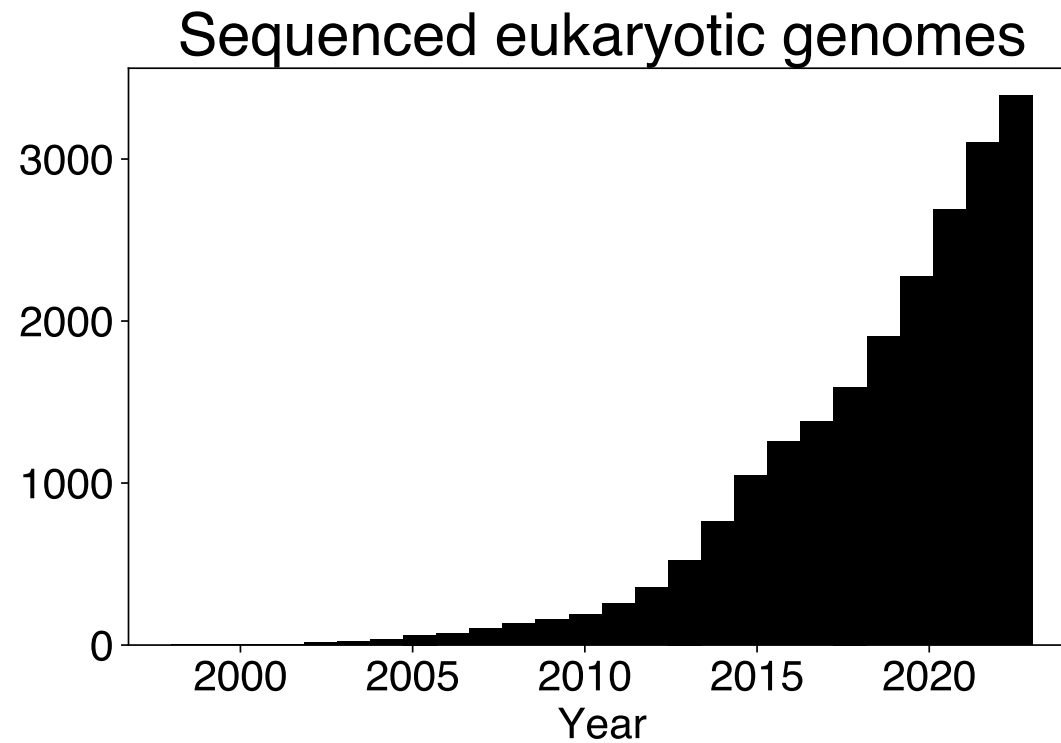


# Hidden Markov Models

# Analyzing genome sequences to learn about regulation



AGTAGTGTGTGCCCCGTCTGTTGTGTGACTCTGGTAGCTAGAGATCCCTCAGACCCTTTGTGGTAGTGTGGA  
AAATCTCTAGCAGTGGCGCCCGAACAGGGACTTAAAAGCGAAAGTAAGACCAGAGGAGATCTCTCGACGCA  
GGACTCGGCTTGCTGAAGTGCACTCGGCAAGAGGGCGAGAGGGGGCGGCTGGTGAGTACGCCATTTTTTTATTT  
GACTAGCGGAGGCTAGAAGGAGAGAGATGGGTGCGAGAGCGTCAATATTAAGAGGGCGAAAAATTAGATAAA  
TGGGAAAGAATTAGGTTAAGGCCAGGGGGGAAAGAAAAGCTATATGATATAGCACTTAATATGGGCAAGCAG  
GGAGCTGGAAAGATTTGCACTCAACTCTGGCCTTTTAGAAACATCAGGAGGCTGTAAACAAATAATGAAAC  
AGCTACAACCAGCTCTACAGACAGGAACAGAGGAACTTAAATCATTATATAACACAGTAGCAACTCTCTAT  
TGTGTACATGAAAAAATAGAAGTACGAGACACCAAGGAAGCCTTAGACAAGATAGAGGAAGAACAAAACAA  
AAGTCAGCAAAAAACACAGCAGGCAGCTGACGGAAAGGTCAGTCAAAATTATCCTATAGTGCAGAATCTTC  
AAGGGCAAATGGTACATCAAGCCATATCACCTAGAACCTTGAATGCATGGGTAAAAGTAATAGAGGAGAAG  
GCTTTTAGCCCAGAGGTAATACCCATGTTTACAGCATTATCAGAAGGAGCCACCCCACAAGATTTAAACAC  
CATGTTAAATACGGTGGGGGGACATCAAGCAGCCATGCAAATGTTAAAGGATACCATCAATGAAGAGGCTG  
CAGAATGGGATAGATTACATCCAGTACATGCGGGGCCTATTGCACCAGGCCAAATGAGAGAACCAAGGGGA  
AGTGACATAGCAGGAACACTACTAGTACCCTTCAGGAACAAATATCATGGATAACAGGTAACCCACCTATTCC  
AGTGGGAGAAATCTATAAAAGATGGATAATTCTGGGGTTAAACAAAATAGTGAGAATGTATAGCCCTGTCA  
GCATTTTGGACATAAGACAAGGGCCAAAGGAACCCTTTAGAGACTATGTAGATCGGTTCTTTAAAACTTTA  
AGAGCTGAACAAGCTACACAAGATGTAAAAAATTGGATGACAGACACCTTGTTGGTCCAAAATGCGAACCC  
AGATTGTAAGACCATTTTTAAGAGCATTAGGACCAGGGGGCCACATTAGAAGAAATGATGACAGCATGTCAGG  
GAGTGGGAGGACCTGGCCACAAAGCAAGAGTGTTGGCTGAGGCAATGAGCCAAGCTAACAATATAAACATA  
ATGATGCAGAGAAGCAATTTTAAAGGCTCTAAGAGAACTATTAAATGTTTCAACTGCGGCAAGGAAGGGCA  
CATAGCTAGAACTGCAGGGCCCCCTAGGAAAAAAGGCTGTTGGAAATGTGGTAAGGAAGGACACCAAATGA  
AAGACTGTACTGAGAGGCAGGCTAATTTTTTTAGGGAAAATTTGGCCTTCCCAGAAGGGGAGGCCAGGGAAT  
TTCCTTCAGAGCAGACCAGAGCCAACAGCCCCACCAGCAGAGAGCTTCAAGTTCGAGGAGACAACCCCCGT  
TCCGAAGCAGGAGCCGAAAGACAGGGAACCCTTAACCTCCCTCAAAT...

What is “written” in here?

# Markov Chains



Andrey Andreyevich Markov  
(1856-1922)- Russian  
mathematician known for his  
work on stochastic processes

# Markov Chains

Random process = a process in which the trajectory of the system is not precisely determined and it is described in terms of probability distributions.

Discrete random process = random process in which the system changes between discrete states. The parameters of these transitions are called transition probabilities.

Markov property : the probability distribution over states at the next time step depends only on the current state of the system

$$\Pr(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) = \Pr(X_{n+1} = x | X_n = x_n)$$

Markov chain = discrete random process with the Markov property.

More generally, Markov chain of order  $m$

$$\Pr(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \dots, X_n = x_n) = \Pr(X_{n+1} = x | X_n = x_n, \dots, X_{n-m+1} = x_{n-m+1})$$

# Probabilities of nucleotide sequences

Imagine that the genome (DNA sequence) is created by randomly picking each base, independently of all the other bases. This would correspond to a Markov chain of order  $m = 0$ .

Let  $(\pi_A, \pi_C, \pi_G, \pi_T)$  be the probabilities of choosing each of the four nucleotides.

Let's further imagine that we do not know these probabilities but we want to estimate them from the resulting DNA sequence.

S = GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACC

How do we approach this question?

We know how!

We write the likelihood of the sequence given the model and we find the model parameters that maximize the likelihood of the data (the observed DNA sequence).

# Estimating nucleotide probabilities

Under our assumption (Markov model of order  $m = 0$ ), the likelihood of the sequence is  $P(s|\vec{\pi}) = (\pi_A)^{n_A} (\pi_C)^{n_C} (\pi_G)^{n_G} (\pi_T)^{n_T}$ , where  $n_A, n_C, n_G, n_T$  are the counts of the respective nucleotides in the sequence.

The log-likelihood will be

$$L(s|\vec{\pi}) = n_A \log(\pi_A) + n_C \log(\pi_C) + n_G \log(\pi_G) + n_T \log(\pi_T)$$

And its maximum needs to obey the constraint  $\pi_A + \pi_C + \pi_G + \pi_T = 1$ .

This can be solved with the method of Lagrange multipliers:

If we want to maximize a function  $f(\vec{x})$  subject to the constraint  $g(\vec{x}) = c$ , we define a function  $\Lambda(\vec{x}, \lambda) = f(\vec{x}) + \lambda(g(\vec{x}) - c)$  and solve for the partial derivatives of  $\Lambda(\vec{x}, \lambda)$  being 0.

# Estimating nucleotide probabilities

$$f(\vec{\pi}) = L(s|\vec{\pi}) = n_A \log(\pi_A) + n_C \log(\pi_C) + n_G \log(\pi_G) + n_T \log(\pi_T)$$

$$g(\vec{\pi}) = \pi_A + \pi_C + \pi_G + \pi_T = 1.$$

$$\Lambda(\vec{\pi}, \lambda) = n_A \log(\pi_A) + n_C \log(\pi_C) + n_G \log(\pi_G) + n_T \log(\pi_T) + \lambda(\pi_A + \pi_C + \pi_G + \pi_T - 1)$$

$$\frac{\partial \Lambda(\vec{\pi}, \lambda)}{\partial \pi_\alpha} = \frac{n_\alpha}{\pi_\alpha} + \lambda \text{ for any } \alpha \in \{A, C, G, T\}$$

$$\frac{\partial \Lambda(\vec{\pi}, \lambda)}{\partial \pi_\alpha} = 0 \Rightarrow \frac{n_\alpha}{\pi_\alpha} = -\lambda \text{ for any } \alpha \in \{A, C, G, T\}$$

$$\sum_{\alpha} \pi_\alpha = \sum_{\alpha} -\frac{n_\alpha}{\lambda} = 1 \Rightarrow n_A + n_C + n_G + n_T = n = -\lambda$$

And the maximum likelihood solution will be  $\pi_\alpha = \frac{n_\alpha}{n}$  for any  $\alpha \in \{A, C, G, T\}$

Note: we assumed that the sequence was generated by a Markov chain of 0<sup>th</sup> order.



# Modeling nucleotide sequences with Markov chains

Markov chain of order 1:

Probability to observe a given nucleotide at position  $i$  in the sequence  $x$  only depends on the nucleotide that was observed at position  $i - 1$  in the sequence

