Markov Chains and Hidden Markov Models

Some slides are from Ben Langmead of the Johns Hopkins University.

Problem statement

- Different types of data (not record or transaction types);
- A different type of "classification" problem

Types of data from lecture 2

- Record
  - Data Matrix
  - Document Data
  - Transaction Data
- Graph
  - World Wide Web
  - Molecular Structures
- Ordered
  - Spatial Data
  - Temporal Data
  - Sequential Data
  - **Genetic Sequence Data**

Goal: identify a region with lots of C followed by G

CpG islands

- Definition: CpG is the pair of nucleotides C and G appearing successively, in this order, along one DNA strand. (e.g. CGCGCGGCCGGCGCG) But it can have "CC", "GG" or even A and T in the islands.

- CpG relatively rare in most DNA sequences. However, in particular sub-sequences, which are a few hundred to a few thousand bees long, CpG is more frequent. Those sequences are called CpG islands.

- **Problem one:** given a short DNA sequence x, decide whether x is from CpG island.

- **Problem two**: given a genome, search for CpG islands in the genome.

Which one is harder ?

Classification problem

Given a short DNA sequence x, decide whether x is from CpG island.

S: AAGCCGGGAAGTTGTATG

Questions:

Is S from inside/outside CpG islands?

Training data: a bunch of sequences from inside and outside of CpG islands

CpG island	Not CpG island
CGCGCGGCCGCGGCGA	CTACCGATCTCGCAAA
CCGGCGCGCGCGTGCC	CGTAACATGACGATTGC
CCGGCGACCCGGGCGCG	CCGTAATCCTTACTAG
……	……

Decision tree? Naïve Bayes? kNN?

Bayes Belief Network?

Classification problem

Given a short DNA sequence x, decide whether x is from CpG island.

S: AAGCCGGGAAGTTGTATG

Questions:

Is S from inside/outside CpG islands?

Compare P(inside|S) and P(outside|S)

Training data: a bunch of sequences from inside and outside of CpG islands

inside

CGCGCGGCCGCGGCGA
CCGGCGCGCGCGTGCC
CCGGCGACCCGGGCGCG
……

outside

CTACCGATCTCGCAAA
CGTAACATGACGATTGC
CCGTAATCCTTACTAG
……

How to compute the posterior probability

S2: CCGCGCGC

P(inside|S2) = P(in |CCGCGCGC)

=

P(in) is the prior probability reflecting the fraction of CpG islands. Usually P(in) < P(out). P(in): the probability that the input sequence is inside a CpG island. P(out): the probability that the input sequence is outside a CpG island.

To focus on the interpretation of Markov chain model, we will assume that P(in) is equal to P(out). Thus

P(inside|S2) and P(outside|S2)' ranking is determined by P(S2|in) and P(S2|out)

Q: how to compute from the training data?

Sequence models

Let P(*x*) be the probability of sequence *x* as assigned by the model

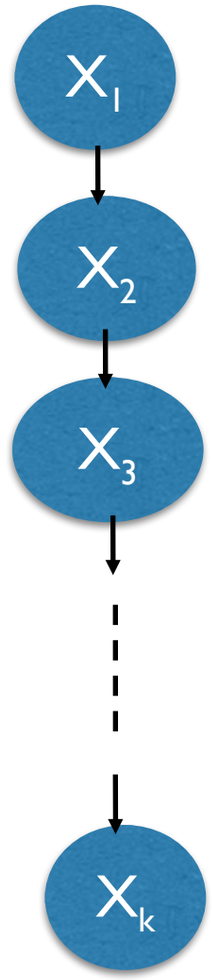
$$P(x) = P(\underbrace{x_k, x_{k-1}, \ldots x_1}) \quad \text{Joint probability of all sequence items appearing as they do (order matters)}$$

To estimate P(*x*), count # times *x* appears in the training set labeled *inside* divided by *total* # times *x* appears in training set

But for sufficiently long *k*, we might not see *any* occurrences of *x*, or very few. Joint probabilities for rare events are hard to estimate well.

Sequence models

Bayes Belief network for 1st-order Markov model



$$P(x) = P(x_k, x_{k-1}, \ldots x_1)$$

Re-write with conditional probability:

$$= P(x_k \mid x_{k-1}, \ldots x_1) P(x_{k-1}, \ldots x_1)$$

$$= P(x_k \mid x_{k-1}, \ldots x_1) P(x_{k-1} \mid x_{k-2}, \ldots x_1) P(x_{k-2}, \ldots x_1)$$

(etc)

Add a simplifying assumption: to know the probability of having a particular item x_k, *we only have to know the previous item*: x_{k-1}

Formally: random variable x_k is *conditionally independent* of $x_1 \ldots x_{k-2}$ given x_{k-1}

Informally: "the future is independent of the past given the present"

Sequence models

A simplifying assumption: to know the probability of having a particular item x_k, *we only have to know the previous item*: x_{k-1}

$$P(x) = P(x_k, x_{k-1}, \ldots x_1)$$

$$= P(x_k \mid x_{k-1}, \ldots x_1)\, P(x_{k-1}, \ldots x_1)$$

$$= P(x_k \mid x_{k-1}, \ldots x_1)\, P(x_{k-1} \mid x_{k-2}, \ldots x_1)\, P(x_{k-2}, \ldots x_1)$$

