Biological Sequence Analysis
Spring 2018

VELI MÄKINEN

TEXTBOOK: WWW.GENOME-SCALE.INFO
Part II

ALIGNMENT SCORES
## Sequence alignment: estimating homologs by sequence similarity

- **Alignment specifies which positions in two sequences match**

<table>
<thead>
<tr>
<th></th>
<th>acgtctag</th>
<th>acgtctag</th>
<th>acgtctag</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>actctag-</td>
<td>-actctag</td>
<td>ac-tctag</td>
</tr>
</tbody>
</table>

- 2 matches
- 5 mismatches
- 1 not aligned

- 5 matches
- 2 mismatches
- 1 not aligned

- 7 matches
- 0 mismatches
- 1 not aligned
Mutations: Insertions, deletions and substitutions

- **Indel**: insertion or deletion of a base with respect to the ancestor sequence

\[
\begin{array}{c}
\text{acgtcttag} \\
\text{actcttag}
\end{array}
\]

- **Mismatch**: substitution (point mutation) of a single base

\[
\begin{array}{c}
\text{actcttag}
\end{array}
\]

• Insertions and/or deletions are called *indels*
Protein sequence alignment

- Homologs can be easier identified with alignment of protein sequences:
  - *Synonymous (silent) mutations* that do not change the amino acid coding are frequent
    - Every third nucleotide can be mismatch in an alignment where amino acids match perfectly
  - Frameshifts, introns, etc. should be taken into account when aligning protein coding DNA sequences
Example

- Consider RNA sequence alignment:
  AUGAUUACUCAUAGA... AUGAUCACCACCGG...
- Versus protein sequence alignment:
  MITHR... MITHR...
Scoring amino acid alignments

- Substitutions between chemically similar amino acids are more frequent than between dissimilar amino acids
- We can check our scoring model against this

Score matrices

Let \( A = a_1a_2...a_n \) and \( B = b_1b_2...b_n \) be sequences of equal length (no gaps allowed to simplify things).

To obtain a score for alignment of \( A \) and \( B \), where \( a_i \) is aligned against \( b_i \), we take the ratio of two probabilities:

1. The probability of having \( A \) and \( B \) where the characters match (match model \( M \))
2. The probability that \( A \) and \( B \) were chosen randomly (random model \( R \))
Score matrices: random model

• Under the random model, the probability of having A and B is

\[ P(A, B|R) = \prod_i q_{ai} \prod_i q_{bi} \]

where \( q_{xi} \) is the probability of occurrence of amino acid type \( x_i \)

• Position where an amino acid occurs does not affect its type
Score matrices: match model

- Let $p_{ab}$ be the probability of having amino acids of type $a$ and $b$ aligned against each other given they have evolved from the same ancestor $c$
- The probability is

$$P(A, B|M) = \prod_i p_{a_i b_i}$$
Score matrices: log-odds ratio score

- We obtain the score $S$ by taking the ratio of these two probabilities

$$
\frac{P(A,B|M)}{P(A,B|R)} = \frac{\prod_i p_{a_i b_i}}{\prod_i q_{a_i} \prod_i q_{b_i}} = \prod_i \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}}
$$

and taking a logarithm of the ratio

$$
S = \log_2 \frac{P(A,B|M)}{P(A,B|R)} = \sum_{i=1}^n \log_2 \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}} = \sum_{i=1}^n s(a_i, b_i)
$$
Score matrices: log-odds ratio score

\[ S = \log_2 \frac{P(A,B|M)}{P(A,B|R)} = \sum_{i=1}^{n} \log_2 \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}} = \sum_{i=1}^{n} s(a_i, b_i) \]

- The score \( S \) is obtained by summing over character pair-specific scores:
  \[ s(a, b) = \log_2 \frac{p_{ab}}{q_a q_b} \]

- The probabilities \( q_a \) and \( p_{ab} \) are extracted from data
Calculating score matrices for amino acids

- Probabilities $q_a$ are in principle easy to obtain:  
  - Count relative frequencies of every amino acid in a sequence database

$$s(a, b) = \log_2 \frac{p_{ab}}{q_a q_b}$$
Calculating score matrices for amino acids