$P(x)$  – probability of sequence  $x$

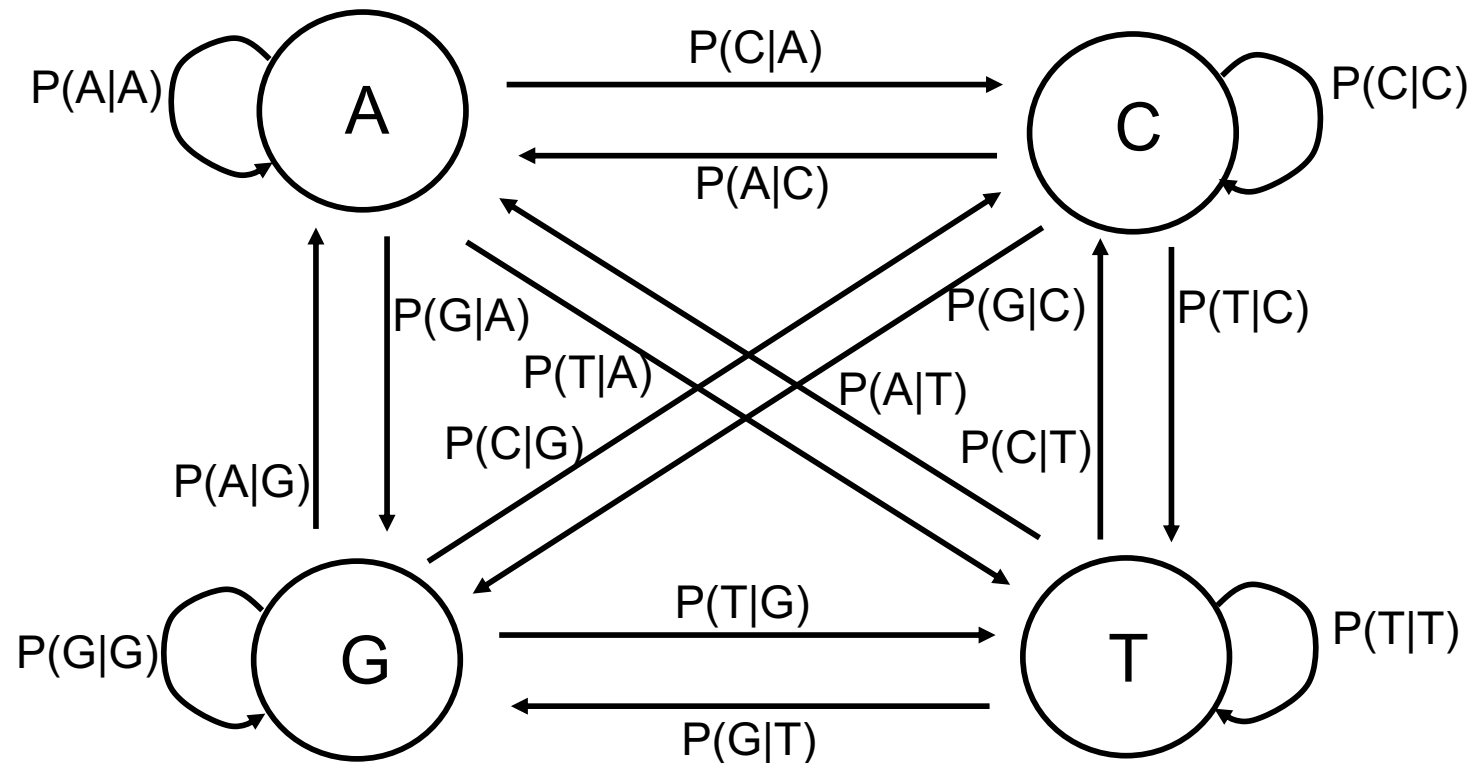
$$= P(x_L, x_{L-1}, \dots, x_2, x_1)$$

$$= P(x_L | x_{L-1}, \dots, x_2, x_1)$$

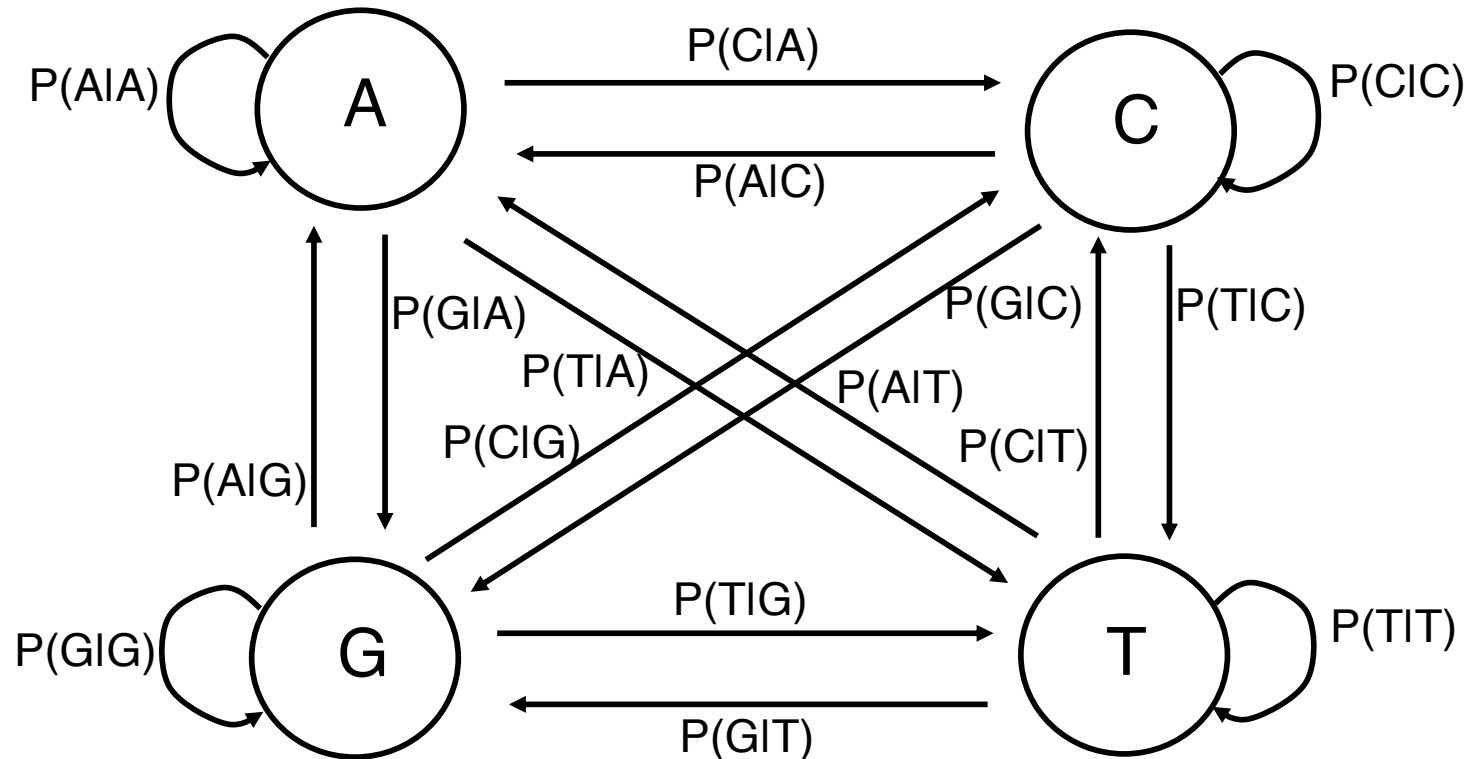
$$= P(x_L | x_{L-1}) P(x_{L-1} | x_{L-2}) \dots P(x_2 | x_1) P(x_1)$$

# Modeling nucleotide sequences with Markov chains

GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGTGGTACA  
CAGGTTAGGAGAGGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAGTGAAGTCCACTA  
GGAAGTGAAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAGAGAAAGAGCACCCGCACT  
GGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTGACAGCGGGAGTGAGCCCCTCTCAA  
AAACTGATGCCAACTACGCAGGACAGAGAGGGGGGCGGGGAAGGGGGGAGTGACCTGAGGGGAGACT  
GGGGCTCAAGAAAAGCCTTTTTTGTGTTGGTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTA  
GTTCTAAGGAGAGGAA



# Modeling nucleotide sequences with Markov chains



How do we get transitions probabilities?

Transitions correspond to di-nucleotides in the sequence.

We know how to obtain maximum likelihood estimates of mono-nucleotide frequencies.

Similarly we obtain estimates of di-nucleotide frequencies.

# Modeling nucleotide sequences with Markov chains

From\To	A	C	G	T	Total
A	$n_{AA}$	$n_{AC}$	$n_{AG}$	$n_{AT}$	$n_{A.}$
C	$n_{CA}$	$n_{CC}$	$n_{CG}$	$n_{CT}$	$n_{C.}$
G	$n_{GA}$	$n_{GC}$	$n_{GG}$	$n_{GT}$	$n_{G.}$
T	$n_{TA}$	$n_{TC}$	$n_{TG}$	$n_{TT}$	$n_{T.}$

We previously found that the maximum likelihood estimates of mono-nucleotide frequencies were  $\pi_{\alpha} = \frac{n_{\alpha}}{n}$  with  $n = n_A + n_C + n_G + n_T$

Similarly, we can work out that the maximum likelihood estimates of di-nucleotide frequencies are given by  $\pi_{\alpha\beta} = \frac{n_{\alpha\beta}}{n_{\alpha.}}$  with  $n_{\alpha.} = \sum_{\beta} n_{\alpha\beta}$

# Using Markov chains to test hypotheses about sequences

Say we have multiple models of the sort:

$$P(\vec{x}) = P(x_L|x_{L-1}) P(x_{L-1}|x_{L-2}) \dots P(x_2|x_1) P(x_1)$$

From\To	A	C	G	T	From\To	A	C	G	T
A	0.20871	0.24734	0.43353	0.12142	A	0.30383	0.17818	0.28970	0.22829
C	0.15532	0.35906	0.29398	0.19163	C	0.32426	0.28058	0.06187	0.33328
G	0.16185	0.34990	0.36959	0.11866	G	0.26985	0.21243	0.29660	0.22112
T	0.10273	0.34874	0.34338	0.20515	T	0.18643	0.21744	0.28698	0.30916

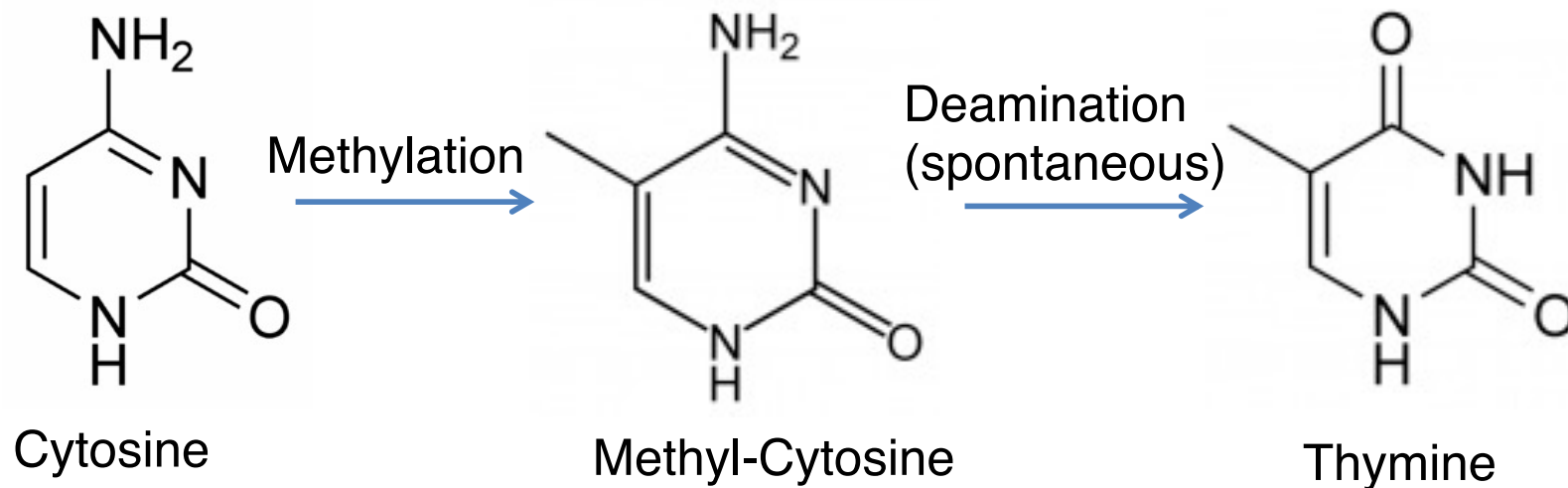
GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGTGGTACA  
CAGGTTAGGAGAGGGGGGAAGGGCAGAGTTTACATTGCCCGTATGCTGGCGAGTGAAGTCCACTA  
GGAAGTGAAGACATGAAGTTGAGGCTTAGCAAAAGAGAGCGACTTAGAGAAAGAGCACCCGCACT  
GGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTGACAGCGGGAGTGAGCCCCCTCTCAA  
AAAGTGAAGCAAGTACGCAGGACAGAGAGGGGGGCGGGGAAGGGGGAGTGACCTGAGGGGAGACT  
GGGGCTCAAGAAAAGCCTTTTTTGTGTTGGTTGTTTTAAAGGCTGGCGATACTGTAGCATGCTTA  
GTTCTAAGGAGAGGAA

Which model is more likely to have given rise to this sequence?

# Application: Detection of CpG islands

CpG island: region of with relatively high frequency of C+G nucleotides and CpG dinucleotides (C-phosphate bond-G).

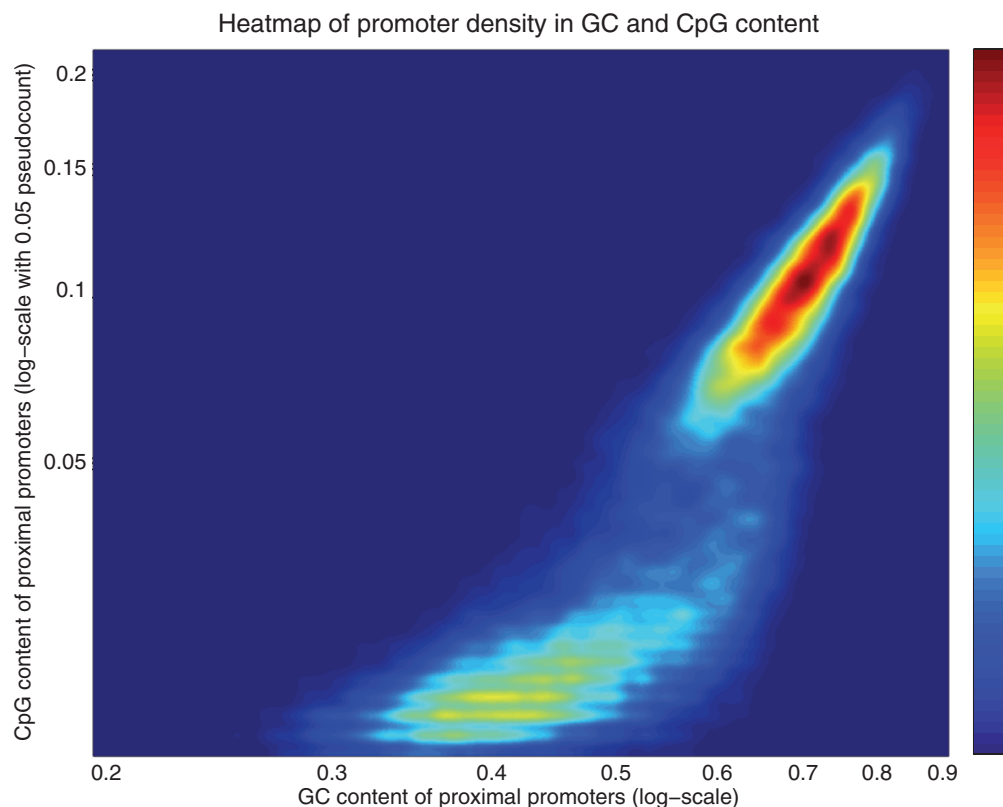
- frequently associated with promoter regions
- while CpGs have a high propensity to mutate, CpGs from CpG islands somehow remain demethylated