(etc)

drop drop (bunch more drops once this is expanded)

$$\approx P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)\, P(x_1)$$

Markov property / Markov assumption

It's a big assumption, but it's often reasonable and it makes the model much easier to work with

Markov chain

Assigning a probability to a sequence using Markov property:

$$P(x) \underset{\text{Markov property}}{\approx} P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)\, P(x_1)$$

Say x is a nucleotide k-mer

$P(x_i \mid x_{i-1})$ probability of seeing nucleotide x_i in i^{th} position given that previous nucleotide is x_{i-1}

Shorthand: P(G I C) = probability of G given previous is C

Markov chain

Say someone gives us the sequences of several CpG islands.
How do we estimate, say, P(G I C)?

P(G I C) = # times CG occurs / # times C occurs

Markov chain

Given CpG island sequences from human chromosome 1, count nucleotide and dinucleotide occurrences and estimate all 16 possible P(x_i | x_{i-1}):

P(A | A) = # times AA occurs / # times A occurs

P(C | A) = # times AC occurs / # times A occurs

P(G | A) = # times AG occurs / # times A occurs

P(T | A) = # times AT occurs / # times A occurs

P(A | C) = # times CA occurs / # times C occurs

(etc)

Markov chain (1st order)

Given CpG island sequences from human chromosome 1, count nucleotide and dinucleotide occurrences and estimate all 16 possible P(x_i | x_{i-1}):

```
>>> iTab, nTab = islandTransitionTables(fn, ifn)
>>> print iTab
[[ 0.18544138  0.27640458  0.40091352  0.13724053]
 [ 0.18958227  0.35905063  0.25324026  0.19812684]
 [ 0.17268916  0.33011349  0.35610656  0.14109079]
 [ 0.09410222  0.34163592  0.37686698  0.18739488]]
```


X_{i-1}: A C G T

X_i: A C G T

P(T | G)

Rows sum to 1

Markov chain

We can do the same for dinucleotides *outside* of CpG islands

Inside

	A	C	G	T
A	0.18544138	0.27640458	0.40091352	0.13724053
C	0.18958227	0.35905063	0.25324026	0.19812684
G	0.17268916	0.33011349	0.35610656	0.14109079
T	0.09410222	0.34163592	0.37686698	0.18739488

