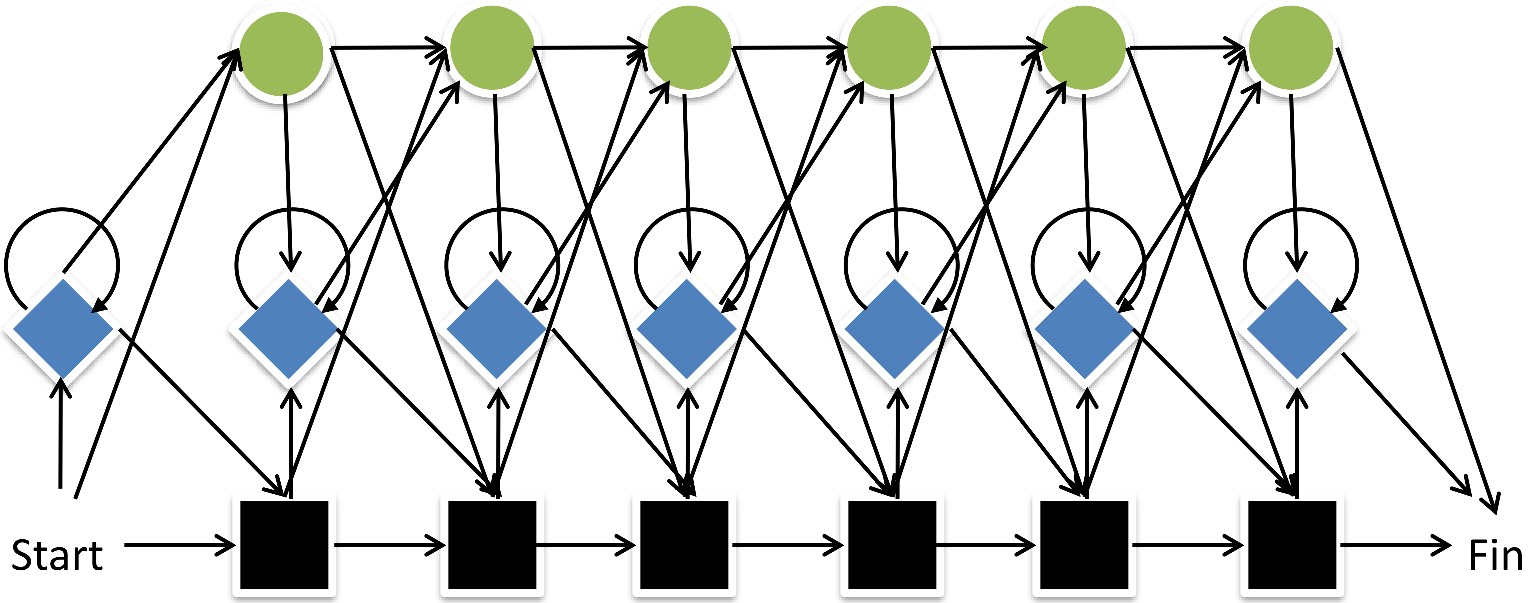
Best path

- There are many paths π through the model for any given sequence *x*

- What is the best path π^* ?

The Viterbi Algorithm

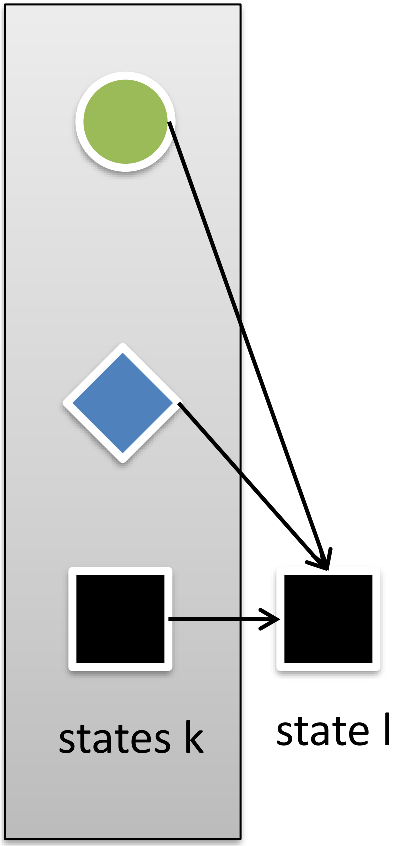
- As with multiple sequence alignment, we cannot be greedy in our choice of path
- But we only need to consider the **best path** to every possible state in the model
- Dynamic programming!



v(Start) = 1

$$v_l(i) = e_i(x_i)\max_k(v_k(i-1)a_{kl})$$

Huh?



i = { A,B,C,D,E,F }

$$\underbrace{v_l(i)}_{\text{Viterbi score of sequence position } l \text{ at state } l} = \underbrace{e_i(x_i)}_{\text{Emission probability of } x_i} \underbrace{\max_k}_{\text{max over all possibilities}} \underbrace{(v_k(i-1)a_{kl})}_{\text{Viterbi score at previous state, times the transision probabillity}}$$

So we are saving the best path for each **character** { A,B,C,D,E,F } at each **state** in the HMM

When we choose our best incoming path, we save a pointer as before and **backtrace**

Complexity = O(LS) (# of characters x # of states in the HMM structure) – kinda like n^2

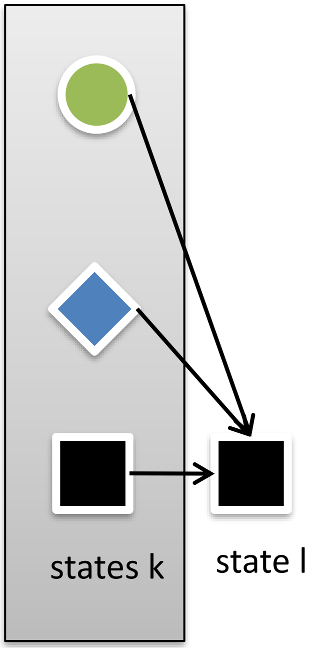
The Viterbi alignment of each member of a set of sequences *X* to a trained HMM yields a ***multiple alignment*** of these sequences