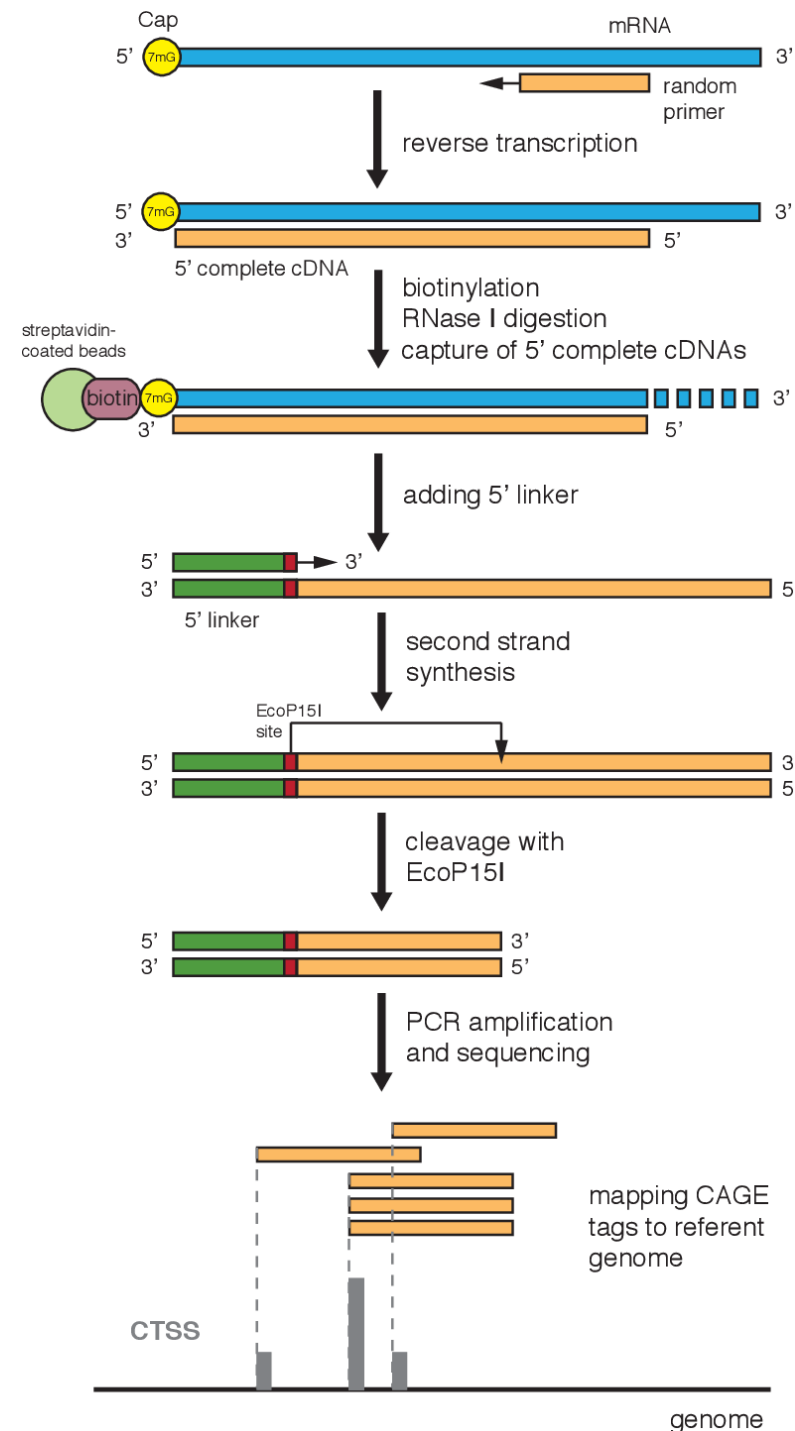
Finding CpG islands in a genome sequence translates into finding promoter regions

# Using Markov chains to detect CpG islands

We need some data to train a model:  
CAGE sequencing

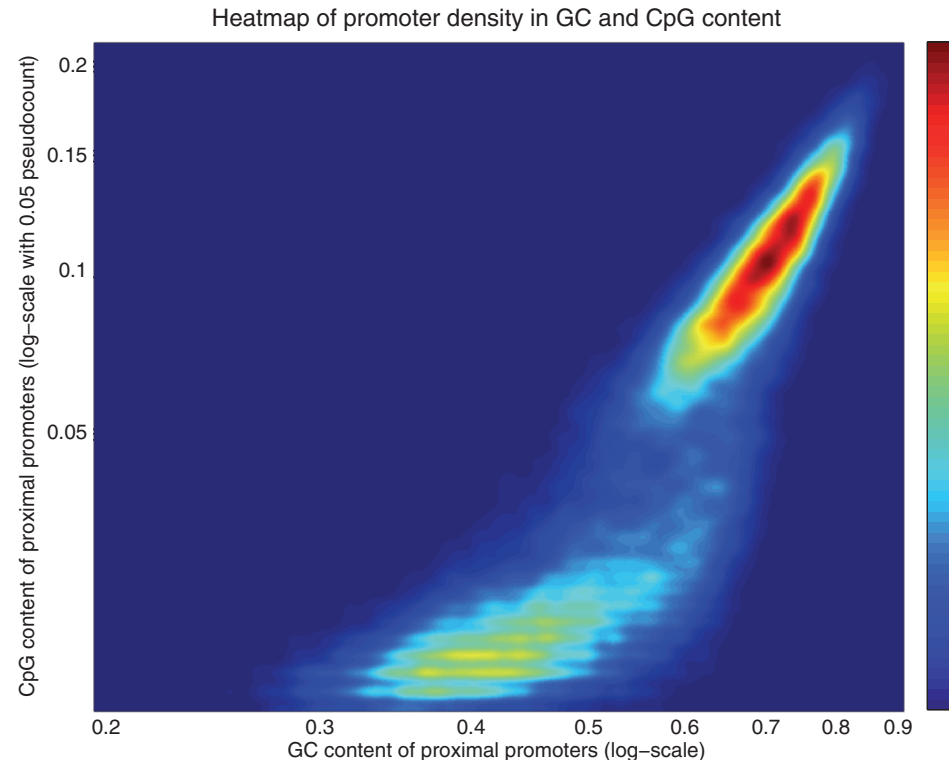


Proximal promoter: region of 0.5-1kb  
upstream of observed transcription start



# Using Markov chains to detect CpG islands

Data from Balwierz et al. Genome Biology 10:R79 (2009)



- There are 2 obvious clusters of promoter sequences that we can use to construct 2 Markov models (CpG and non-CpG).
- We then phrase the problem of deciding if a particular genomic sub-sequence is a CpG island as a discrimination problem: does the sub-sequence have higher likelihood under the CpG model or the non-CpG model?



# Detection of CpG islands

Given a sequence

```
GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG
ACAGCGGGAGTGAGCCCCTCTCAAAAAGTATGCCAACTACGCAGGACAGAGAGGGGGG
CGGGGAAGGGGGAGTGACCTGAGGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA
```

We compute the log-odds ratio:

$$S(x) = \log \left( \frac{P(\vec{x} | \text{CpG model})}{P(\vec{x} | \text{non-CpG model})} \right) \\ = \log(P(\vec{x} | \text{CpG model})) - \log(P(\vec{x} | \text{non-CpG model}))$$

We typically work in the log space to avoid underflows

# Detection of CpG islands

GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAAGTATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the CpG island model

$$L_{CpG}$$

# Detection of CpG islands

GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAAGTATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the CpG island model

$$L_{cpG} = \log(p_{cpG}(G))$$

# Detection of CpG islands

GAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAAGTATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the CpG island model

$$L_{cpG} = \log(p_{cpG}(G)) \\ + \log(p_{cpG}(G|G))$$

# Detection of CpG islands

GGAAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAACCTGATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the CpG island model

$$\begin{aligned} L_{CpG} = & \log(p_{CpG}(G)) \\ & + \log(p_{CpG}(G|G)) \\ & + \log(p_{CpG}(A|G)) \end{aligned}$$

# Detection of CpG islands

GG**A**CCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAACCTGATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the CpG island model

$$\begin{aligned} L_{CpG} = & \log(p_{CpG}(G)) \\ & + \log(p_{CpG}(G|G)) \\ & + \log(p_{CpG}(A|G)) \\ & + \log(p_{CpG}(A|A)) \end{aligned}$$

# Detection of CpG islands

GGA<sup>A</sup><sup>C</sup>CAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCCTCTCAAAAACCTGATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the CpG island model

$$\begin{aligned} L_{CpG} = & \log(p_{CpG}(G)) \\ & + \log(p_{CpG}(G|G)) \\ & + \log(p_{CpG}(A|G)) \\ & + \log(p_{CpG}(A|A)) \\ & + \log(p_{CpG}(C|A)) \end{aligned}$$

# Detection of CpG islands

GGAA**CA**AAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCCTCTCAAAAACCTGATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the CpG island model

$$\begin{aligned} L_{cpG} = & \log(p_{cpG}(G)) \\ & + \log(p_{cpG}(G|G)) \\ & + \log(p_{cpG}(A|G)) \\ & + \log(p_{cpG}(A|A)) \\ & + \log(p_{cpG}(C|A)) \\ & + \log(p_{cpG}(C|C))... \end{aligned}$$



# Detection of CpG islands

GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAAGTATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the non-CpG island model

$$L_{\overline{CpG}}$$

# Detection of CpG islands

GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAAGTATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the non-CpG island model

$$L_{\overline{CpG}} = \log(p_{\overline{CpG}}(G))$$

# Detection of CpG islands

GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAAGTATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

Log-probability of the sequence under the non-CpG island model

$$\begin{aligned} L_{\overline{CpG}} = & \log(p_{\overline{CpG}}(G)) \\ & + \log(p_{\overline{CpG}}(G|G)) \\ & + \log(p_{\overline{CpG}}(A|G)) \\ & + \log(p_{\overline{CpG}}(A|A)) \\ & + \log(p_{\overline{CpG}}(C|A)) \\ & + \log(p_{\overline{CpG}}(C|C))... \end{aligned}$$

# Detection of CpG islands

GGAACCAAGAACGAGGGGCAAGTGGGAGGAGGTGGTCACCTGGAGGGTGTGGACCAGT  
GGTACACAGGTTAGGAGAGGGGGAAGGGCAGAGTTTACATTGCCCCGTATGCTGGCGAG  
TGAAGTCCACTAGGAACTGAGACATGAACTTGAGGCTTAGCAAAAGAGAGCGACTTAG  
AGAAAGAGCACCCGCACTGGTGACTGTGGGCTGCATGGTGAAGGGGGGGCAAAGCAGTG  
ACAGCGGGAGTGAGCCCCTCTCAAAAACCTGATGCCAACTACGCAGGACAGAGAGGGGG  
CGGGGAAGGGGGAGTGACCTGAGGGGAGACTGGGGCTCAAGAAAAGCCTTTTTTGTGTTG  
GTTGTTTTTAAAGGCTGGCGATACTGTAGCATGCTTAGTTCTAAGGAGAGGAA

$$L_{cpg} - L_{\overline{cpg}} = -19.29347$$

Unlikely that this sequence corresponds to a CpG island.