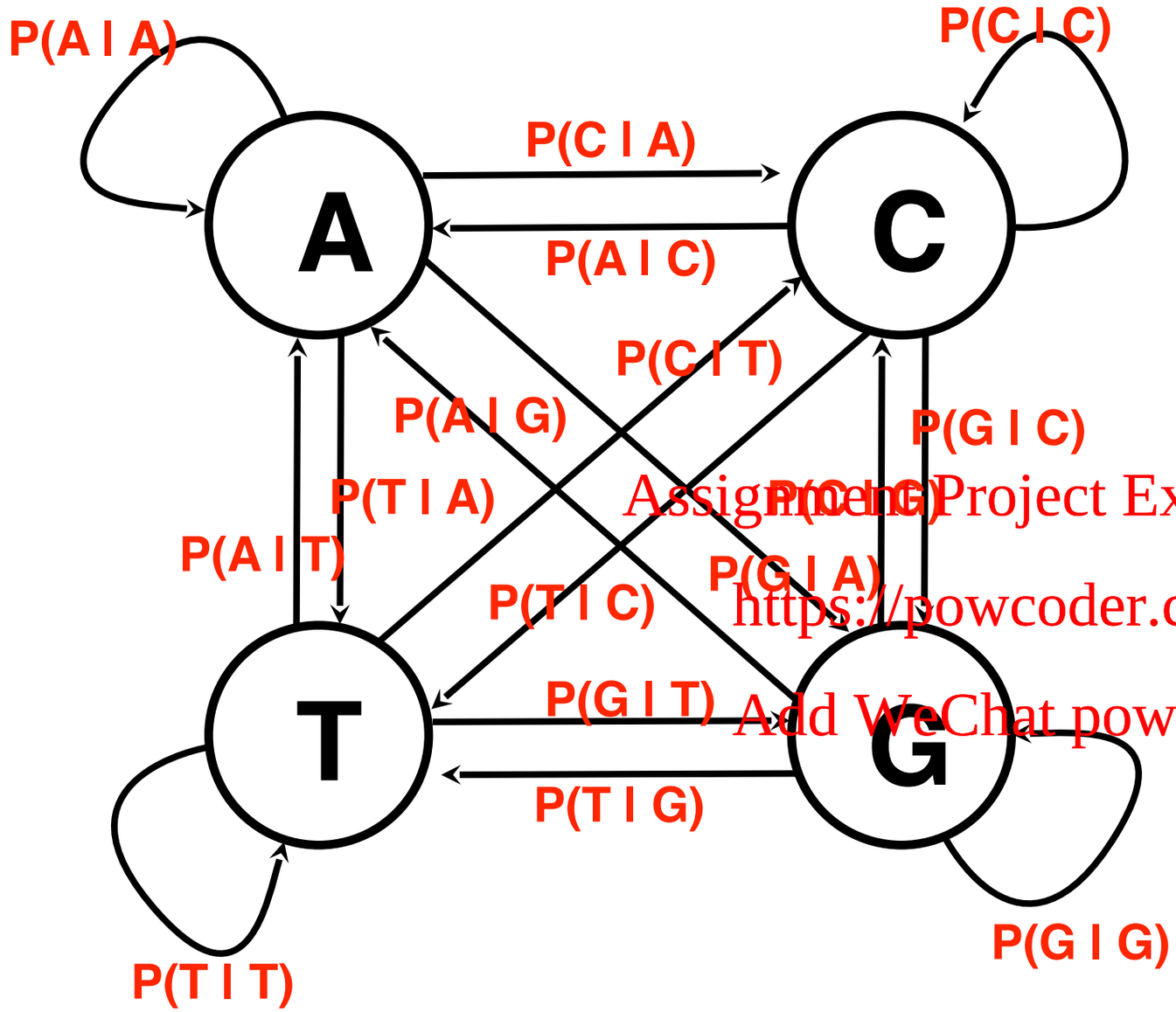
Outside

	A	C	G	T
A	0.2948135	0.19467897	0.28696205	0.22354548
C	0.32681187	0.29415529	0.06172587	0.31730697
G	0.25713351	0.23354071	0.29423494	0.21509084
T	0.17956538	0.23250026	0.29462341	0.29331096

Notice anything interesting about the outside conditional probabilities?

P(G | C) is low, matching our expectation that there are few CpGs outside islands

Markov chain



Markov chain is a probabilistic automaton

Each edge has a *transition probability*: probability that edge's destination is the next node visited after edge's source

Here, nodes labels are symbols and transition labels are conditional probabilities

P(A I A) = # times AA occurs / # times A occurs

P(C I A) = # times AC occurs / # times A occurs

P(G I A) = # times AG occurs / # times A occurs

Markov chain

Recall how we assign a probability to a single string

$$P(x) \underset{\text{Markov property}}{\approx} P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)\, P(x_1)$$

For simplicity, drop $P(x_1)$

$$P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)\, \cancel{P(x_1)}$$

$$P(x) \approx P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)$$

$P(x)$ now equals product of all the Markov chain edge weights on our string-driven walk through the chain

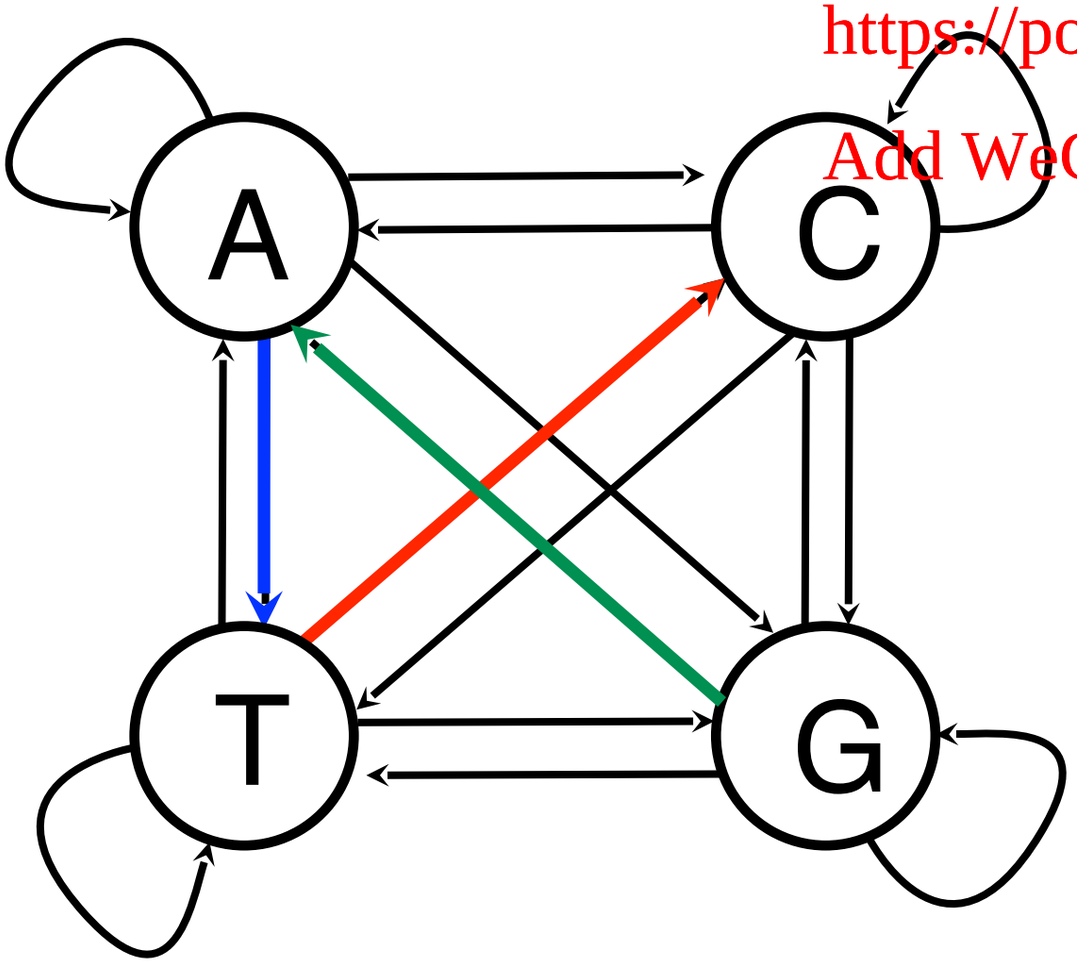
Markov chain

```
>>> iTab, nTab = islandTransitionTables(fn, ifn)
>>> print iTab
[[ 0.18544138  0.27640458  0.40091352  0.13724053]
 [ 0.18958227  0.35905063  0.25324026  0.19812684]
 [ 0.17268916  0.33011349  0.35610656  0.14109079]
 [ 0.09410222  0.34163592  0.37686698  0.18739488]]
```