All Paths

FORWARD algorithm

sums over incoming paths

instead of taking max



i = { A,B,C,D,E,F }

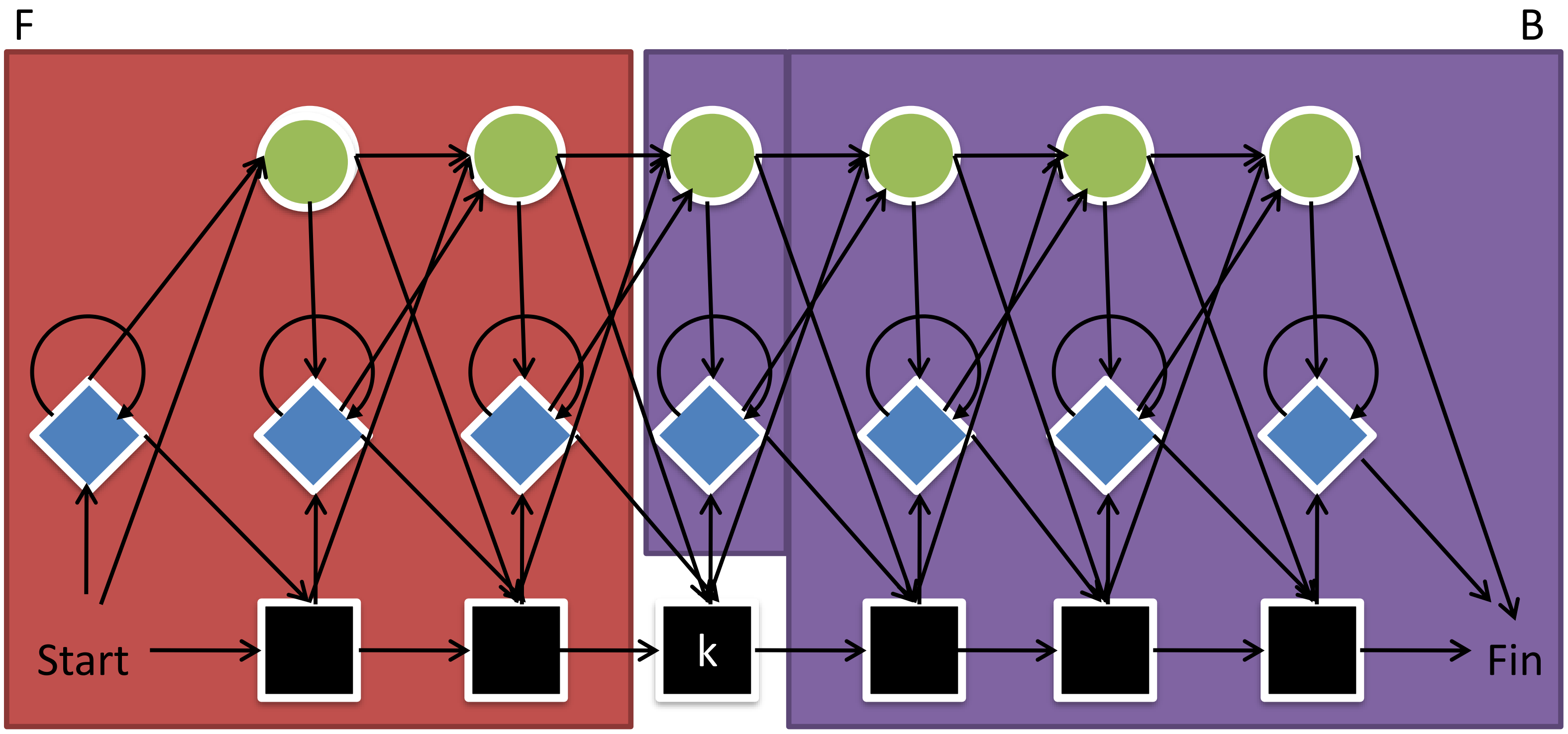
$$f_l(i) = e_i(x_i)\sum_k (f_k(i-1)a_{kl})$$

The Backward Algorithm

- Kind of like the forward algorithm, but starts from the finish and works backward

$$b_k(i) = \sum_l a_{kl} e_l(x_{i+1}) b_l(i+1)$$

- Why would we want to do this?

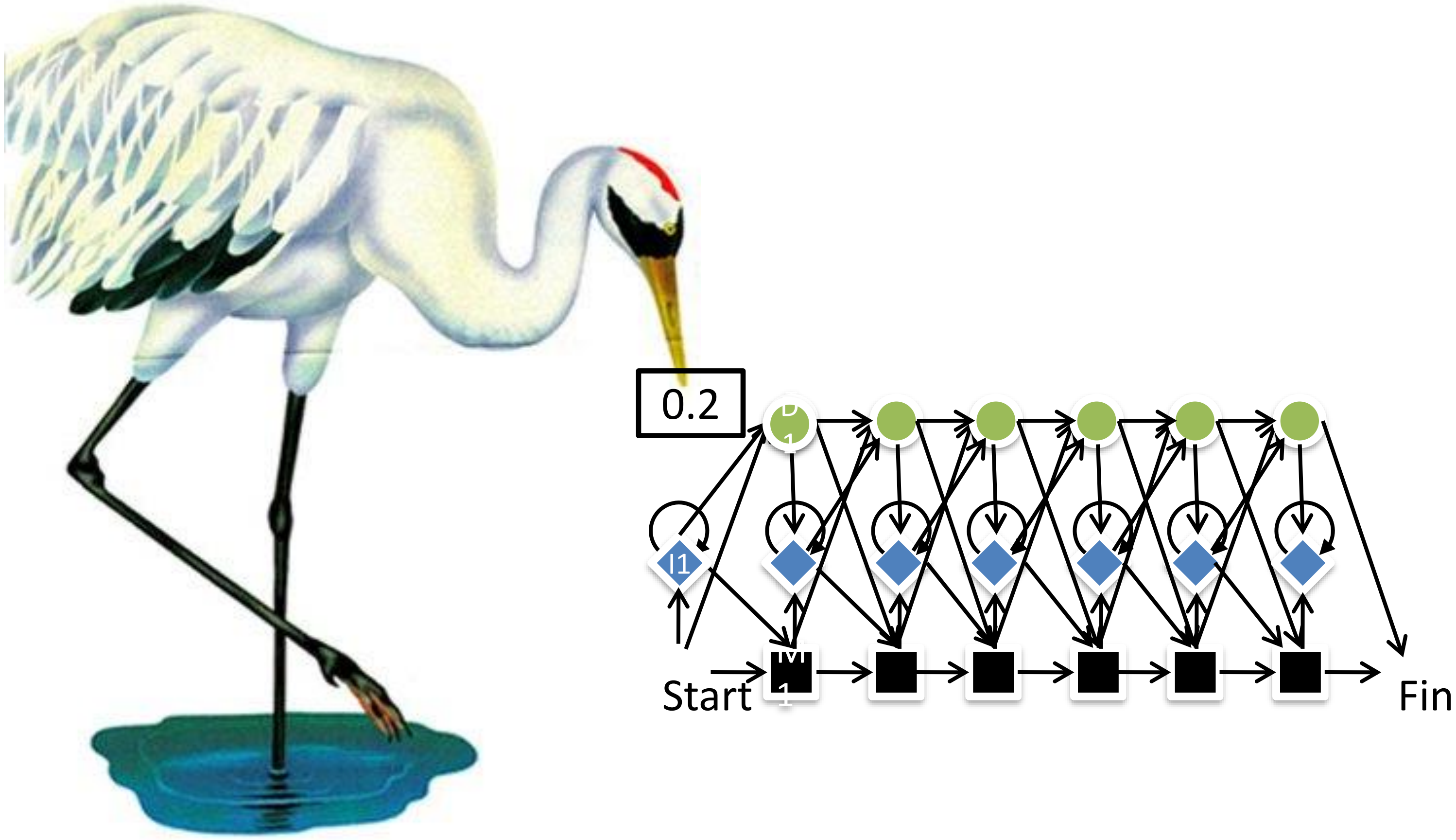


By running the forward and backward algorithms together for a given sequence, we can compute the probability that character *i* in sequence *x* maps to state *k*

ABCDEF

P(x, *k* = D)?

Training HMMs



Two components of training

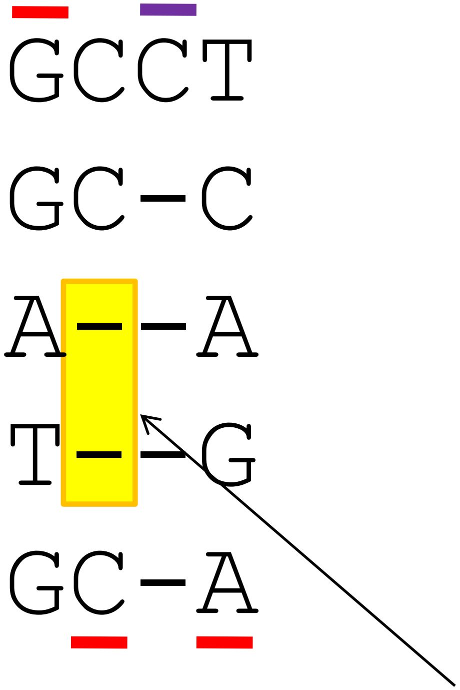
- Build the HMM structure or ‘skeleton’
  - Custom-tailored with exquisite knowledge of the problem to be modelled
  - In ignorance, build a complete model

- Assign transition and emission probabilities to the thing

Training (supervised)

- Construct a multiple sequence alignment using some method, and build the HMM using empirical frequencies

- Supervised because we’re specifying exactly WHAT sequences belong in the model



```
GCCT
GC-C
A--A
T--G
GC-A
```

Match states

Insertion state

Deletions

Note that we now get custom gap costs!