# Detection of CpG islands

GGAA**CC**AAGAA**CG**AGGGG**CA**AGTGGGAGGAGGTGGT**CACCT**TGGAGGGTGTGGAC**CC**AGT  
 GGTAC**AC**AGGTTAGGAGAGGGGGGAAGGG**CA**GAGTTTACATTG**CCC****CG**TATG**CT**GG**CG**AG  
 TGAAGT**CC**ACTAGGA**CT**GAGACATGA**CT**TGAGG**CT**TAG**CA**AAAGAGAG**CG**AC**CT**TAG  
 AGAAAGAG**CA****CCC****CG**CA**CT**TGGTGAC**CT**GTGGG**CT**G**CA**TGGTGAAGGGGGGG**CA**AAG**CA**GTG  
 ACAG**CG**GGAGTGAG**CCCC****CT**CA**AAAA**CTGATG**CCA**CTA**CG**CAGGACAGAGAGGGGGG  
**CG**GGGAAGGGGGAGTGAC**CT**GAGGGAGAC**CT**GGGG**CT**CAAGAAAAG**CC**TTTTTTGTGTTG  
 GTTGTTTTAAAGG**CT**GG**CG**ATA**CT**GTAG**CA**TG**CT**TAGTT**CT**AAGGAGAGGAA

From\To	A	C	G	T
A	0.20871	0.24734	0.43353	0.12142
C	0.15532	0.35906	<b>0.29398</b>	0.19163
G	0.16185	0.34990	0.36959	0.11866
T	0.10273	0.34874	0.34338	0.20515

From\To	A	C	G	T
A	0.30383	0.17818	0.28970	0.22829
C	0.32426	0.28058	<b>0.06187</b>	0.33328
G	0.26985	0.21243	0.29660	0.22112
T	0.18643	0.21744	0.28698	0.30916

$$\frac{\text{CpG Observed}}{\text{CpG Expected}} = \frac{9}{26} = 0.35$$

A	C	G	T
110	68	151	71

# Detection of CpG islands

Let's take another candidate sequence

```
TCTCACTCACACACTCGCCCCACCCCGGGCTCCGGGCCAGCCGCGGGCCCTAGCGCCGA  
CTAAGACGCCCAAATCCAGCCGGCGGGCCGGGAGAAGAAGCCTCCAGGGAAAGCGAGGG  
TTAAACTTCCACCTCGCGCAGCTCTCCGTGCCCTGCGGAGACCACCCTCACCTCTGGC  
CTCCCCCACTCCCCTCCCCGATTTTAAAGGAGAGTCGGGGGGTCGCTGGGGAGCCTGAA  
GCACTAAGTTCCCACGACGGGGGGCTGTGGAAAGTCATTCATCACAGCAAAGCTCCCGG  
GGATGGGGATGGGGATGGGGATGGGGGGGCGGGGACGCCGCGGTGGGAGGGTGCGGAC  
GCCTGGCCCAGCCCCAGCTCGCCTCGCTCAGCCACCTGTAAGCAGGCCCGGG
```

# Detection of CpG islands

Let's take another candidate sequence

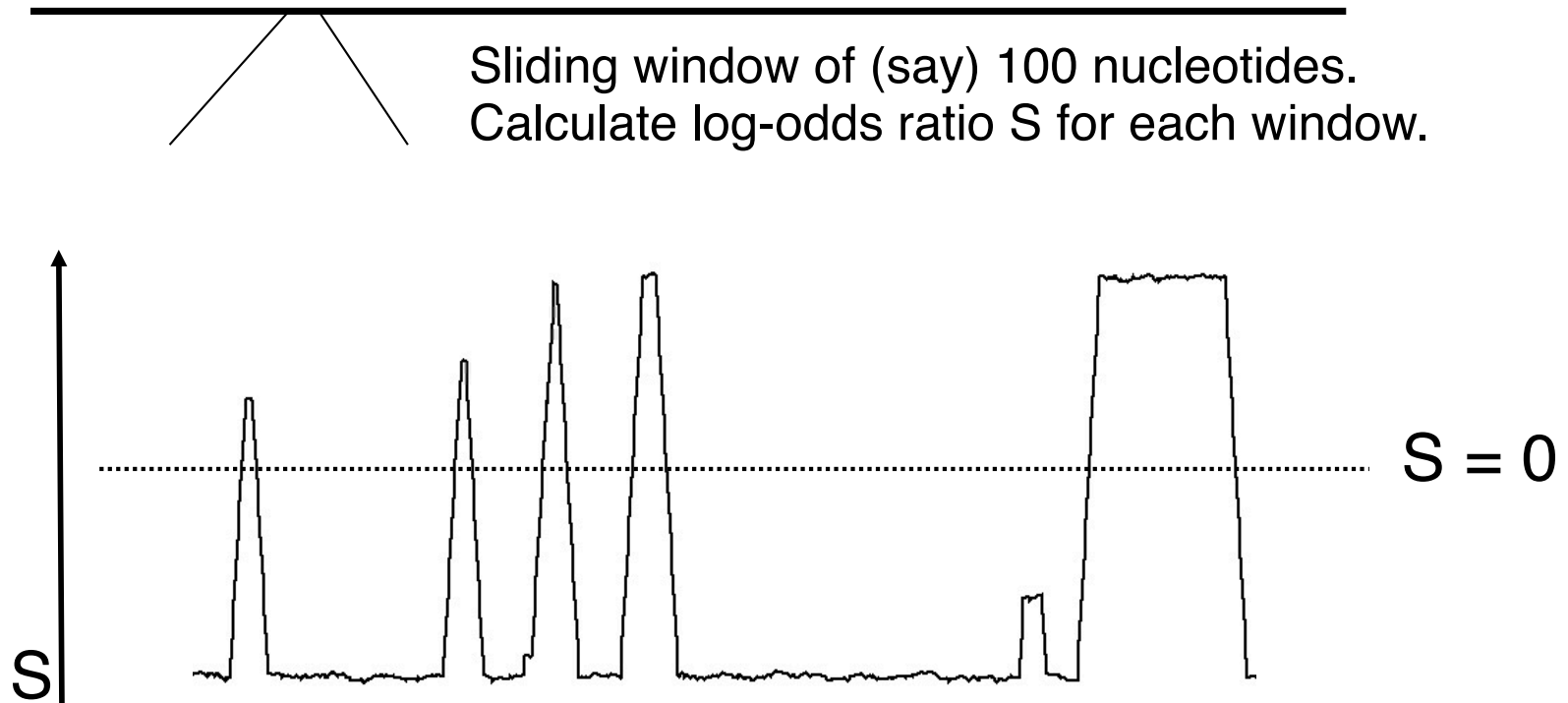
TCTCACTCACAACTCGCCCCACCCCGGGCTCCGGCCCAGCCGCGGCCCTAGCGCCGA  
CTAAGA CGCCCAAATCCAGCCGGCGGCCGGAGAAGAAGCCTCCAGGGAAAGCGAGGG  
TTAAACTTCCACCTCGCGCAGCTCTCGTGCCCTGCGGAGACCACCTCACCTCTGGC  
CTCCCCCACTCCCCTCCCGATTTTTAAGGAGAGTGGGGGTGCTGGGGAGCCTGAA  
GCACCTAAGTTCCCA CGA CGGGGGCTGTGGAAAGTCATT CATCACAGCAAAGCTCCCGG  
GGATGGGGATGGGGATGGGGATGGGGGGGCGGGGACGCCCGCGGTGGGAGGGTGCGGAC  
GCCTGGCCCAGCCCCAGCTCGCCTCGCTCAGCCACCTGTAAGCAGGCCCGGG

$$L_{CpG} - L_{\overline{CpG}} = 54.86262 \quad \frac{CpG \text{ Observed}}{CpG \text{ Expected}} = \frac{31}{47} = 0.66$$

A	C	G	T
72	142	131	55

Very likely that this sequence corresponds to a CpG island

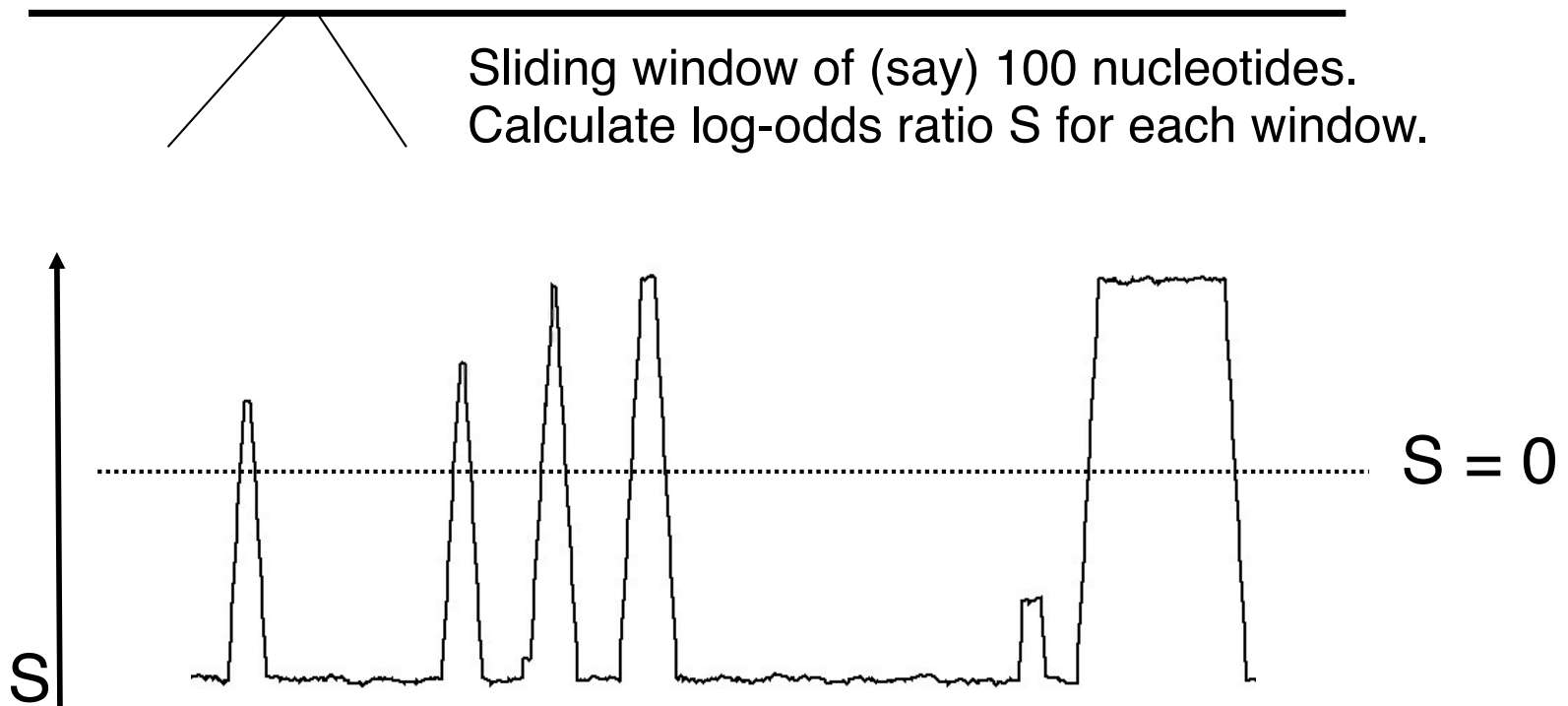
# Application to the entire genome



Predicted CpG islands are the “peaks” in the profile of  $S$ .

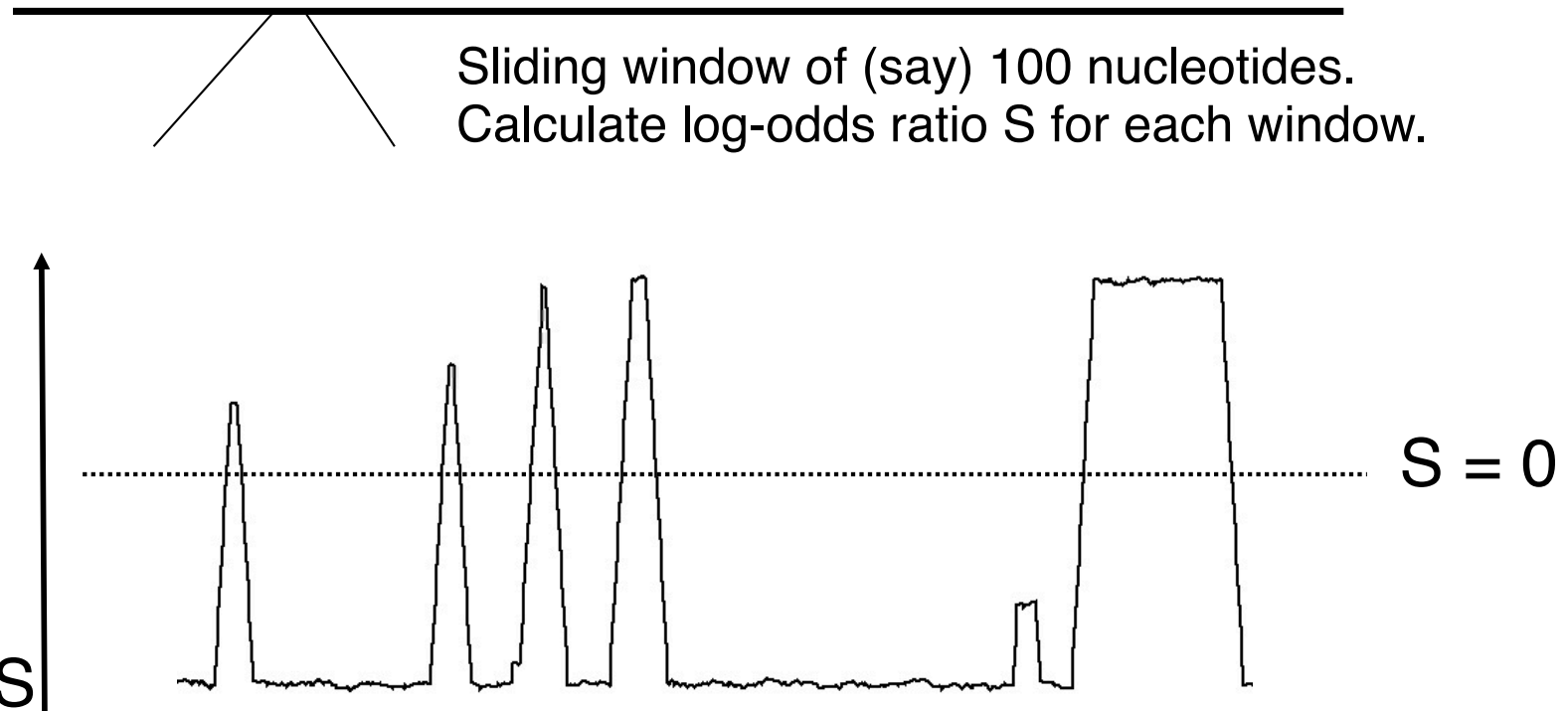


# Application to the entire genome



- We need to choose the length of sliding window without a clear justification
- We still need to set some arbitrary cut-offs to determine where CpG islands start and end