X_{i-1} A C G T

A C G T

X_i



x = GATC

$P(x) = P(x_4 \mid x_3)\, P(x_3 \mid x_2)\, P(x_2 \mid x_1)$

$P(x) = P(C \mid T)\, P(T \mid A)\, P(A \mid G)$

$= 0.34163592 * 0.13724053 * 0.17268916$

$= 0.00809675$

Markov chain

To avoid repeated multiplies yielding small numbers, we switch to log domain

$$\log P(x) \approx \log [\, P(x_k \mid x_{k-1})\, P(x_{k-1} \mid x_{k-2}) \ldots P(x_2 \mid x_1)\,]$$

$$= \log P(x_k \mid x_{k-1}) + \log P(x_{k-1} \mid x_{k-2}) + \ldots + \log P(x_2 \mid x_1)$$

$$= \sum_{i=2}^{k} \log P(x_i \mid x_{i-1})$$

Switching to log domain, multiplies become adds

I'll use base-2 logs

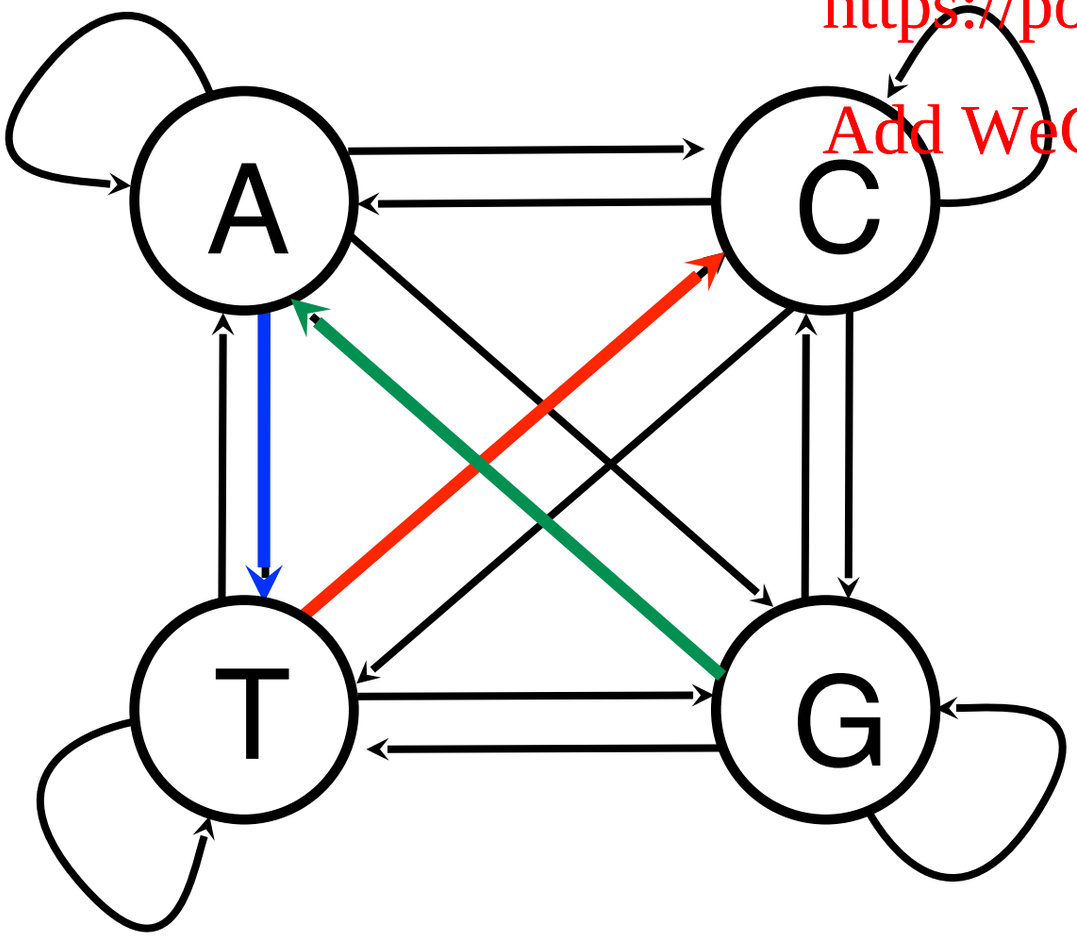
Markov chain

```
>>> iTab, nTab = islandTransitionTables(fn, ifn)
>>> print numpy.log2(iTab)
[[-2.43096492 -1.85514658 -1.31863704 -2.86522151]
 [-2.39910406 -1.4777408  -1.98142131 -2.33550376]
 [-2.53375061 -1.59896599 -1.48961909 -2.82530423]
 [-3.40962748 -1.54946844 -1.40787269 -2.41584653]]
```