- To calculate $p_{ab}$ we can use a known pool of aligned sequences
- BLOCKS is a database of highly conserved regions for proteins
- It lists *multiple aligned*, ungapped and conserved protein segments
- Example from BLOCKS shows genes related to human gene associated with DNA-repair defect xeroderma pigmentosum

$$s(a, b) = \log_2 \frac{p_{ab}}{q_a q_b}$$

Block PR00851A
ID XRODRMPGMNTB; BLOCK
AC PR00851A; distance from previous block=(52,131)
DE Xeroderma pigmentosum group B protein signature
BL adapted; width=21; seqs=8; 99.5%=985; strength=1287

<table>
<thead>
<tr>
<th>Gene</th>
<th>Accession</th>
<th>Length</th>
<th>Conserved Amino Acids</th>
</tr>
</thead>
<tbody>
<tr>
<td>XPB_HUMAN</td>
<td>P19447</td>
<td>74</td>
<td>RPLWVAPDGHIFLEAFSPVK 54</td>
</tr>
<tr>
<td>XPB_MOUSE</td>
<td>P49135</td>
<td>74</td>
<td>RPLWVAPDGHIFLEAFSPVK 54</td>
</tr>
<tr>
<td>P91579</td>
<td>80</td>
<td></td>
<td>RPLYAEPDGHIFLEAFSPVK 67</td>
</tr>
<tr>
<td>XPB_DROME</td>
<td>Q02870</td>
<td>84</td>
<td>RPLWVAPDGHIFLEAFSPVK 79</td>
</tr>
<tr>
<td>RA25_YEAST</td>
<td>Q00578</td>
<td>131</td>
<td>PLWISPSDGRILIEAFSPLEAE 100</td>
</tr>
<tr>
<td>Q38861</td>
<td>52</td>
<td></td>
<td>RPLWACADGRILFTFSPLIF 71</td>
</tr>
<tr>
<td>O13768</td>
<td>90</td>
<td></td>
<td>PLWINPGRILIEAFSPLEAE 100</td>
</tr>
<tr>
<td>O00835</td>
<td>79</td>
<td></td>
<td>RPIWVCAPDHIFLETSAIYK 86</td>
</tr>
</tbody>
</table>

http://blocks.fhcrc.org
BLOSUM is a score matrix for amino acid sequences derived from BLOCKS data.

First, count pairwise matches $f_{x,y}$ for every amino acid type pair $(x, y)$.

For example, for column 3 and amino acids L and W, we find 8 pairwise matches: $f_{L,W} = f_{W,L} = 8$.  

```
RPLWVAPD
RPLWVAPR
RPLWVAPN
PLWISPSD
RPLWACAD
PLWINPID
RPIWVCOD
```
Creating a BLOSUM matrix

- Probability $p_{ab}$ is obtained by dividing $f_{ab}$ with the total number of pairs:

$$p_{ab} = \frac{f_{ab}}{\sum_{x=1}^{20} \sum_{y=1}^{x} f_{xy}}$$

- We get probabilities $q_a$ by

$$q_a = \sum_{b=1}^{20} p_{ab}$$
Creating a BLOSUM matrix

- The probabilities \( p_{ab} \) and \( q_a \) can now be plugged into

\[
    s(a, b) = \log_2 \frac{p_{ab}}{q_a q_b}
\]

to get a **20 x 20** matrix of scores \( s(a, b) \).