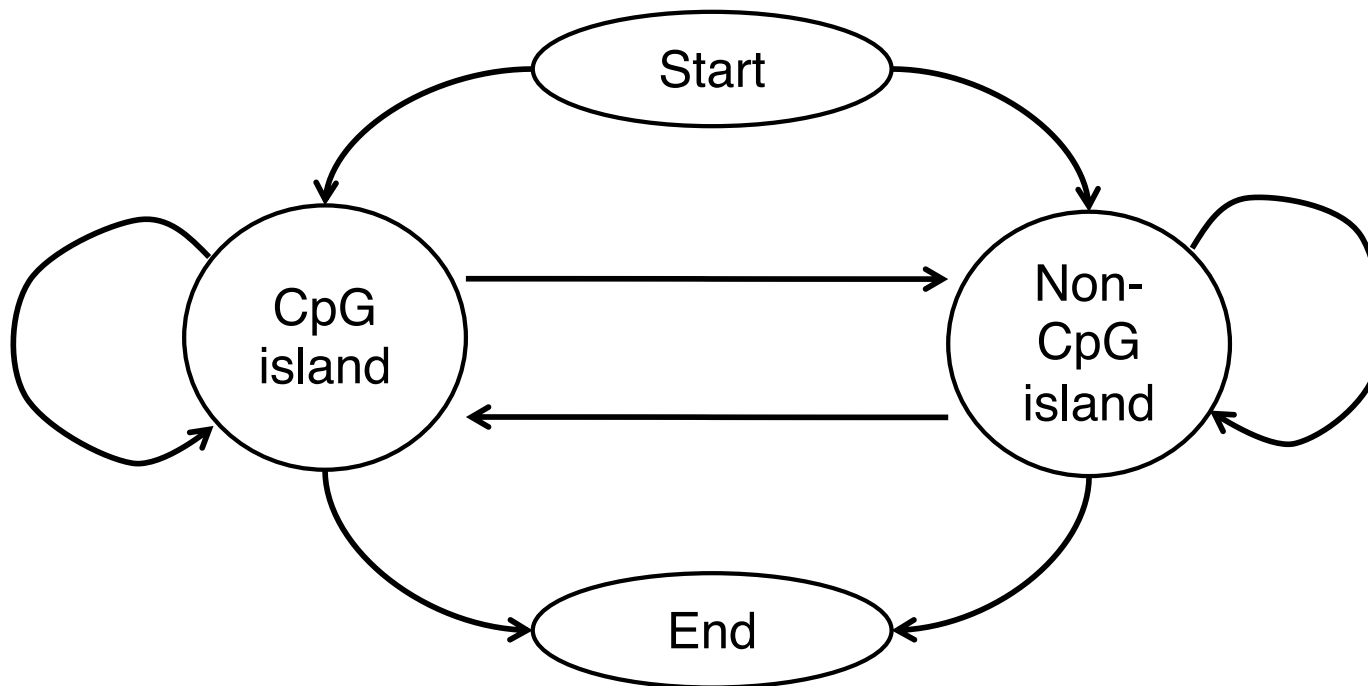
# Application to the entire genome



- We need to choose the length of sliding window without a clear justification
- We still need to set some arbitrary cut-offs to determine where CpG islands start and end
- Ideally, the delineation of the islands would be **part** of the Markov model
- **Solution:** Combine the Markov models of the CpG and no-CpG into a **Hidden Markov model**

# Hidden Markov models

- We model simultaneously the states (CpG island and non-CpG island) and the nucleotides that are observed in these states
- The path ( $\pi$ ) between states is modeled as a Markov chain
- Transitions in and out of the CpG island state specify the CpG island boundaries



# Hidden Markov models

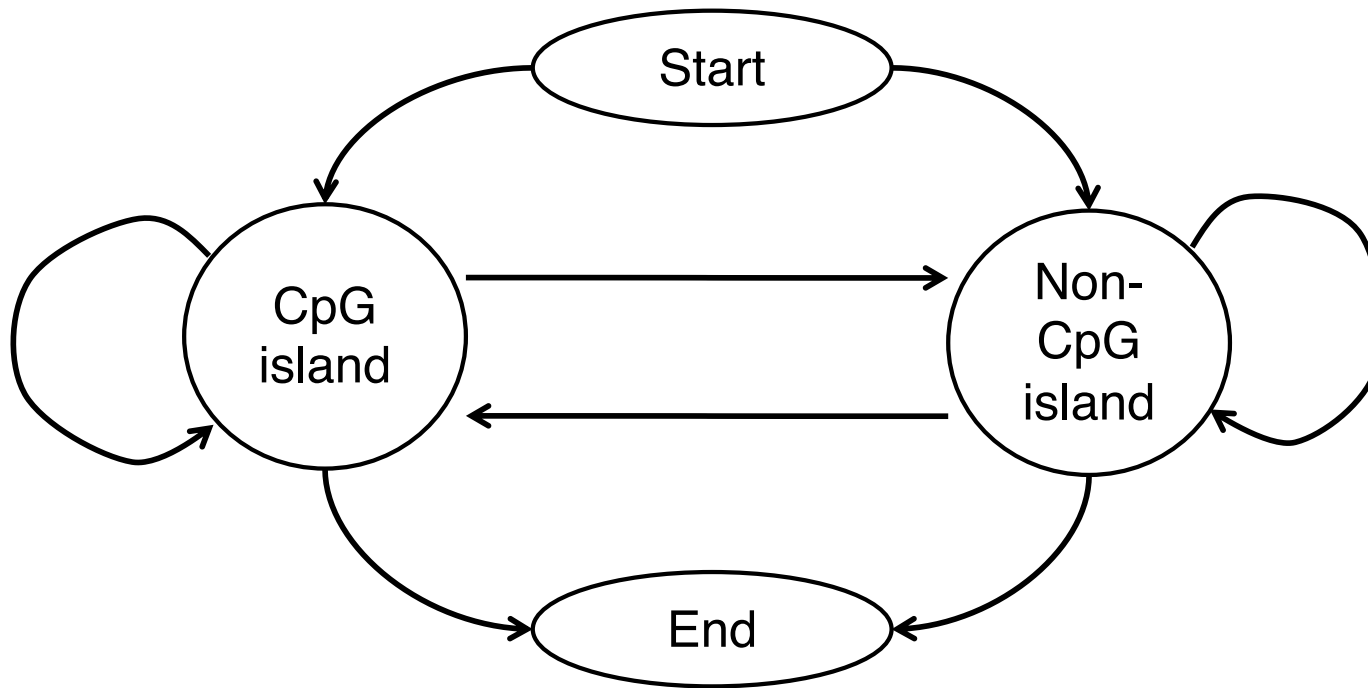
TCTCACTCACACACTCGCCCCACCCCGGGCTCCGGCCCCAGCCGCGGGCCCTAGCGCCGA  
CTAAGACGCCCAAATCCAGCCGGCGGGCCGGGAGAAGAAGCCTCCAGGGAAAGCGAGGG  
TTAAACTTCCACCTCGCGCAGCTCTCCGTGCCCTGCGGAGACCACCCTCACCTCTGGC  
CTCCCCCACTTCTCACTCACACACTCGCCCCACCCCGGGCTCCGGCCCCAGCCGCGGGCC  
CTAGCGCCGACTAAGACGCCCAAATCCAGCCGGCGGGCCGGGAGAAGAAGCCTCCAGGG  
AAAGCGAGGGTTAAACTTCCACCTCGCGCAGCTCTCCGTGCCCTGCGGAGACCACCCT  
CACCTCTGGCCTCCCCCACTCCCCTCCCCGATTTTTTAAGGAGAGTCGGGGGGTCGCTGG  
GGAGCCTGAAGCACTAAGTTCCACGACGGGGGGCTGTGGAAAGTCATTCATCACAGCA  
AAGCTCCCGGGGATGGGGATGGGGATGGGGATGGGGGGGGCGGGGACGCCGCGGTGGGA  
GGGTGCGGACGCCTGGCCCAGCCCCAGCTCGCCTCGCTCAGCCACCTGTAAGCAGGCC  
CGGGCCCCCTCCCCGATTTTTTAAGGAGAGTCGGGGGGTCGCTGGGGAGCCTGAAGCACTA  
AGTTCCACGACGGGGGGCTGTGGAAAGTCATTCATCACAGCAAAGCTCCCGGGGATGG  
GGATGGGGATGGGGATGGGGGGGGCGGGGACGCCGCGGTGGGAGGGTGCGGACGCCTGG  
CCCAGCCCCAGCTCGCCTCGCTCAGCCACCTGTAAGCAGGCCCGGG

We cannot tell where the CpG island is (the path through the model)  
simply by looking at the symbols, the states are “hidden”

# Hidden Markov models: model parameters

State-to-state transition probabilities: probability to go from state  $k$  to state  $l$

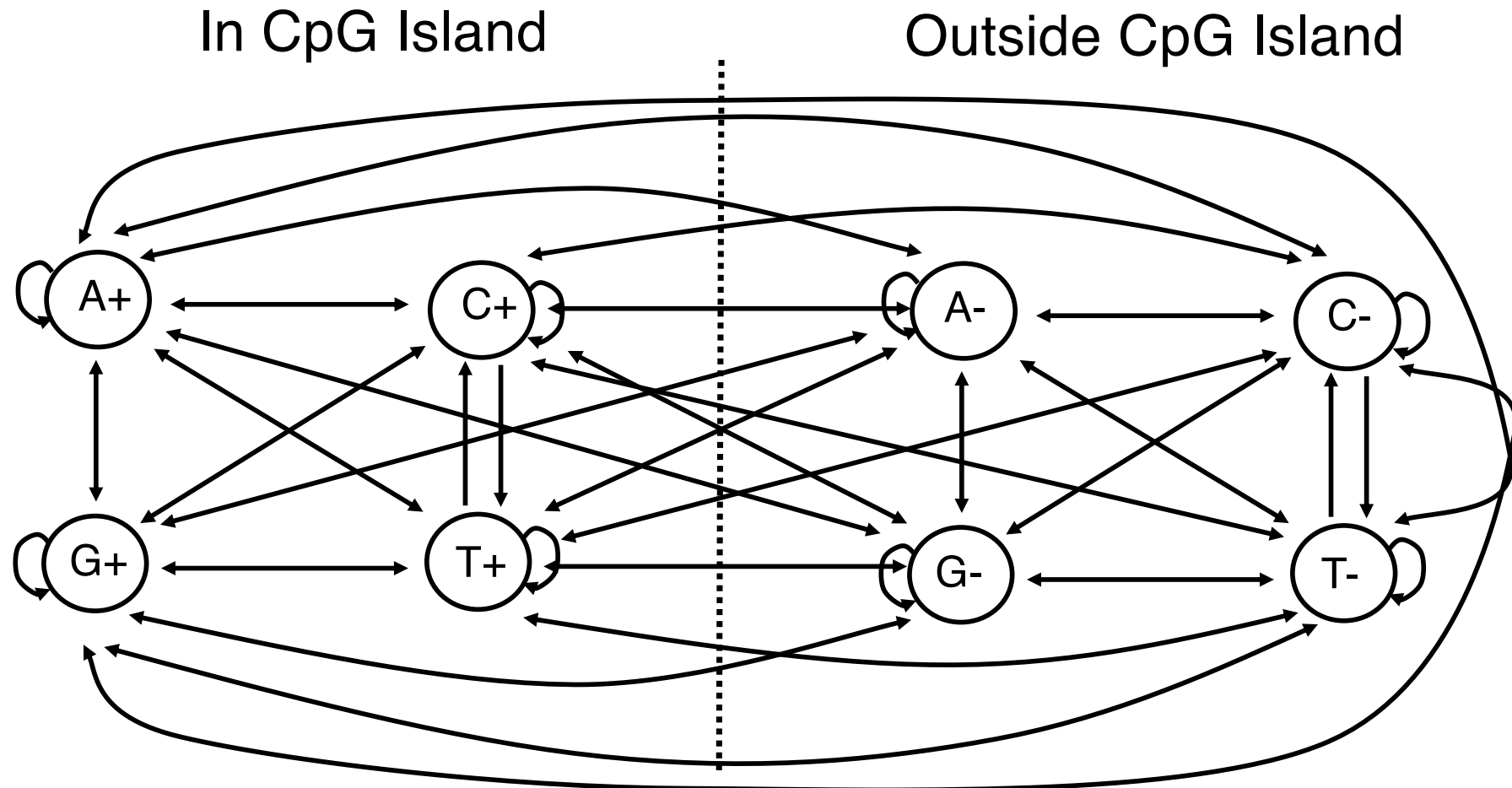
$$a_{kl} = P(\pi_i = l | \pi_{i-1} = k)$$