X_{i-1}: A C G T

A C G T

X_i



x = GATC

$$\log P(x) = \sum_{i=2}^{4} \log P(x_i / x_{i-1})$$

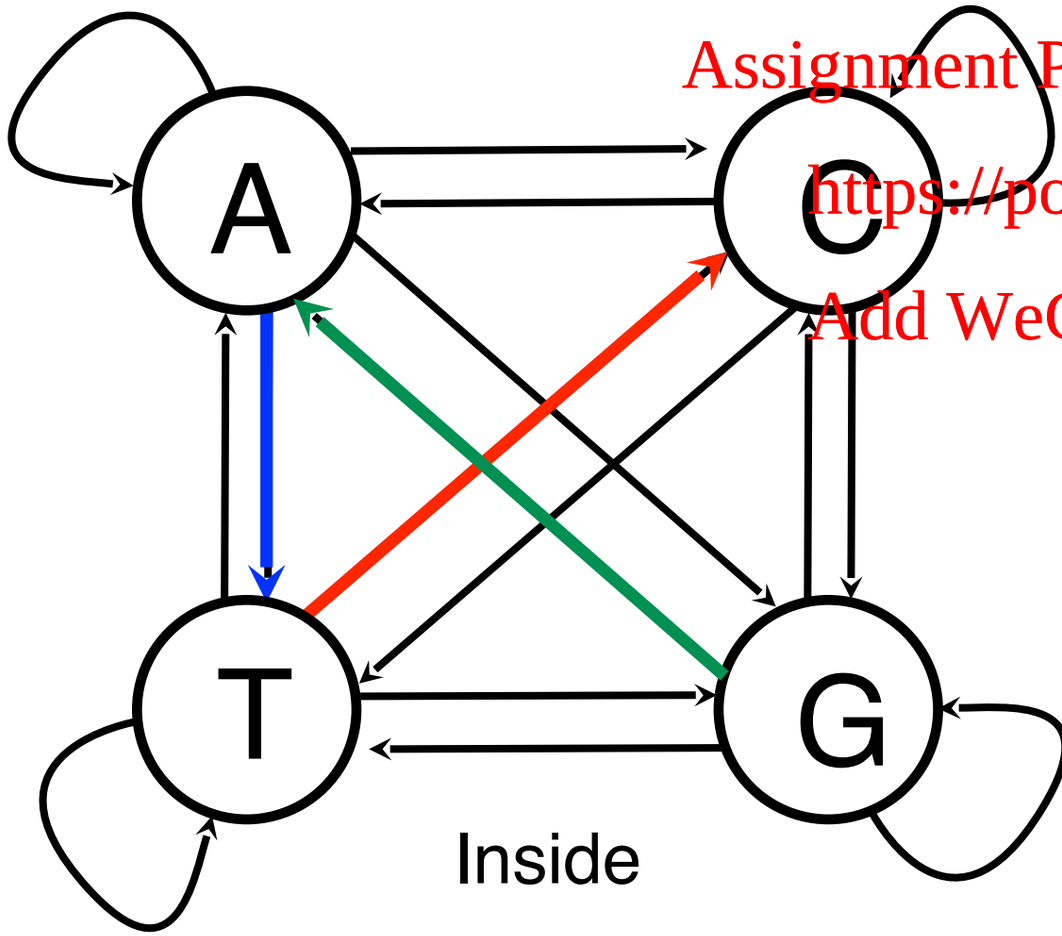
$$= -1.54946844 + -2.86522151 + -2.53375061$$