- Next slide presents the **BLOSUM62** matrix
  - Values scaled by factor of 2 and rounded to integers
  - Additional step required to take into account expected evolutionary distance
  - Described in more detail in:
|    | A   | R   | N   | D   | C   | Q   | E   | G   | H   | I   | L   | K   | M   | F   | P   | S   | T   | W   | Y   | V   | B   | Z   | X   | *   |
|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| A  | 4   | -1  | -2  | -2  | 0   | -1  | -1  | 0   | -2  | -1  | -1  | -1  | -1  | -1  | -2  | -1  | 1   | 0   | -3  | -2  | 0   | -2  | -1  | 0   | -4  |
| R  | -1  | 5   | 0   | -2  | -3  | 1   | 0   | -2  | 0   | -3  | -2  | 2   | -1  | -3  | -2  | -1  | -1  | -3  | -2  | -3  | -1  | 0   | -1  | -4  |     |
| N  | -2  | 0   | 6   | 1   | -3  | 0   | 0   | 0   | 1   | -3  | -3  | 0   | -2  | -3  | -2  | 1   | 0   | -4  | -2  | -3  | 3   | 0   | -1  | -4  |     |
| D  | -2  | 1   | 6   | 3   | 0   | 2   | -1  | -1  | -1  | -3  | -4  | -1  | -3  | -3  | -1  | 0   | -1  | -4  | -3  | -3  | 4   | 1   | -1  | -4  |     |
| C  | 0   | -3  | -3  | -3  | 9   | -3  | -4  | -3  | -3  | -1  | -1  | -3  | -1  | -2  | -3  | -1  | -1  | -2  | -2  | -1  | -3  | -3  | -2  | -4  |     |
| Q  | -1  | 1   | 0   | 0   | -3  | 5   | 2   | -2  | 0   | -3  | -2  | 1   | 0   | -3  | -1  | 0   | -1  | -2  | -1  | -2  | 0   | 3   | -1  | -4  |     |
| E  | -1  | 0   | 0   | 2   | -4  | 2   | 5   | -2  | 0   | -3  | -3  | 1   | -2  | -3  | -1  | 0   | -1  | -3  | -2  | -2  | 1   | 4   | -1  | -4  |     |
| G  | 0   | -2  | 0   | -1  | -3  | -2  | -2  | 6   | -2  | -4  | -4  | -2  | -3  | -3  | -2  | 0   | -2  | -2  | -3  | -3  | -1  | -2  | -1  | -4  |     |
| H  | -2  | 0   | 1   | -1  | -3  | 0   | 0   | -2  | 8   | -3  | -3  | -1  | -2  | -1  | -2  | -1  | -2  | -2  | 2   | 3   | 0   | 0   | -1  | -4  |     |
| I  | -1  | -3  | -3  | -3  | -1  | -3  | -3  | -4  | -3  | 4   | 2   | -3  | 1   | 0   | -3  | -2  | -1  | -3  | -1  | 3   | -3  | -3  | -1  | -4  |     |
| L  | -1  | -2  | -3  | -4  | -1  | -2  | -3  | -4  | -3  | 2   | 4   | -2  | 2   | 0   | -3  | -2  | -1  | -2  | -1  | 1   | -4  | -3  | -1  | -4  |     |
| K  | -1  | 2   | 0   | -1  | -3  | 1   | 1   | -2  | -1  | -1  | -3  | -2  | 5   | -1  | -3  | -1  | 0   | -1  | -3  | -2  | -2  | 0   | 1   | -1  | -4  |
| M  | -1  | -1  | -2  | -3  | -1  | 0   | -2  | -3  | -2  | 1   | 2   | -1  | 5   | 0   | 0   | -2  | 1   | -1  | -1  | -1  | 1   | -3  | -1  | -1  | -4  |
| F  | -2  | -3  | -3  | -3  | -2  | -3  | -3  | -3  | -1  | 0   | 0   | -3  | 0   | 6   | -4  | -2  | -2  | 1   | 3   | 1   | -3  | -3  | -1  | -4  |     |
| P  | -1  | -2  | -2  | -1  | -3  | -1  | -2  | -2  | -3  | -2  | -3  | -1  | 2   | -1  | -2  | -4  | 7   | -1  | -1  | -4  | -3  | -2  | -2  | -1  | -2  | -4  |
| S  | 1   | -1  | 1   | 0   | -1  | 0   | 0   | 0   | -1  | -2  | -2  | 0   | -1  | -2  | -1  | 4   | 1   | -3  | -2  | -2  | 0   | 0   | 0   | -4  |     |
| T  | 0   | 1   | 0   | -1  | -1  | -1  | -1  | -2  | -2  | -1  | -1  | -1  | -1  | -2  | -1  | 1   | 5   | -2  | -2  | 2   | 0   | -1  | -1  | 0   | -4  |
| W  | -3  | -3  | -4  | -4  | -2  | -2  | -3  | -2  | -3  | -2  | -3  | -1  | 1   | -4  | -3  | -2  | 1   | 1   | -2  | -3  | -4  | -3  | -2  | -4  |     |
| Y  | -2  | -2  | -2  | -3  | -2  | -1  | -2  | -3  | 2   | -1  | -2  | -1  | 3   | -3  | -2  | -2  | 2   | 7   | -1  | -3  | -2  | -1  | -4  |     |
| V  | 0   | -3  | -3  | -3  | -1  | -2  | -2  | -3  | -3  | 3   | 1   | -2  | 1   | -1  | -2  | -2  | 0   | -3  | -1  | 4   | -3  | -2  | -1  | -4  |     |
| B  | -2  | -1  | 3   | -4  | -3  | 0   | 1   | -1  | 0   | -3  | -4  | 0   | -3  | -3  | -2  | 0   | -1  | -4  | -3  | -3  | 4   | 1   | -1  | -4  |     |
| Z  | -1  | 0   | 0   | 1   | -3  | 3   | 4   | -2  | 0   | -3  | -3  | 1   | -1  | -3  | -1  | 0   | -1  | -3  | -2  | -2  | 1   | 4   | -1  | -4  |     |
| X  | 0   | -1  | -1  | -1  | -2  | -1  | -1  | -1  | -1  | -1  | -1  | -1  | -1  | -1  | -2  | 0   | 0   | -2  | -1  | -1  | -1  | -1  | -1  | -4  |     |
| *  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  | -4  |
Using BLOSUM62 matrix