Emission probabilities:  $e_l(\beta) = P(x_i = \beta | \pi_i = l, x_{i-1} = \alpha)$

Probability to go from state  $k$  to state  $l$  and to emit letter  $\beta$ :  $e_l(\beta)a_{kl}$

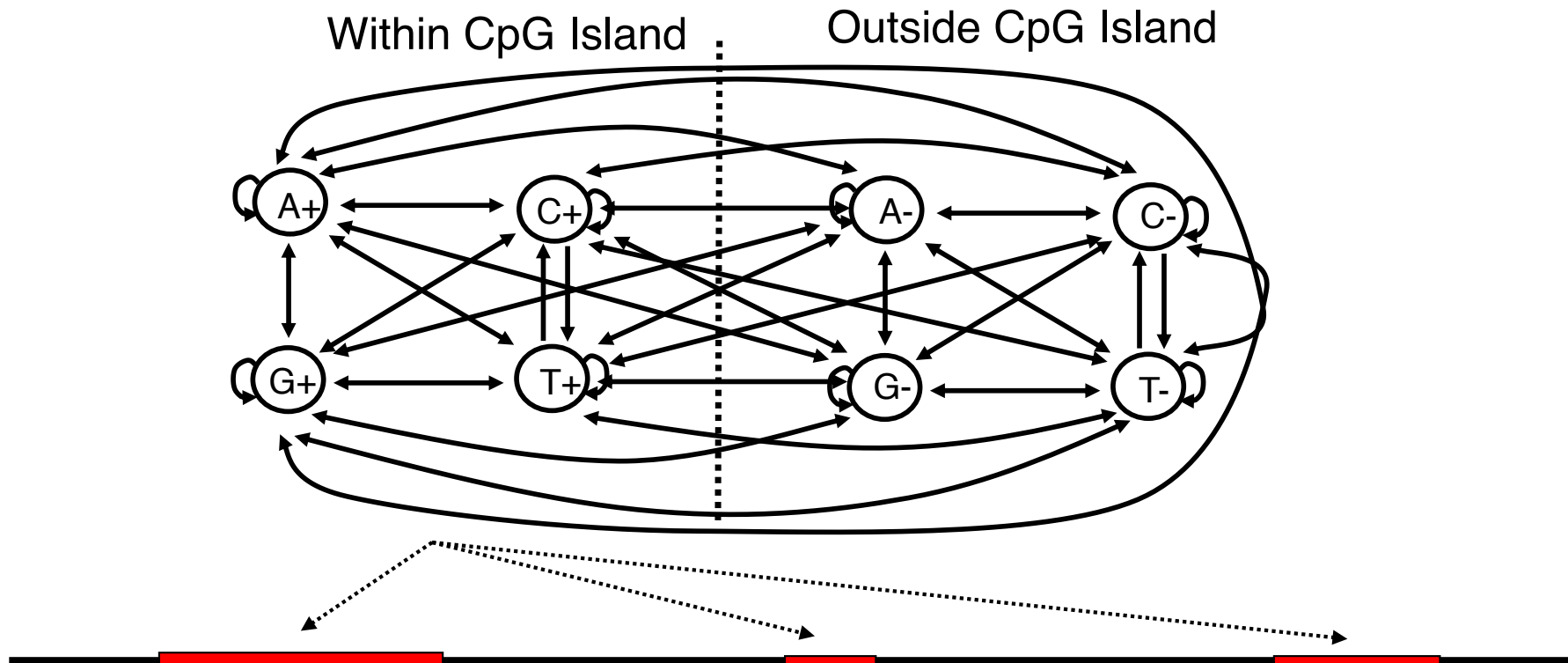
# Modeling CpG islands with hidden Markov models: Alternative representation



# Hidden Markov model paths

Taking a particular path through the hidden Markov model for sequence  $x$  is equivalent to an assignment of CpG islands to the sequence  $x$ .

Thus, the most likely path through the model given a sequence  $x$  corresponds to the most likely assignment of CpG islands through the sequence.



# Finding the most likely path: the Viterbi algorithm

Objective: find the highest probability path through a sequence  $x$

Approach: taking advantage of the Markov property, we will use a recurrence relation

Define  $v_k(i)$  = maximum probability that we can achieve after observing the first  $i$  letters of the sequence, ending in state  $k$

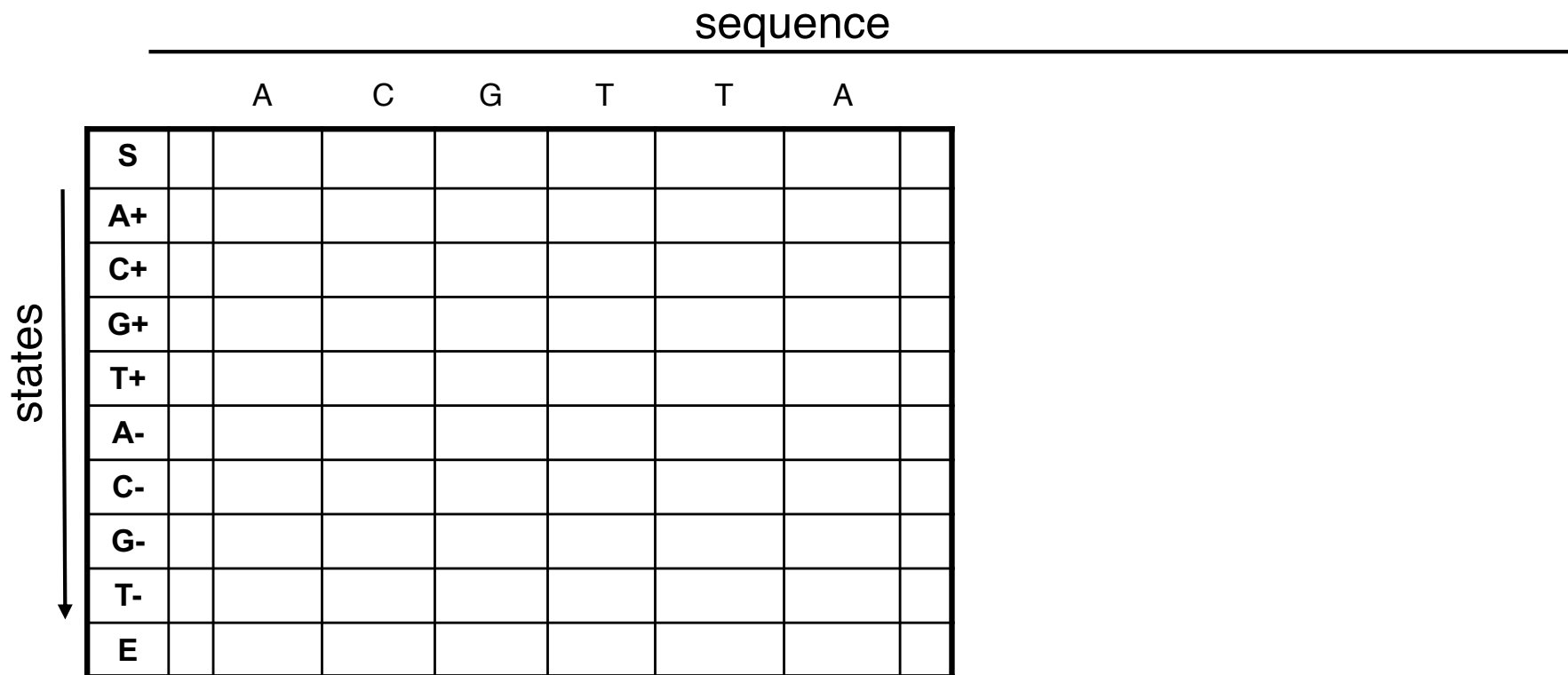
Recursion:  $v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$

To reconstruct the path, we have to keep track of the sequence of states  $k$  that gave us the maximum probability at each nucleotide  $i$  in the sequence.




# The Viterbi algorithm


$$\text{Recursion: } v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$$



# The Viterbi algorithm

$$\text{Recursion: } v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$$

sequence 

		A	C	G	T	T	A	
 states	S	1	0	0	0	0	0	
	A+	0						
	C+	0						
	G+	0						
	T+	0						
	A-	0						
	C-	0						
	G-	0						
	T-	0						
	E	0						

# The Viterbi algorithm

$$\text{Recursion: } v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$$

sequence →

		A	C	G	T	T	A	
states ↓	S	1	0	0	0	0	0	
	A+	0	$v_{A+}(1)$					
	C+	0	0					
	G+	0	0					
	T+	0	0					
	A-	0	$v_{A-}(1)$					
	C-	0	0					
	G-	0	0					
	T-	0	0					
	E	0	0					

$$v_{A+}(1) = e_{A+}(A) a_{SA+}$$

$$v_{A-}(1) = e_{A-}(A) a_{SA-}$$

$$v_E(1) = \max_l (v_l(1) a_{lE})$$

# The Viterbi algorithm

$$\text{Recursion: } v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$$

		sequence →						
		A	C	G	T	T	A	
states ↓	S	1	0	0	0	0	0	
	A+	0	$v_{A+}(1)$	0				
	C+	0	0	$v_{C+}(2)$				
	G+	0	0	0				
	T+	0	0	0				
	A-	0	$v_{A-}(1)$	0				
	C-	0	0	$v_{C-}(2)$				
	G-	0	0	0				
	T-	0	0	0				
	E	0	0	0				

$$v_{C+}(2) = e_{C+}(C) \max[v_{A+}(1)a_{A+C+}, v_{A-}(1)a_{A-C+}]$$

$$v_{C-}(2) = e_{C-}(C) \max[v_{A+}(1)a_{A+C-}, v_{A-}(1)a_{A-C-}]$$

# The Viterbi algorithm

$$\text{Recursion: } v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$$

sequence →

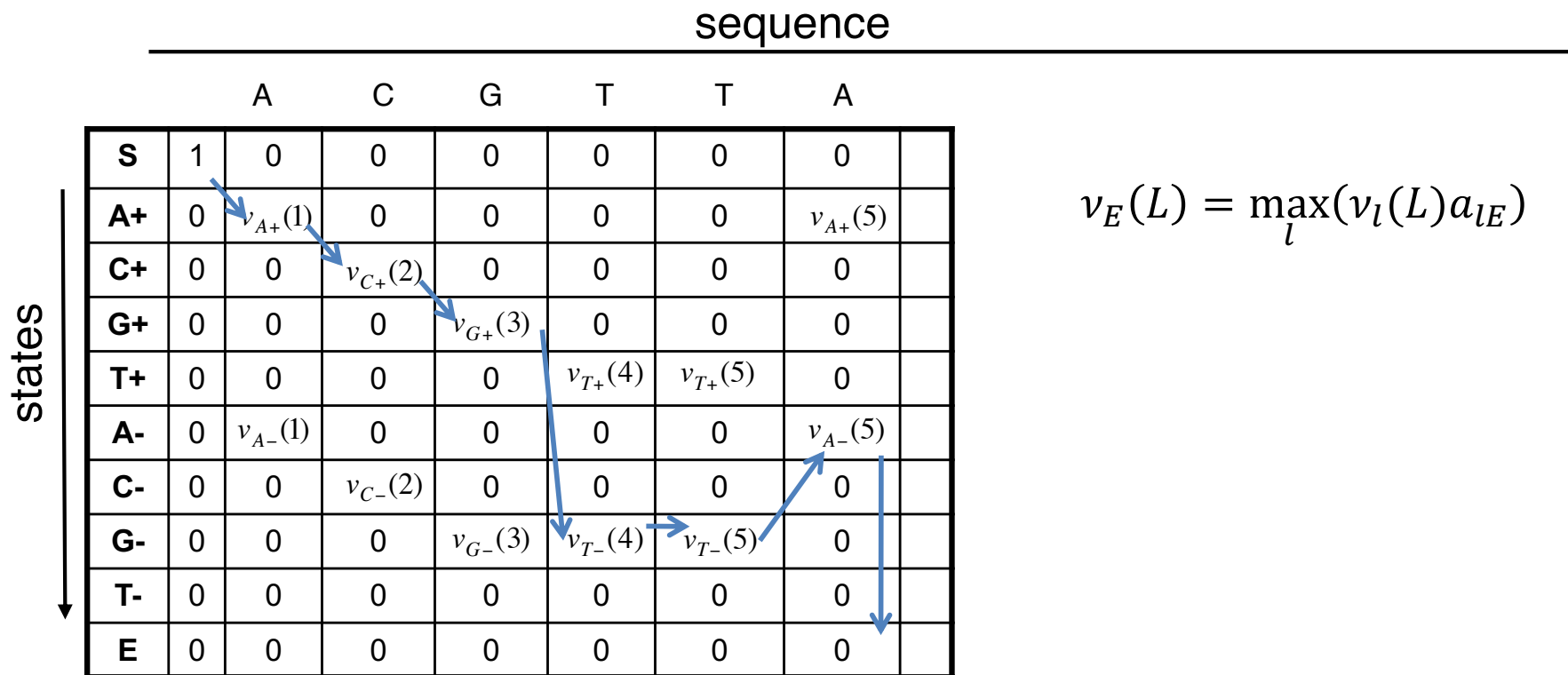
		A	C	G	T	T	A	
states ↓	<b>S</b>	1	0	0	0	0	0	
	<b>A+</b>	0	$v_{A+}(1)$	0	0			
	<b>C+</b>	0	0	$v_{C+}(2)$	0			
	<b>G+</b>	0	0	0	$v_{G+}(3)$			
	<b>T+</b>	0	0	0	0			
	<b>A-</b>	0	$v_{A-}(1)$	0	0			
	<b>C-</b>	0	0	$v_{C-}(2)$	0			
	<b>G-</b>	0	0	0	$v_{G-}(3)$			
	<b>T-</b>	0	0	0	0			
	<b>E</b>	0	0	0	0			

$$v_{G+}(3) = e_{G+}(G) \max[v_{C+}(2) a_{C+G+}, v_{C-}(2) a_{C-G+}]$$

$$v_{G-}(3) = e_{G-}(G) \max[v_{C+}(2) a_{C+G-}, v_{C-}(2) a_{C-G-}]$$

# The Viterbi algorithm

$$\text{Recursion: } v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$$



$$v_E(L) = \max_l (v_l(L) a_{lE})$$

# Viterbi algorithm

Initialization: before reading any symbol we are with certainty in the start state

$$v_S(\varphi) = 1, v_k(\varphi) = 0 \text{ for } \forall k \text{ state other than start}$$