$$= -7.30174249$$

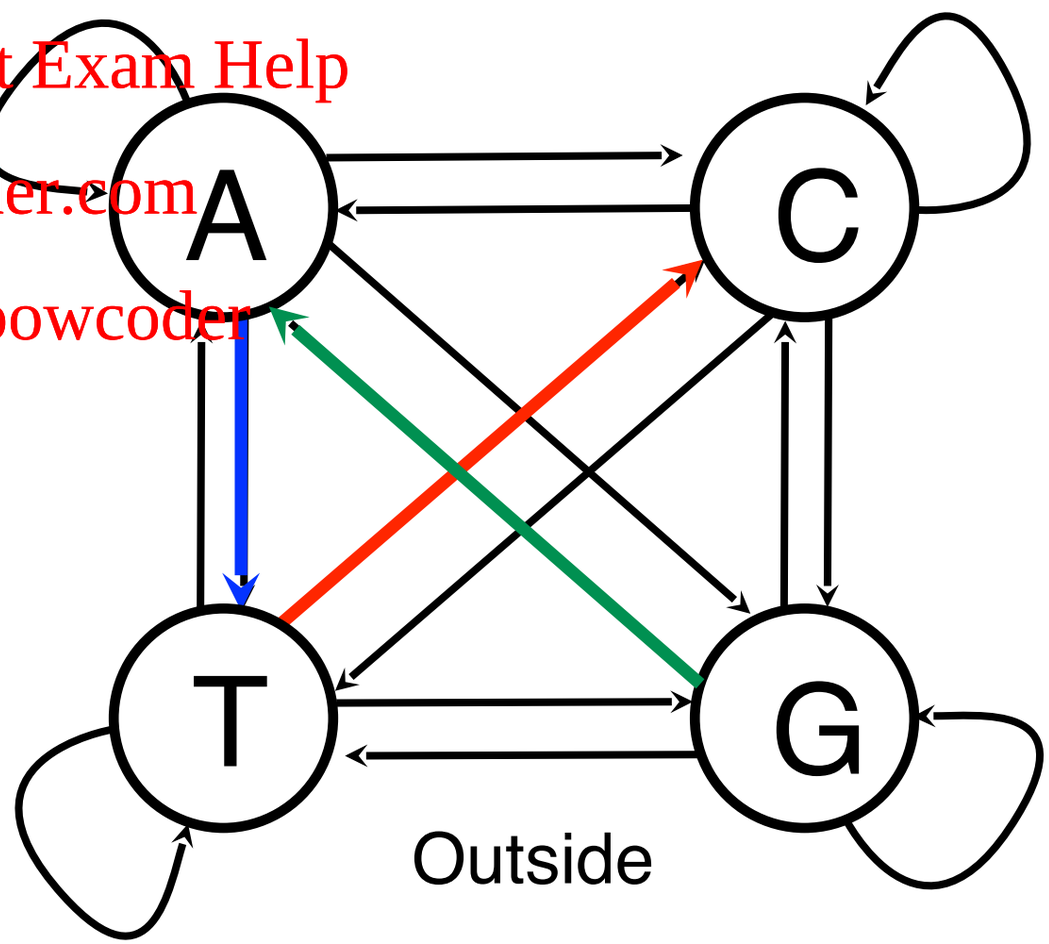
Markov chain

P(x) given the inside-CpG model is helpful, but we really want to know which model is better, inside CpG or outside CpG?

Use *ratio:* $\frac{\text{P(} x \text{) from inside model}}{\text{P(} x \text{) from outside model}}$



Inside



Outside

Markov chain

Taking log, we get *log ratio*:

$$S(x) = \log \frac{P(x) \text{ inside CpG}}{P(x) \text{ outside CpG}}$$

If inside more probable than outside, fraction is > 1 and log ratio is > 0. Otherwise, fraction is ≤ 1 and log ratio is ≤ 0.

$$S(x) = \log \frac{P(x) \text{ inside CpG}}{P(x) \text{ outside CpG}}$$

$$= \log [\, P(x) \text{ inside CpG} \,] - \log [\, P(x) \text{ outside CpG} \,]$$

$$= \sum_{i=2}^{k} \left(\log [\, P(x_i \mid x_{i-1}) \text{ inside CpG} \,] \right) - \sum_{i=2}^{k} \log \left([\, P(x_i \mid x_{i-1}) \text{ outside CpG} \,] \right)$$

$$= \sum_{i=2}^{k} \left(\log [\, P(x_i \mid x_{i-1}) \text{ inside CpG} \,] - \log [\, P(x_i \mid x_{i-1}) \text{ outside CpG} \,] \right)$$

New table: take elementwise log of the inside/outside tables, subtract outside from inside

Markov chain

```
>>> iTab, nTab = islandTransitionTables(fn, ifn)
>>> print iTab
[[ 0.20328697  0.26144423  0.40629367  0.12897512]
 [ 0.18175425  0.35880255  0.24915835  0.21028485]
 [ 0.17900663  0.32594344  0.35910409  0.13594584]
 [ 0.09718687  0.34541934  0.35518406  0.20220973]]
>>> print nTab
[[ 0.32756059  0.17183665  0.24355314  0.25704963]
 [ 0.35218354  0.25880566  0.04404104  0.34496977]
 [ 0.28883529  0.20906356  0.25862313  0.24347803]
 [ 0.21890134  0.20417181  0.24903103  0.32789582]]
>>> lrTab = numpy.log2(iTab) - numpy.log2(nTab)
>>> print lrTab
[[-0.68824404  0.6054655   0.73828635 -0.99495413]
 [-0.95433841  0.471321    2.5001426  -0.71412499]
 [-0.69023394  0.64068002  0.47355078 -0.84075959]
 [-1.17144749  0.75856518  0.51224132 -0.69738508]]
```

Inside ↑ / ↓: A C G T

Outside ↑ / ↓: A C G T

Log ratio ↑ / ↓: A C G T

A C G T

Markov chain

Now, given a string x, we can easiy assign it a log ratio "score" $S(x)$:

$$S(x) = \log \frac{\text{P}(x) \text{ inside CpG}}{\text{P}(x) \text{ outside CpG}}$$

$$\approx \sum_{i=2}^{k} \left(\log \left[\text{P}(x_i \mid x_{i-1}) \text{ inside CpG} \right] - \log \left[\text{P}(x_i \mid x_{i-1}) \text{ outside CpG} \right] \right)$$