Training (Unsupervised)

- What if we don’t already have an alignment of the sequences?

- In this case, we can use an **iterative** approach to maximize the probability of the model

Unsupervised training: Baum-Welch Algorithm

- Random start for all emission probabilities (e_k) and transition probabilities (a_{kl})

- Run the forward and backward algorithms on all training sequences to count empirical probabilities E_k and A_{kl}

- Use these probability distributions to generate new e_k and a_{kl}

Big Alphabets, Few Sequences

Homologous residues
from a family of sequences

I
N
D
Q
R
S
D
Q
R
N
M
I
I
D
D
S

Incomplete sampling
in our database

Sampled set

I
N
D
Q
Q
D

Build matrix

A	0
C	0
D	2
E	0
F	0
G	0
H	0
I	1
K	0
L	0
M	0
N	1
P	0
Q	2
R	0
S	0
T	0
V	0
W	0
Y	0

What happens when the probability of
character *i* at position *k* is = 0?

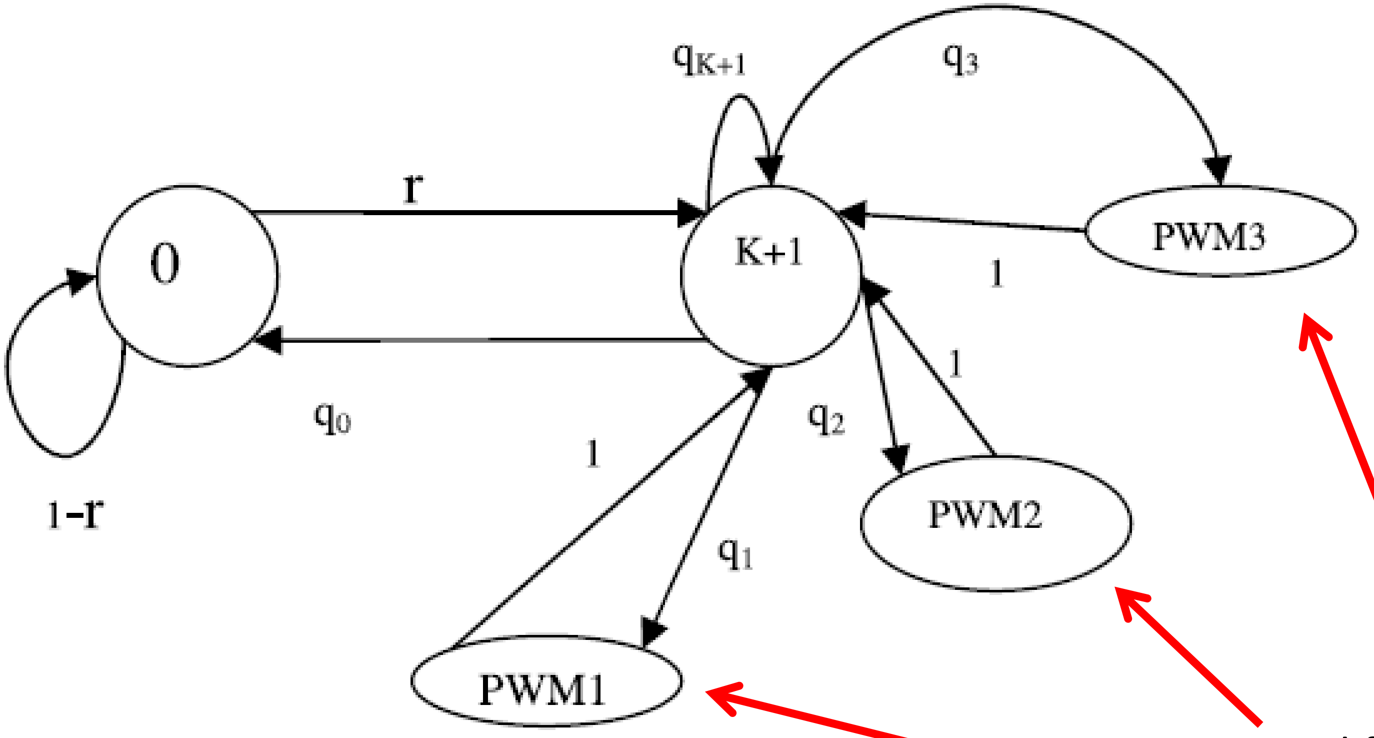
Psolution

- Add **pseudocounts** to each column of the multiple sequence alignment

- Laplace’s Rule: Add 1 to every count (!)
- Add small counts in proportion to background frequencies
- Modify added counts using PAM matrix or other distributions (Dirichlet mixtures)

Beyond sequences:
Other applications of HMMs

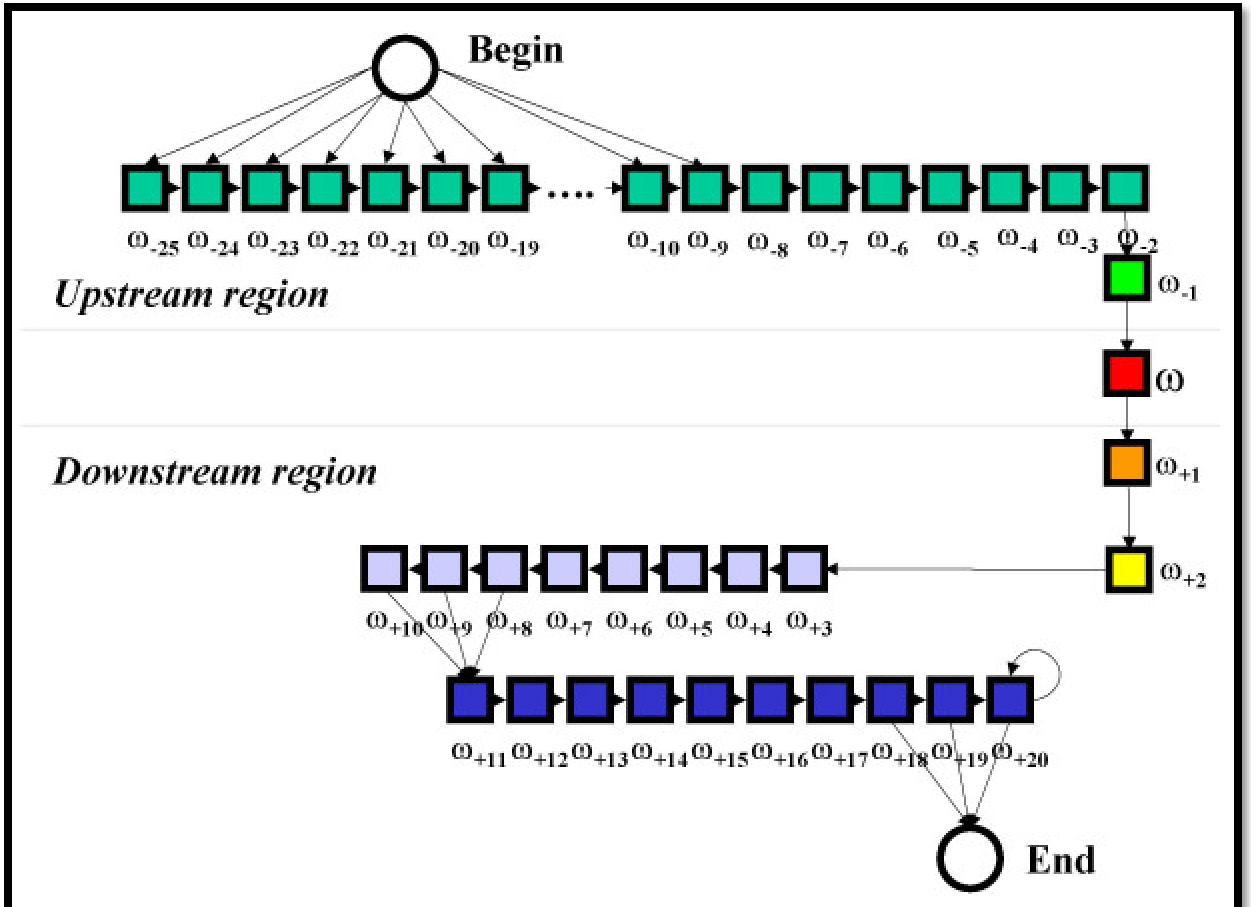
Regulatory Element Detection



3 different Res
(like promoters)