Recursion: at each symbol  $i$  that we read, we have the condition

$$v_k(i) = e_k(x_i) \max_l (v_l(i-1) a_{lk})$$

$$ptr_i(k) = \operatorname{argmax}_l (v_l(i-1) a_{lk})$$

Termination: we reach the end of the sequence and have

$$P(\vec{x}, \pi^*) = \max_k (v_k(L) a_{kE})$$

$$\pi_L^* = \operatorname{argmax}_k (v_k(L) a_{kE})$$

Traceback:  $\pi_{i-1}^* = ptr_i(\pi_i^*) \forall i = L \dots 1$

# Probabilities over paths



We have determined the maximal probability path which has probability  $v_*$ .  
How *unique* is this best path? Are there other paths with very close probabilities?

Examples of other high probability (but suboptimal) paths:

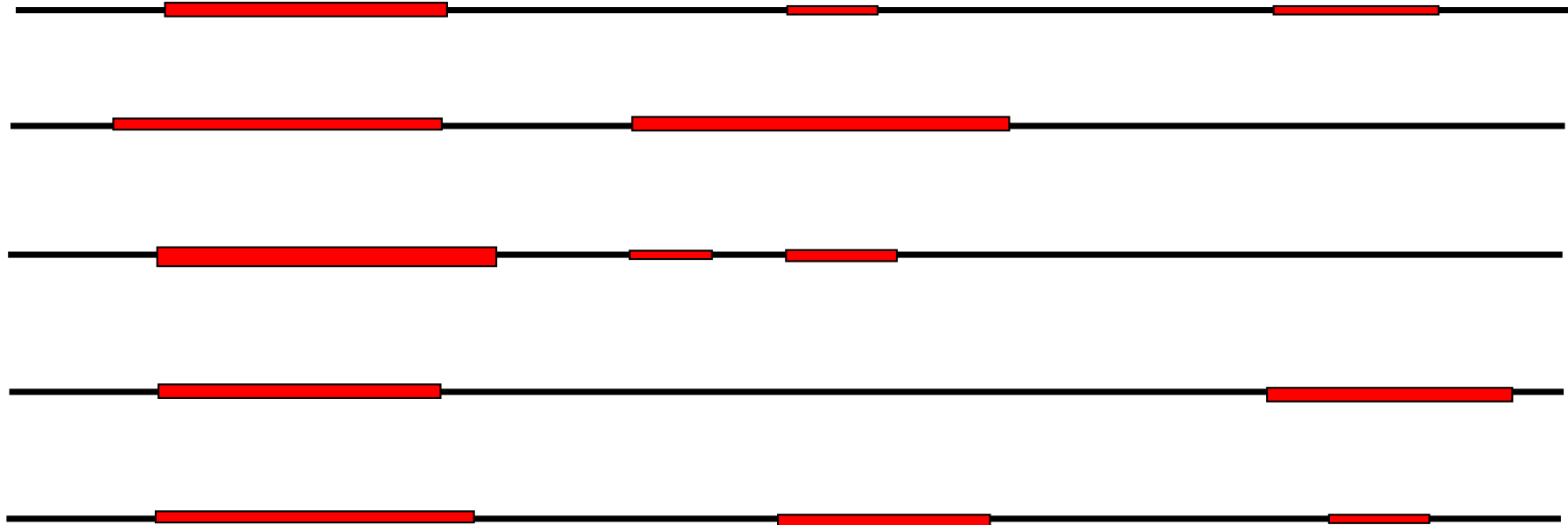


Stable CpG island

Less stable CpG islands



# Probabilities over paths



How can we find identify “stable” CpG islands?

Basic insight: we use posterior probabilities of paths (and parts of paths)

However,  
this means that we cannot limit ourselves to the maximum likelihood path

# Probability of a sequence considering all paths: forward algorithm

**Viterbi** algorithm gives us the most likely path (i.e. sequence of states).

**Forward** algorithm gives us the probability of a sequence taking into account that multiple paths can give rise to the same sequence of symbols:

$$P(\vec{x}) = \sum_{\pi} P(\vec{x}, \pi)$$

This will allow us to compute the probabilities of individual paths as

$$P(\pi|\vec{x}) = \frac{P(\vec{x}, \pi)}{P(\vec{x})} = \frac{P(\vec{x}, \pi)}{\sum_{\pi'} P(\vec{x}, \pi')}$$

Or similarly to compute the probability of having a particular state at a particular position in the sequence.

Algorithm similar to Viterbi, except that instead of calculating the maximum probability to end in state  $k$  with symbol  $x_i$ , we calculate the probability to end in state  $k$  after reading first  $i$  symbols.

$$f_k(i) = P(x_1 \dots x_i, \pi_i = k)$$

# Forward algorithm

Initialization: before reading any symbol we are with certainty in the start state:

$$f_S(\varphi) = 1, f_k(\varphi) = 0 \text{ for } \forall k \text{ state other than start}$$

Recursion: at each symbol  $i$  that we read, we have the relation

$$f_k(i) = e_k(x_i) \sum_l (f_l(i-1) a_{lk})$$

Termination:

$$P(\vec{x}) = \sum_k (f_k(L) a_{kE})$$

# Probability of a sequence considering all paths: backward algorithm

**Backward** algorithm also gives us the probability of a sequence taking into account that multiple paths can give rise to the same sequence of symbols, but doing the computations from the end towards the beginning of the sequence.

Algorithm similar to forward algorithm, except that instead of calculating the probability to end in state  $k$  after reading symbol  $i$ , we calculate the probability of symbols  $x_{i+1} \dots x_L$  starting in state  $k$  at position  $i$ :

$$b_k(i) = P(x_{i+1} \dots x_L, \pi_i = k)$$

# Backward algorithm

Initialization: after the  $L^{th}$  symbol, only the transitions into the end state are possible

$$b_k(L) = a_{kE}, \forall k$$

Recursion: at each position  $i = L - 1, \dots, 1$  we have the condition

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

Termination: at the beginning of the string we have to end in the start state:

$$b_S(\varphi) = P(x_1 \dots x_L | \pi_0 = S) = P(\vec{x}) = \sum_k b_k(1) e_k(x_1) a_{Sk}$$

# Most probable state for an observation $x_i$

Having the forwards and backwards probabilities allows us to compute other quantities of interest.

Is position  $i$  within a CpG island?  $P(\pi_i = \text{CpG} | \vec{x})?$

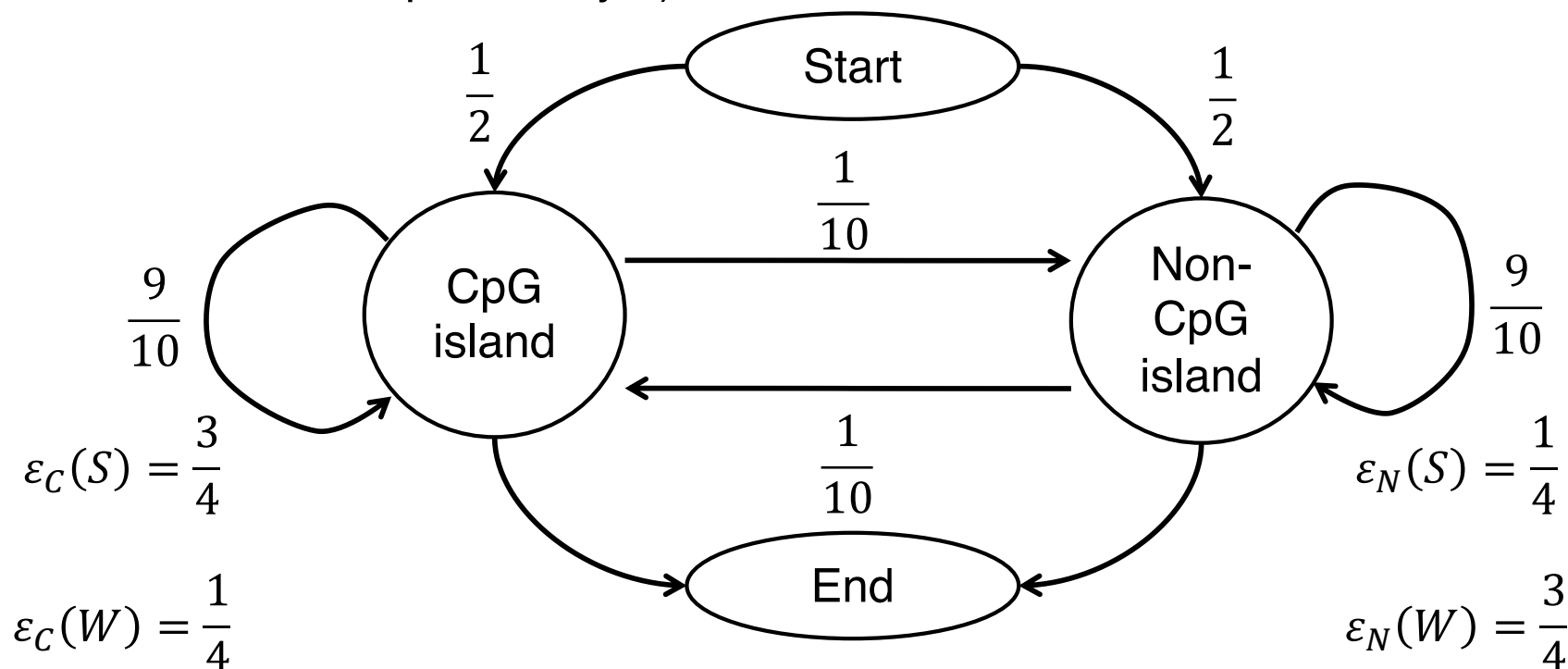
$$P(\vec{x}, \pi_i = k) = P(x_1 \dots x_i, \pi_i = k) P(x_{i+1} \dots x_L, \pi_i = k)$$
$$f_k(i) \qquad b_k(i)$$

By applying the definition of conditional probability we have:

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

# Example

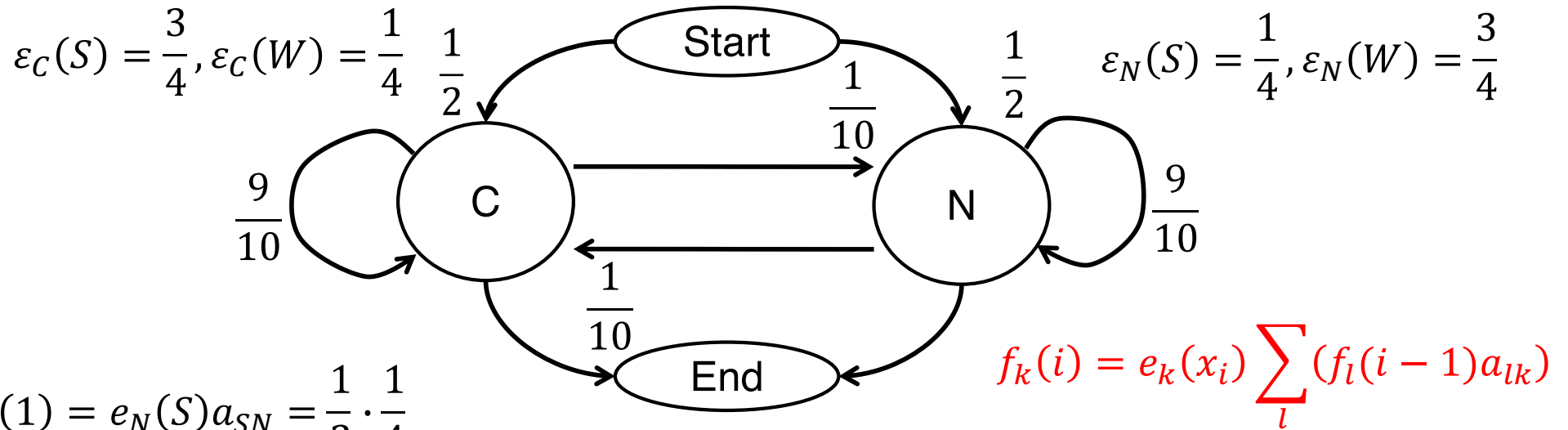
Let's take a simplified example, the sequence SSWS, defined over a reduced alphabet of 2 letters, S and W, which we can think of as the strong (C,G) and weak (A,T) bases. Let's further consider the CpG island model, with the indicated transitions (note: we only consider transitions to the end state at the very last position, and there, the transitions from any state to end state are assumed to have probability 1).