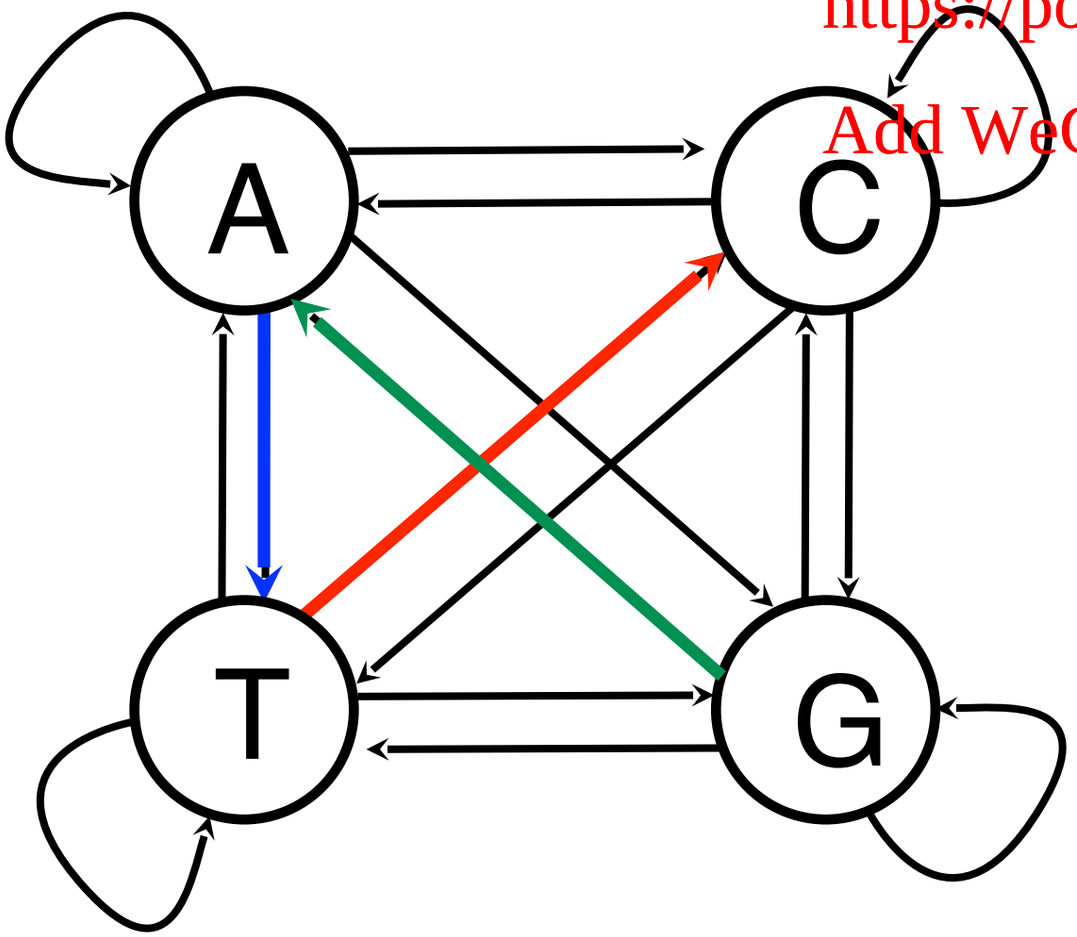
Markov chain

```
>>> iTab, nTab = islandTransitionTables(fn, ifn)
>>> lrTab = numpy.log2(iTab) - numpy.log2(nTab)
>>> print lrTab
[[-0.66883939  0.50568449  0.48243108 -0.70386181]
 [-0.78563635  0.28760934  2.03655959 -0.67945489]
 [-0.57434013  0.49928806  0.27534041 -0.60832223]
 [-0.9322086   0.55522735  0.35518335 -0.6463494 ]]
```