Wu and Xie, *J Comput Biol* (2007)

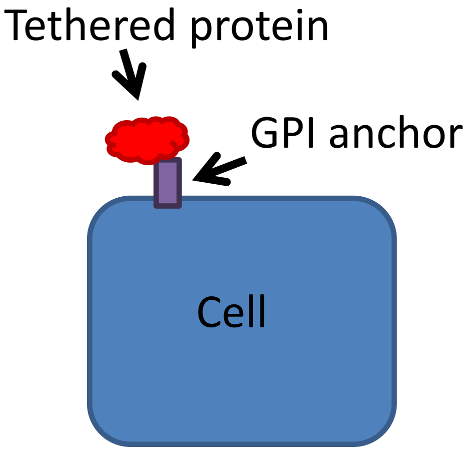
Glycosylphosphatidylinositol anchors



Begin
ω-25 ω-24 ω-23 ω-22 ω-21 ω-20 ω-19
ω-10 ω-9 ω-8 ω-7 ω-6 ω-5 ω-4 ω-3 ω-2
ω-1
Upstream region
ω
Downstream region
ω+1
ω+2
ω+10 ω+9 ω+8 ω+7 ω+6 ω+5 ω+4 ω+3
ω+11 ω+12 ω+13 ω+14 ω+15 ω+16 ω+17 ω+18 ω+19 ω+20
End

The HMM model of the ω-site. Different colors represent different emission probability sets. ω-site is represented in red. Surrounding residues are colored in green, orange and yellow. The preceding region is represented in dark green. The spacer and the C terminal hydrophobic regions are depicted in violet and blue, respectively. The total number of independent trainable parameters is 147.

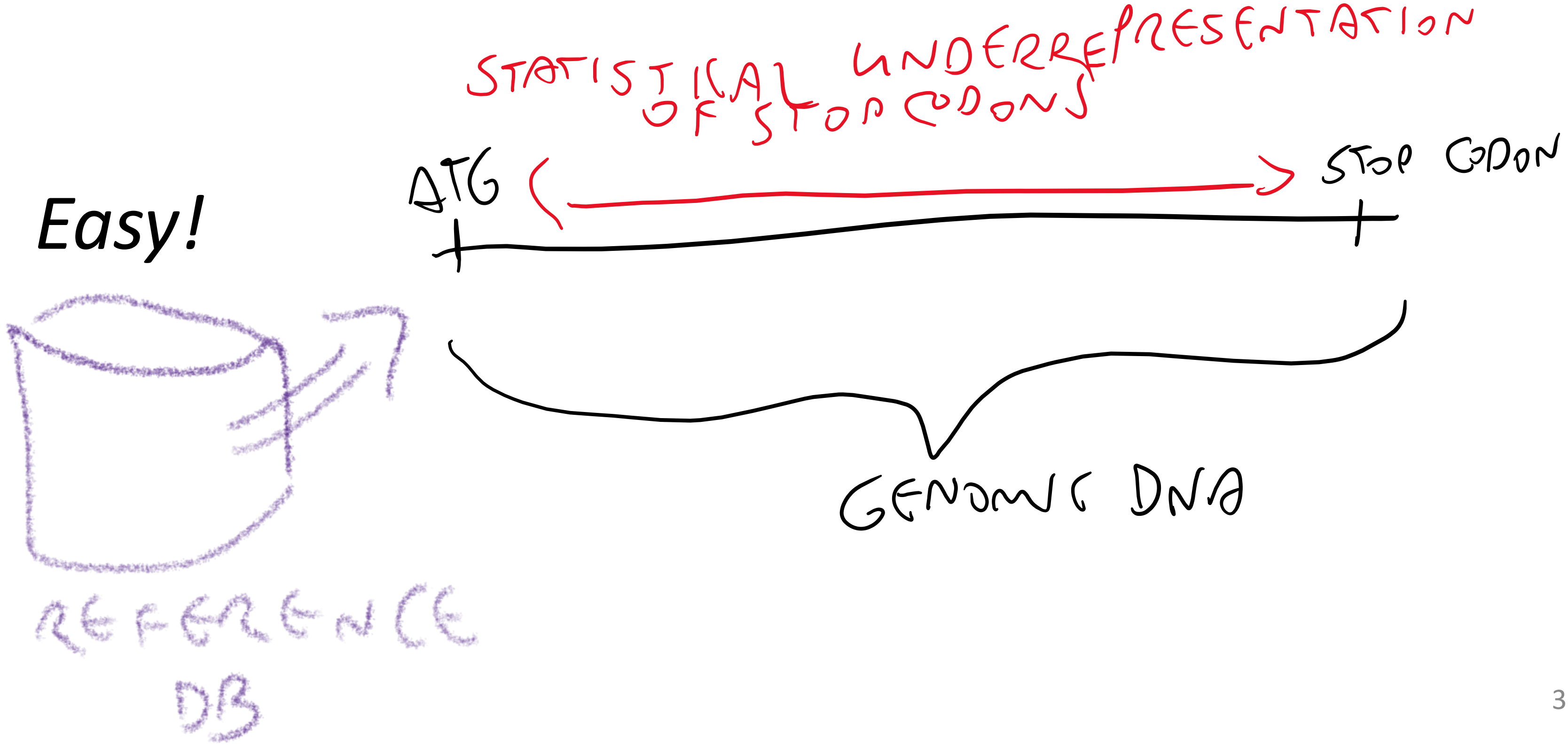
Pierleoni *et al. BMC Bioinformatics* 2008 **9**:392 doi:10.1186/1471-2105-9-392



Gene prediction

Given a genome sequence (complete or draft), identify all of the genes

Easy!

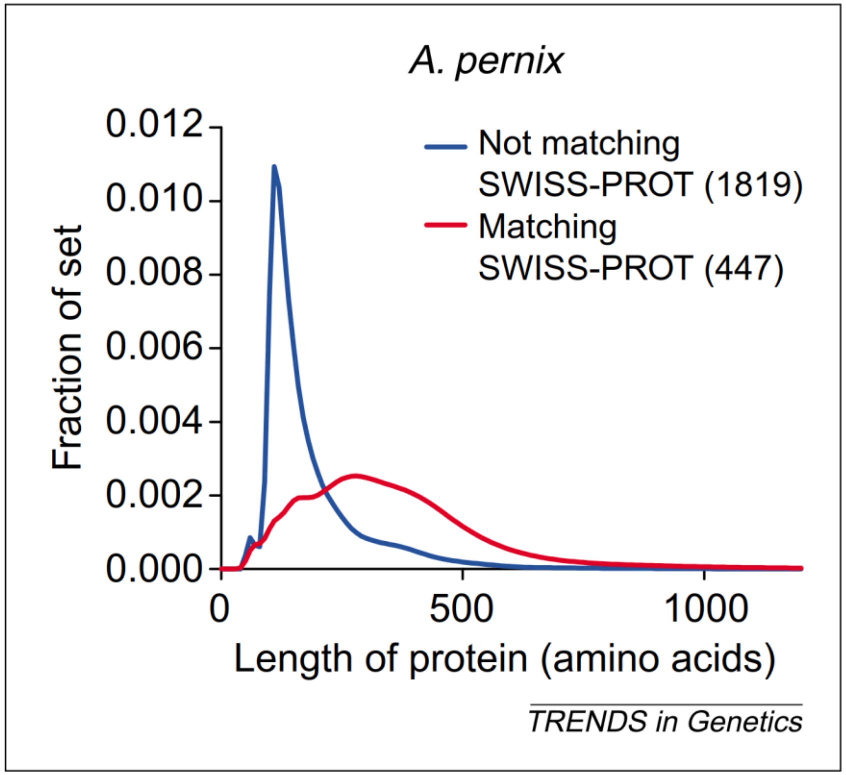


Maybe not so easy

Because:

- Alternative start codons (TTG, GTG)
- Uncertain start codons (which ATG?)
- Introns
- Short genes
- Non-protein-coding genes
- Genes that overlap
- Genes with no known homologs

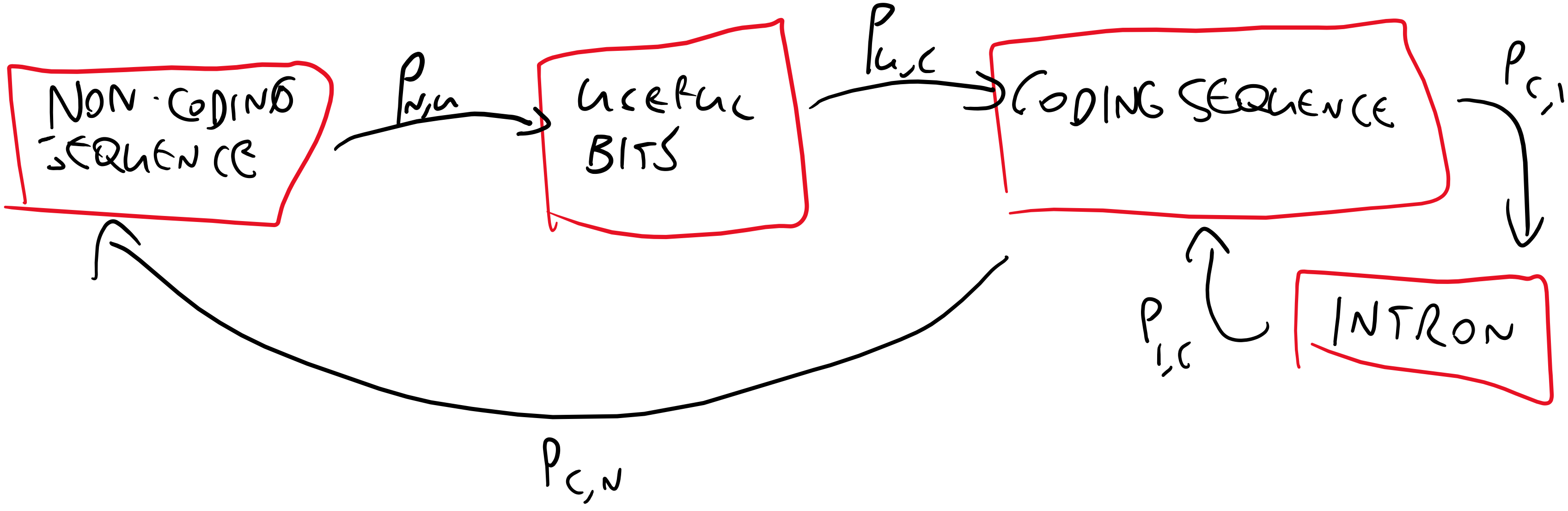
What not to do



Aeropyrum pernix genome:
If distance between start and stop codon is > 100 nt, call it a gene!

Skovgaard et al (2001) *Trends in Genetics*

Hidden Markov Models – the basic idea

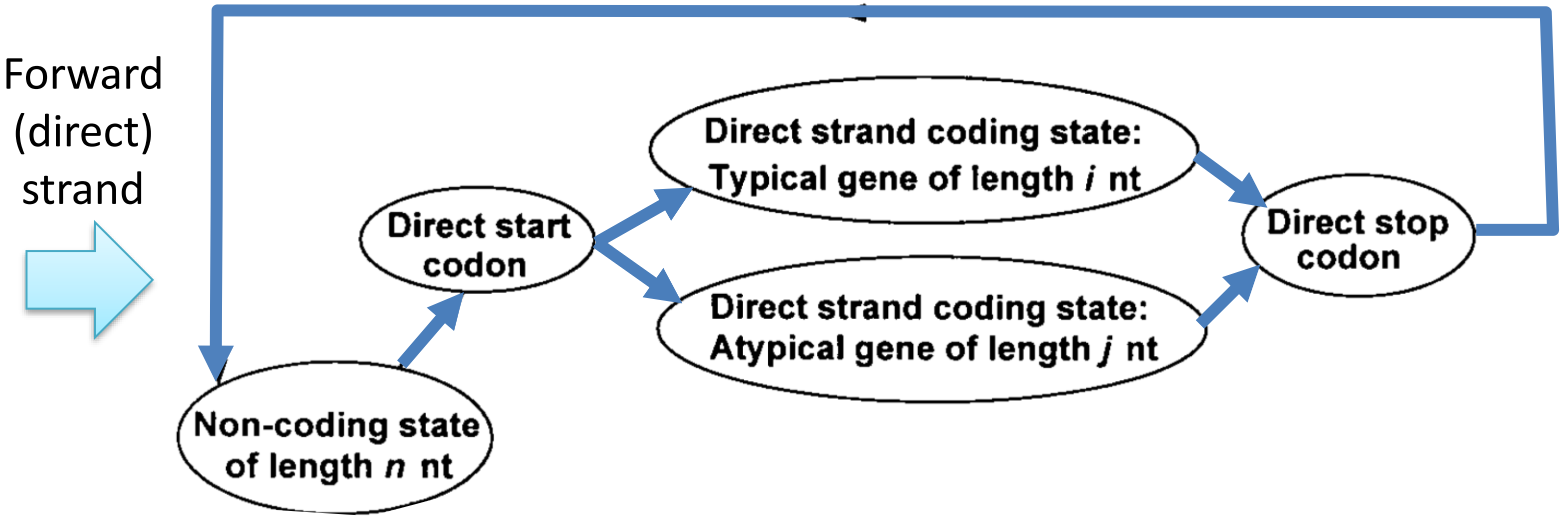


Useful pieces of the puzzle

- Non-coding and coding sequence have very different patterns (G+C content, periodicity, etc)
- There are useful translation start signals beyond just the start codon

GeneMark.hmm

Lukashin and Borodovsky (1998) *Nucleic Acids Res*



Forward
(direct)
strand
Direct strand coding state:
Typical gene of length *i* nt
Direct start codon
Direct stop codon
Direct strand coding state:
Atypical gene of length *j* nt
Non-coding state of length *n* nt

Note that hidden states have different lengths!

The key to the whole thing

We’re going to find the trajectory (series of states) that has the highest probability of occurring with the sequence

S = the sequence $\{b_1b_2\ldots b_L\}$

A^* = the best trajectory

a_id_i = the length of sequence (d) assigned to state a_i

$$\underline{P_{\max}} = P(\underline{A\,*, S}) = \max_{(a_1d_1)\ldots(a_Md_M)} \underline{Prob\left\{(a_1d_1)(a_2d_2)\ldots(a_Md_M), b_1b_2\ldots b_L\right\}}$$

Probabilities!!