Recall the recurrence relations:

$$f_k(i) = e_k(x_i) \sum_l (f_l(i-1) a_{lk}) \quad b_k(i) = \sum_l b_l(i+1) e_l(x_{i+1}) a_{kl}$$

# Applying the recurrence relations we get



$$f_N(1) = e_N(S) a_{SN} = \frac{1}{2} \cdot \frac{1}{4}$$

$$f_C(1) = e_C(S) a_{SC} = \frac{1}{2} \cdot \frac{3}{4}$$

$$f_N(2) = e_N(S) [f_N(1) a_{NN} + f_C(1) a_{CN}] = \frac{1}{4} \left[ \frac{1}{8} \cdot \frac{9}{10} + \frac{3}{8} \cdot \frac{1}{10} \right] = \frac{3}{80}$$

$$f_C(2) = e_C(S) [f_N(1) a_{NC} + f_C(1) a_{CC}] = \frac{3}{4} \left[ \frac{1}{8} \cdot \frac{1}{10} + \frac{3}{8} \cdot \frac{9}{10} \right] = \frac{21}{80}$$

f		S	S	W	S	
Position	0	1	2	3	4	
Start	1	0	0	0	0	
N	0	1/8	3/80			
C	0	3/8	21/80			
End	0					



# Applying the recurrence relations we get

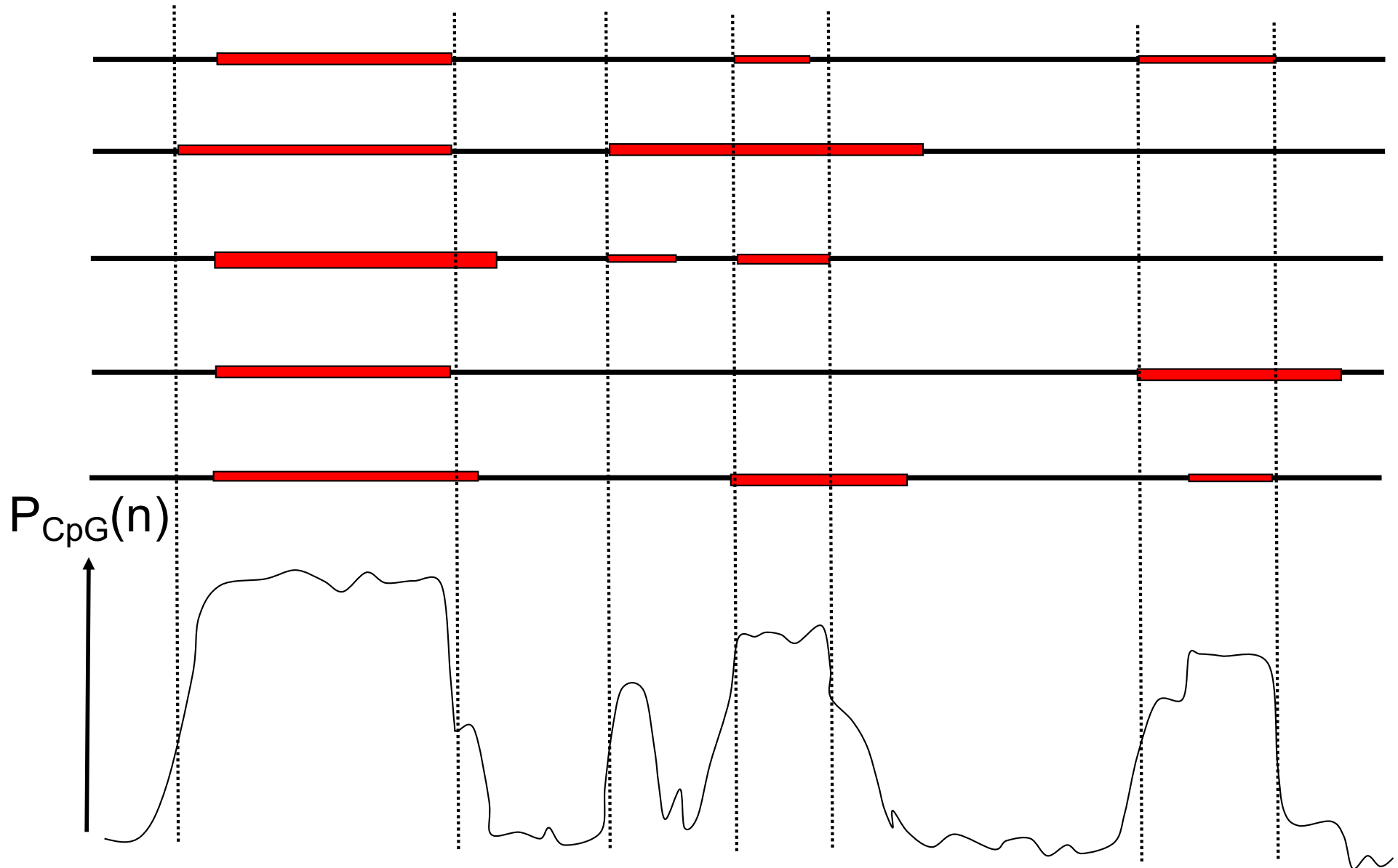
<b>f</b>		<b>S</b>	<b>S</b>	<b>W</b>	<b>S</b>	
Position	0	1	2	3	4	
Start	1	0	0	0	0	
N	0	1/8	3/80	9/200	93/8000	
C	0	3/8	21/80	12/200	351/8000	
End	0					444/8000

<b>b</b>		<b>S</b>	<b>S</b>	<b>W</b>	<b>S</b>	
Position						
Start	444/8000					
N		63/1000	22/100	3/10	1	
C		127/1000	18/100	7/10	1	
End						

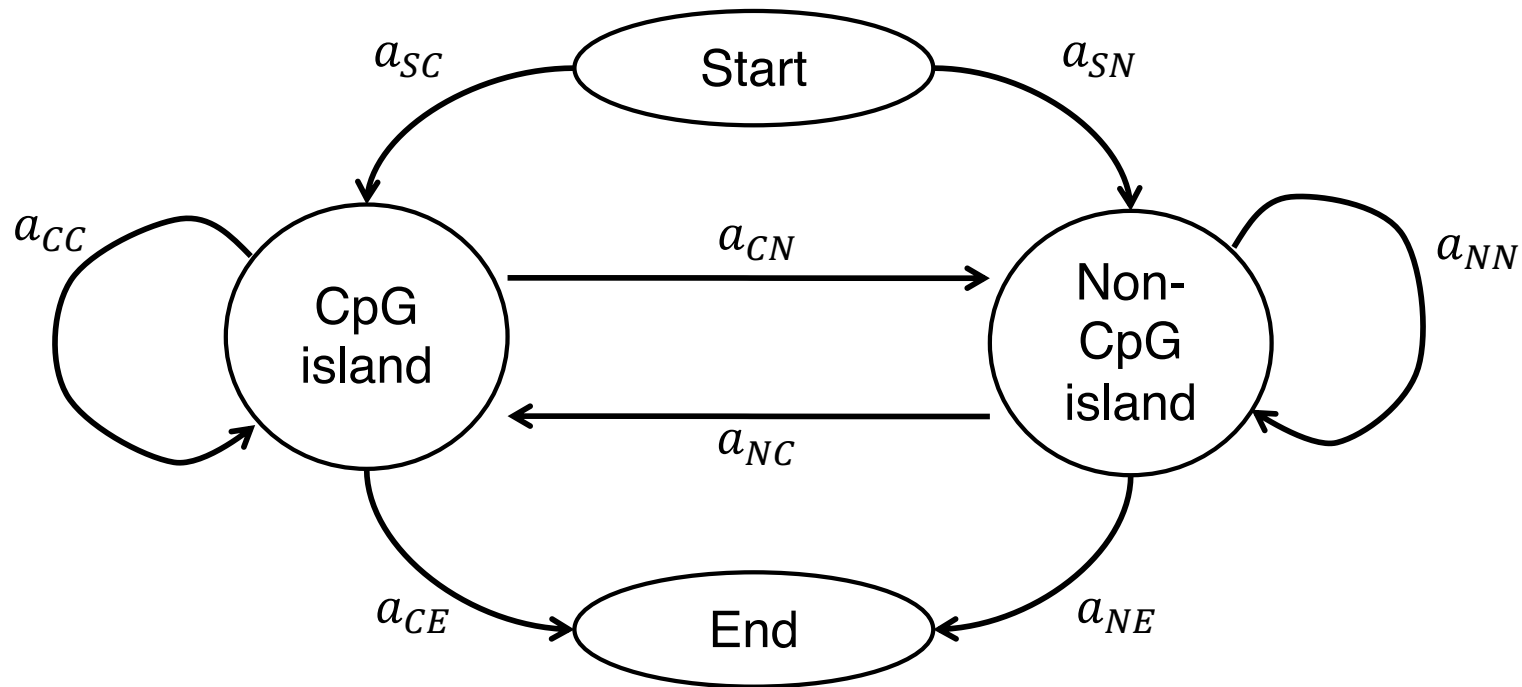
So, at position 3 in the sequence we have:

$$P_C(3) = \frac{f_C(3)b_C(3)}{P(\vec{x})} = \frac{\frac{12}{200} \frac{7}{10}}{\frac{444}{8000}} = \frac{84}{111} \quad P_N(3) = \frac{f_N(3)b_N(3)}{P(\vec{x})} = \frac{\frac{9}{200} \frac{3}{10}}{\frac{444}{8000}} = \frac{27}{111}$$

# Probabilities over paths



# Distribution of state lengths



What is the average length of a CpG island generated by this model?

Can be estimated considering that the probability that the length is  $l$  is given by

$$P(l) = a_{CC}^l (1 - a_{CC})$$

# Distribution of state lengths

Probability that the length is  $l$  is given by  $P(l) = a_{CC}^l (1 - a_{CC})$

$$\langle l \rangle = \sum_{l=0}^{\infty} l P(l) = \sum_{l=0}^{\infty} l a_{CC}^l (1 - a_{CC})$$

$$= (1 - a_{CC}) \sum_{l=0}^{\infty} l a_{CC}^l = a_{CC} (1 - a_{CC}) \sum_{l=0}^{\infty} l a_{CC}^{l-1}$$

$$= a_{CC} (1 - a_{CC}) \frac{\partial}{\partial a_{CC}} \left[ \sum_{l=0}^{\infty} a_{CC}^l \right] = a_{CC} (1 - a_{CC}) \frac{\partial}{\partial a_{CC}} \left[ \frac{1}{1 - a_{CC}} \right]$$

$$= a_{CC} (1 - a_{CC}) \frac{1}{(1 - a_{CC})^2} = \frac{a_{CC}}{1 - a_{CC}}$$

Therefore,  $\langle l \rangle = \frac{a_{CC}}{1 - a_{CC}}$

Note: so far we assumed that we know the model parameters.

In principle, if we have examples of sequences annotated with CpG island and non-CpG island regions, we can infer these parameters from a maximum likelihood argument.

What if we don't have this annotation, but only a set of sequences?

# Parameter estimation for HMMs

---

When the state sequence is **known**, we simply need to compute the number of transitions and emissions to calculate

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}} \text{ and } e_k(\beta) = \frac{E_k(\beta)}{\sum_{\beta'} E_k(\beta')}$$

# Parameter estimation for HMMs

When the state sequence is **not known**, one has to use an optimization procedure

Expectation - Maximization

The Baum-Welch algorithm:

Based on the current values of  $a_{kl}$  and  $e_k(\beta)$ , compute expected values of  $A_{kl}$  and  $E_k(\beta)$ , considering probable paths. Then we re-estimate  $a_{kl}$  and  $e_k(\beta)$ .

Posterior probability that transition  $a_{kl}$  is used at position  $i$  in sequence  $\vec{x}$

$$P(\pi_i = k, \pi_{i+1} = l | \vec{x}, \theta) = \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{P(\vec{x})}$$

By summing over all training sequences  $j$  and all positions  $i$ , we obtain

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

Similarly, the expected number of times symbol  $\beta$  appears in state  $k$  is

$$E_k(\beta) = \sum_j \frac{1}{P(\vec{x}^j)} \sum_{\{i | x_i^j = \beta\}} f_k^j(i) b_k^j(i)$$

And we recalculate  $a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}}$  and  $e_k(\beta) = \frac{E_k(\beta)}{\sum_{\beta'} E_k(\beta')}$

# HMM applications in computational biology

- Pairwise alignment of sequences
- Multiple alignment of sequences
- Finding genes in genomes
- Representing transcription factor binding sites and finding them in genomic sequences
- Representing, recognizing and finding signals in protein sequences (protein family members, domains, functional elements)
- RNA-folding and prediction of RNA genes

## HMM applications in other fields

LOTS!

Parsing sentences

Modeling and recognition of music fragments

Speech recognition

ETC.