X_{i-1} A C G T

A C G T

X_i



x = GATC

S(*x*) = 0.55522735 + -0.70386181 + -0.57434013

= -0.72297459

Negative, so probability with *outside* model is greater

Markov chain

$$S(x) = \log \frac{P(x) \text{ inside CpG}}{P(x) \text{ outside CpG}}$$

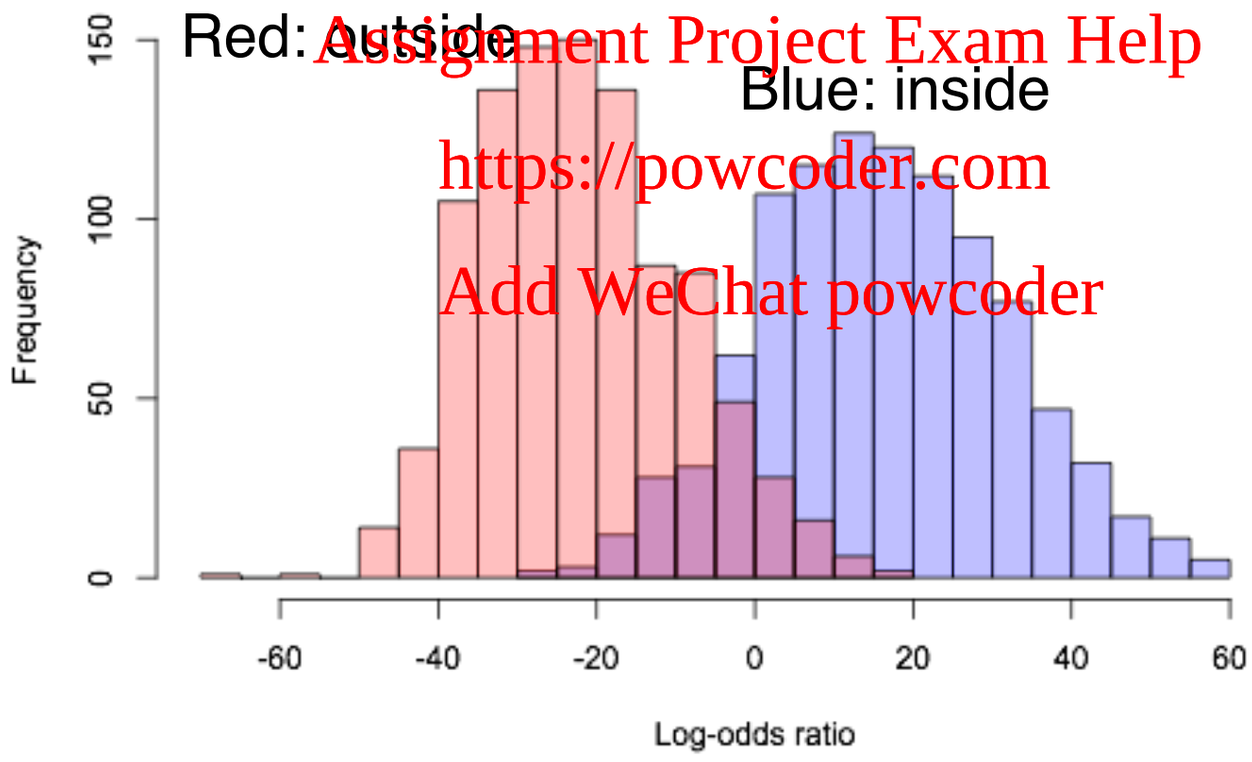
S(CGCGCGCGCGCGCGCGCGCGCGCG) =2.246609048

S(ATTCTACTATCATCTATCTATCTTCT) =9.501209765

Markov chain: experiment

Drew 1,000 100-mers from inside CpG islands on chromosome 1, and another 1,000 from outside, and calculated log ratios for all

Trained markov chain on dinucleotides from chromosome 22



Markov chain

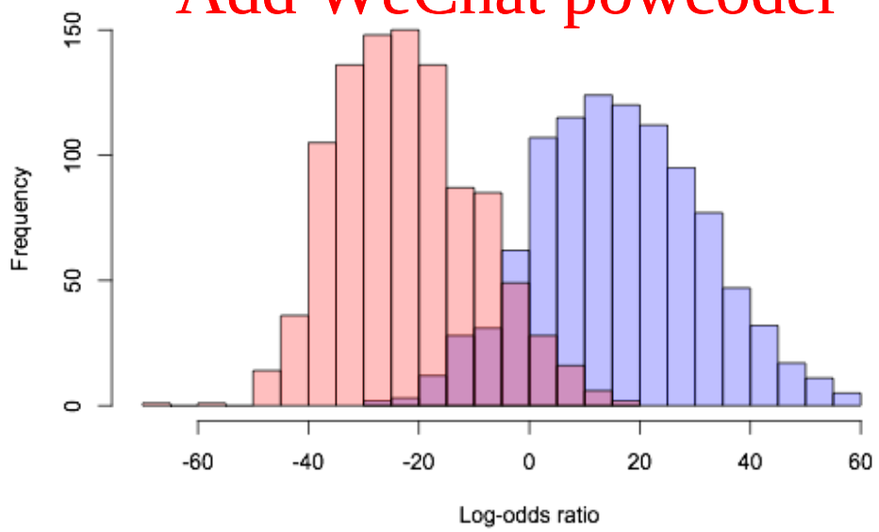
Markov property made our problem very tractable

P(x_i | x_{i-1})s estimated in single, simple pass through training data

Transition probability tables have | ∑ |^2 cells; fine for DNA & protein

Calculating S(*x*) is O(| *x* |); just lookups and additions

... and discriminates well between inside & outside examples in CpG island example



Higher order Markov model

$$P(x) \approx P(x_k \mid x_{k-1}\, x_{k-2})\, P(x_{k-1} \mid x_{k-2}\, x_{k-3}) \ldots P(x_3 \mid x_2\, x_1)\, P(x_2\, x_1)$$

2nd order

$$P(x) \approx P(x_k \mid x_{k-1}\, x_{k-2}\, x_{k-3})\, P(x_{k-1} \mid x_{k-2}\, x_{k-3}\, x_{k-4}) \ldots P(x_3 x_2 x_1)$$

3rd order