EMISSION

$p_{a_m}(d_m)$ Probability of duration d_m for state a_m

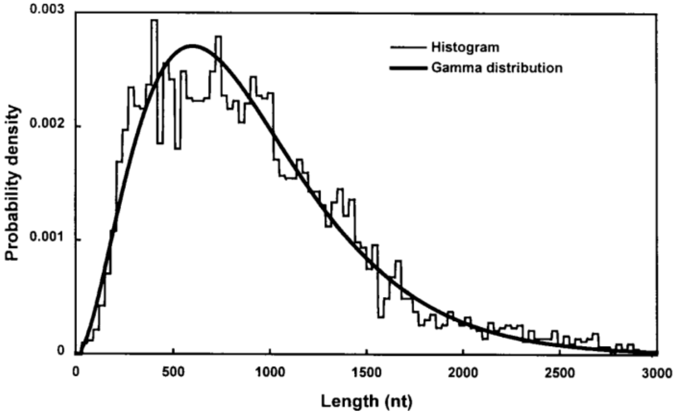
$p_{am}(b)$ Probability of subsequence b being observed in state a_m

TRANSITION

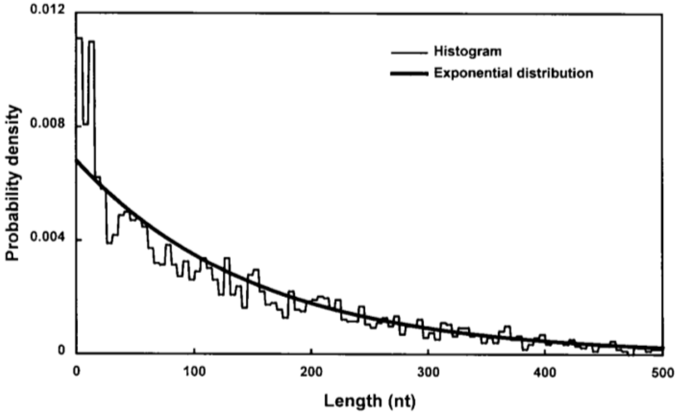
$q_{a_{m-1}a_m}$ Probability of change from state m – 1 to state m

And that gives us Viterbi, forward, and backward algorithms

Parameters!



Length distribution of genes in *E. coli*



Length distribution of intergenic regions in *E. coli*

Sequence probabilities are based on codon-aware (for coding sequence) and homogeneous (for non-coding sequence) Markov models

Genome	Genes annotated	Genes predicted	Exact prediction (%)	Missing genes (%)	Wrong genes (%)
A.fulgidus	2407	2530	73.1	10.8 (2.0)	15.1
B.subtilis	4101	4384	77.5	3.6 (2.8)	9.8
E.coli	4288	4440	75.4	5.0 (2.7)	8.2
H.influenzae	1718	1840	86.7	3.8 (3.2)	10.2
H.pylori	1566	1612	79.7	6.0 (4.4)	8.7
M.genitalium	467	509	78.4	9.9 (1.7)	17.3
M.jannaschii	1680	1841	72.7	4.6 (0.8)	12.9
M.pneumoniae	678	734	70.1	7.8 (4.1)	13.6
M.thermoauthotrophicum	1869	1944	70.9	5.0 (3.5)	8.6
Synechocystis	3169	3360	89.6	4.0 (1.5)	9.4
Averaged	21 943	23 194	78.1	5.4 (2.7)	10.4

From reference database (GenBank)

Start and stop codons the same

“false negatives”

“false positives”

Refinements to GeneMark.hmm

Look for ribosome binding site in preceding gene to identify overlaps (1998)

Model construction from very small datasets (1999)

Unsupervised application to any prokaryotic genome (2005)

Mapping of RNA reads to identify intron / exon boundaries (2014)

Using protein databases for “hints” (2020)

Advantages of HMMs

- Probabilistic framework – the forward algorithm returns the probability of the data (sequence) given the model (the HMM)

- Eminently tweakable – can be designed carefully to capture the patterns in biological sequences

Disadvantages

- Must be designed carefully to adequately capture the patterns in biological sequences
    - Or, use a generic framework

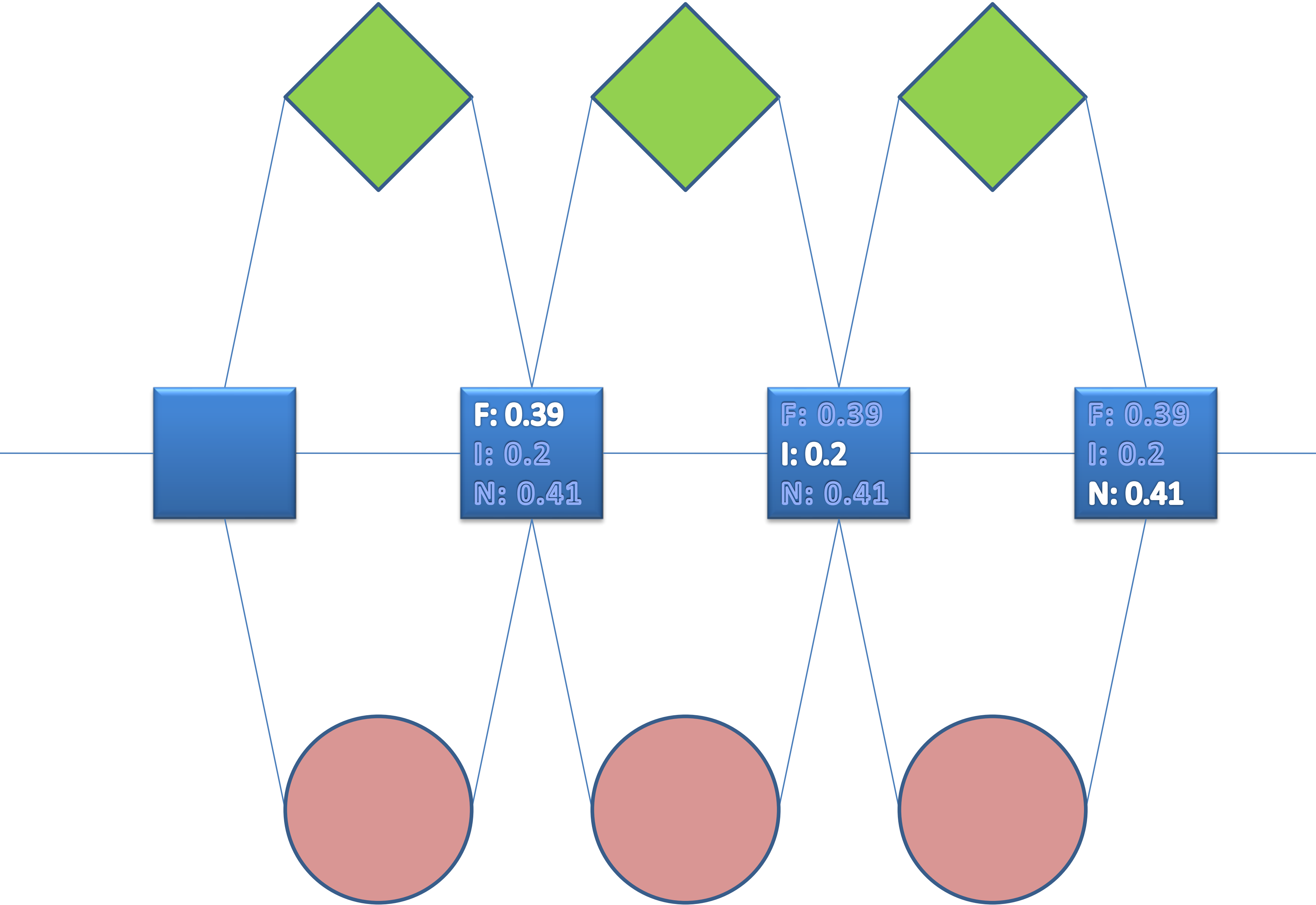
- Can be computationally expensive (kind of like DP for sequence alignment)

- It’s Markovian, so you cannot represent correlations of matches at different sites

Implementations

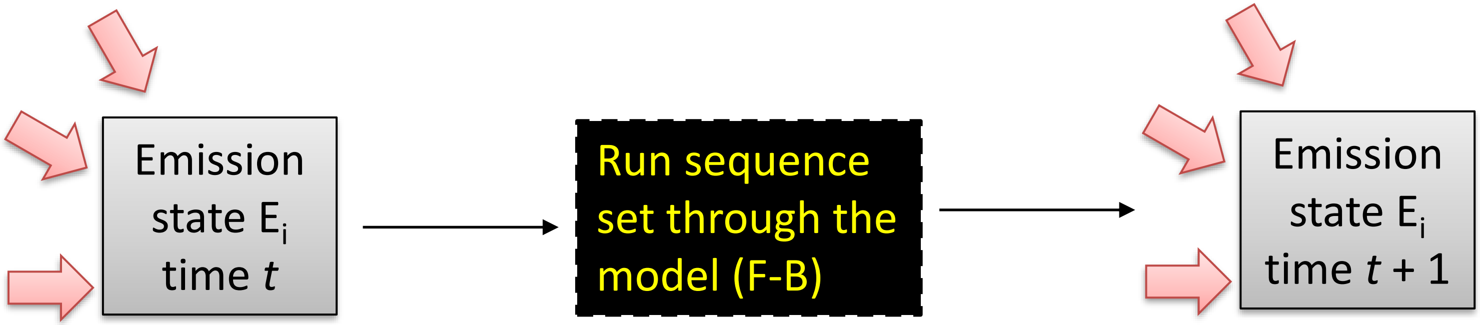
- HMMER (http://hmmer.janelia.org/)

- SAM (http://compbio.soe.ucsc.edu/sam.html)



F: 0.39
I: 0.2
N: 0.41
F: 0.39
I: 0.2
N: 0.41
F: 0.39
I: 0.2
N: 0.41

(Baum-Welch in depth)



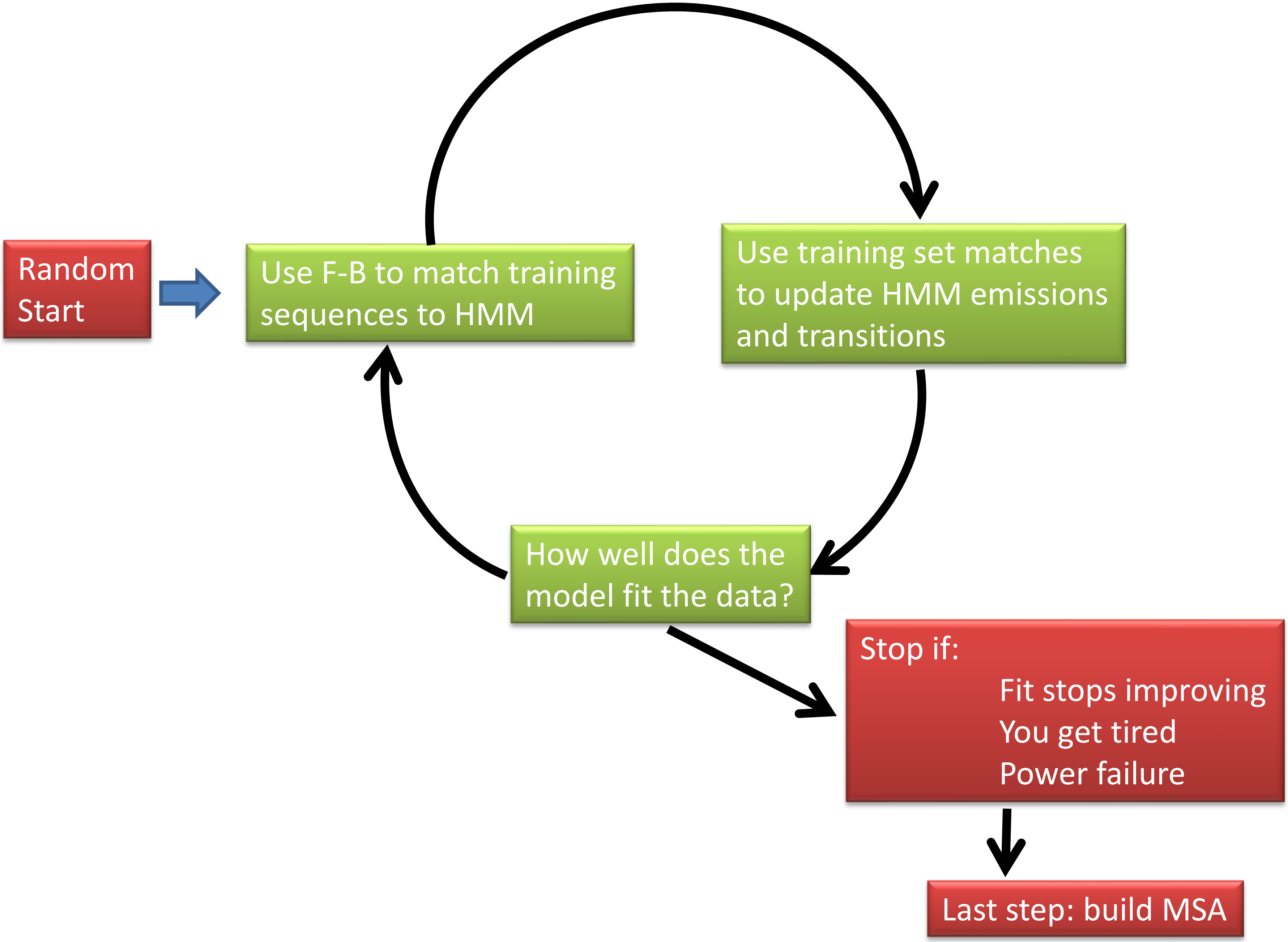
Emission state E_i time *t*

Run sequence set through the model (F-B)

Emission state E_i time *t* + 1

Old emission frequencies:
A = 0.2
B = 0.05
& = 0.03
Я = 0.1
ℵ= 0.01
🔔 = 0.07

New emission frequencies:
A = 0.24
B = 0.06
& = 0.01
Я = 0.11
ℵ= 0.21
🔔 = 0.02



Random Start

Use F-B to match training sequences to HMM

Use training set matches to update HMM emissions and transitions

How well does the model fit the data?

Stop if:

- Fit stops improving
- You get tired
- Power failure

Last step: build MSA

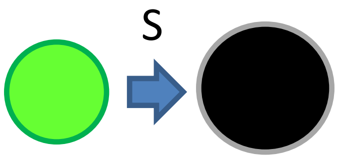
Problem with Baum-Welch

- Gradient descent, therefore sensitive to random starting conditions!

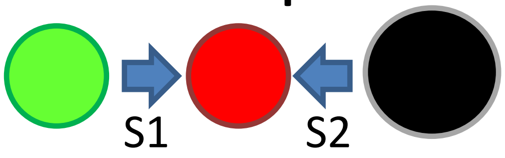
- You can try multiple starting points, or methods that perturb the probabilities to try and escape local optima

Example: Remote homology searching

- What is my mystery sequence?



  - Take sequence of interest
  - Compare against a database using algorithm X
  - Identify statistically significant alignments
- Instead of comparing against a set of individual sequences, we can instead compare to:
  - Intermediate sequence



  - Profile
  - HMM

Park et al., J. Mol. Biol. (1998)

- Contrast these various approaches on a reference set from the Structural Classification of Proteins (SCOP) database

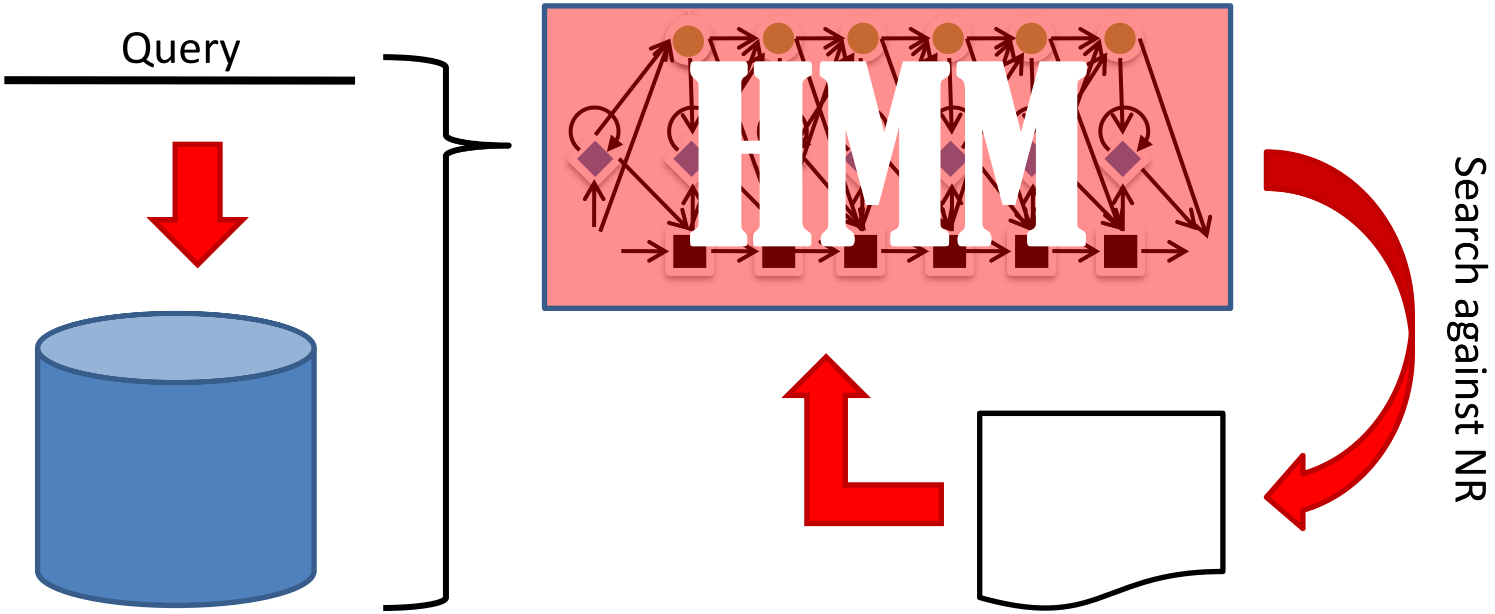
- A challenging problem – low (<40%) sequence identity means potentially lots of false negatives

Data Set

- Homologous superfamilies in the SCOP database
- Collect a total of 935 proteins
  - 436,645 (935 choose 2) pairs
  - Of these, 2096 pairs (0.48%) are definitely homologous
  - 1896 pairs are of uncertain relationship (same protein fold, uncertain homology)

HMM training

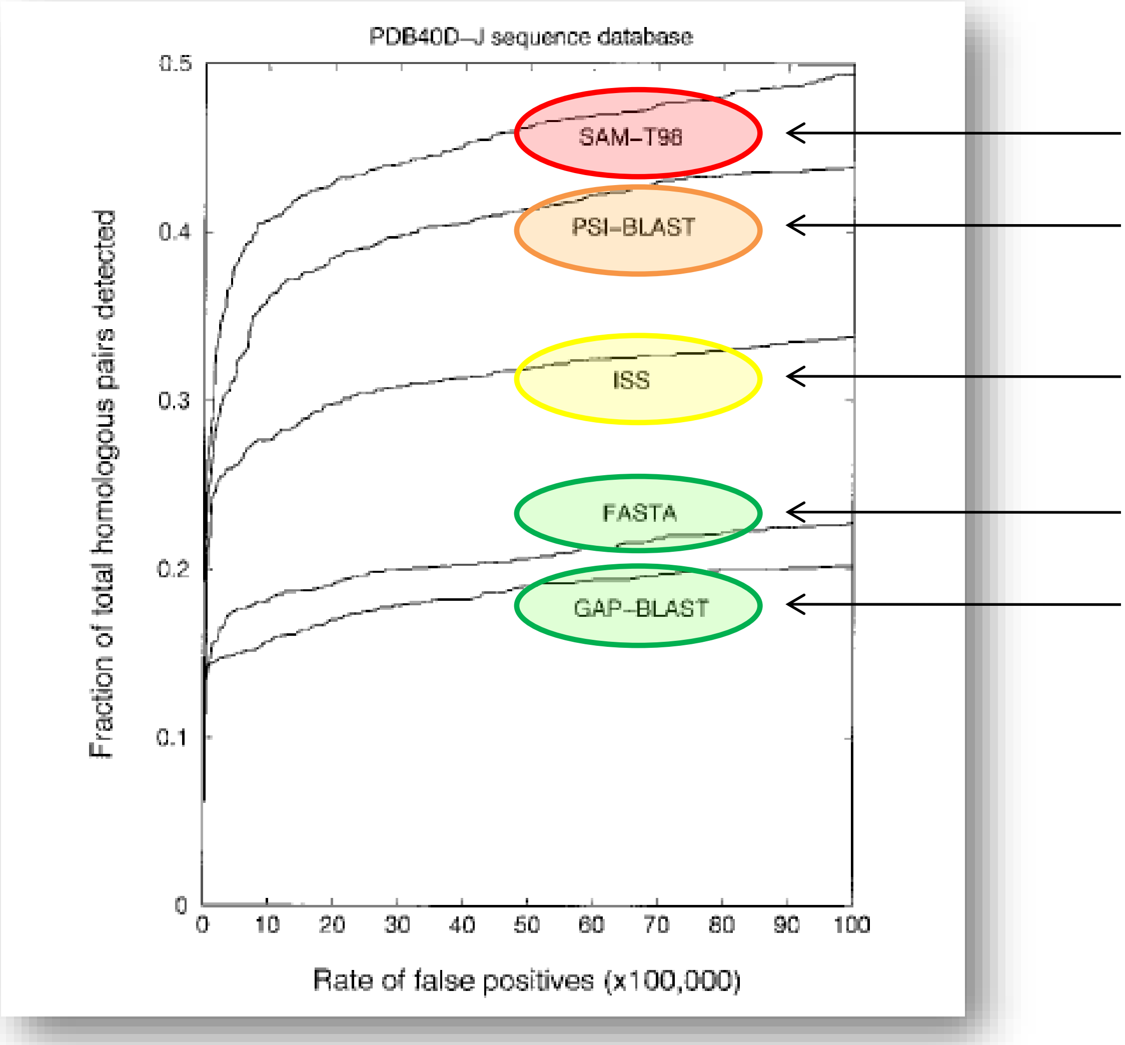
For each of the 935 sequences:



Results

×3

Stringent search against NR
(non-redundant protein) database



HMM (amazing)

Profile (coming next lecture)

Intermediate sequence
(not really used)

Pairwise methods (straightforward)

Also

- Detection of short homologous sequences
- Intron / exon boundaries
- Transmembrane domains
- Other 2D and 3D structural features
- Protein-protein interactions
- Gene predictions (GeneMark)
- Recombined regions in DNA
- Evolutionary rate variation
- Free babysitting