MQLEANADTSV
|   |   |
LQEQAEAGEM

\[ s = \sum_{i=1}^{11} s(a_i, b_i) \]
\[ = 2 + 5 - 3 - 4 + 4 + 0 + 4 + 0 - 2 + 0 + 1 \]
\[ = 7 \]
Why positive score alignment is meaningful?

- We have designed scoring matrix so that expected score of random match at any position is negative:
\[
\sum_{a,b} q_a q_b s(a, b) < 0.
\]

- This can be seen by noticing that
\[
\sum_{a,b} q_a q_b s(a, b) = - \sum_{a,b} q_a q_b \log \left( \frac{q_a q_b}{p_{ab}} \right) = -H(q^2 \| p),
\]
where \( H(q^2 \| p) \) is the relative entropy (or Kullback-Leibler divergence) of distribution \( q^2 = q \times q \) with respect to distribution \( p \). Value of \( H(q^2 \| p) \) is always positive unless \( q^2 = p \). (Exercise: show why.)
What about gap penalties?

- Similar log-odds reasoning gives that the gap penalty should be \(-\log f(k)\), where \(k\) is the gap length, and \(f()\) is the function modeling the replication process (See Durbin et al., page 17).
  - \(-\log \delta k\) for the linear model
  - \(-\log (\alpha + \beta(k - 1))\) for the affine gap model

- However, logarithmic gap penalties are difficult (yet possible) to take into account in dynamic programming:
What about gap penalties? (2)

- Typically some *ad hoc* values are used, like $\delta = 8$ in the linear model and $\alpha = 12$, $\beta = 2$ in the affine gap model.

- It can be argued that penalty of insertion + deletion should be always greater than penalty for one mismatch.
  - Otherwise expected score of random match may get positive.
Multiple alignment

- Consider a set of d sequences on the right
  - Orthologous sequences from different organisms
  - Paralogs from multiple duplications
- How can we study relationships between these sequences?
- Aligning simultaneously many sequences gives better estimates for the homology, as many sequences vote for the same "column".

```
AGCAGTGATGCTAGTTCG
ACAGCAGTGCTAGCTAGTTCG
ACAGAGCTAGCCTAGTTCG
CAGCAGTGCTAGTTCG
ACAAGTGATGCTAGTTCG
ACAGCAGTGCTAGCTAGTTCG
ACAGCGATGCTAGGGTTCG
AAGTGATGCTAGTTCG
ACAGCGATGCTAGGTTGTCG
```
Multiple alignment notation

- Let $M$ denote the multiple alignment, i.e., a matrix with $d$ sequences being the rows with gap symbols ”-” inserted so that all rows are the same length.
- Let $M_{i*}$ and $M_{*j}$ denote the $i$-th row ($j$-th column) in the alignment, respectively, and $M_{ij}$ the symbol at $i$-th row and $j$-th column.

\[
\begin{align*}
M_{i,j} &= A \\
A--GC--AGTG--ATGCTAGTCG \\
ACAGC--AGTG--GATGCTAGTCG \\
ACAG--AGT--GATGCTA--TCG \\
-CAGC--AGTG--CTG--TAGTCG \\
ACA---AGTG--ATGCTAGTCG \\
i ACAGC--AGTG--ATGCTAG--CG \\
A--GC--AGTG--GATGCTAGTCG \\
A--AG------TG--ATGCTAGTCG \\
ACAGCGA--TGCTAGGGT--CG \\
\end{align*}
\]
Applications of multiple alignment

- Amino acid scoring matrix estimation (chicken or the egg problem)
- Phylogeny by parsimony (chicken or the egg problem again)

\[
\begin{align*}
A -- GC -- AGTG -- & ATGCTAGTCG \\
ACAGC -- AGTG -- & GATGCTAGTCG \\
ACAG -- AGT -- & GATGCTA -- TCG \\
-- CAGC -- AGTG -- & CTG -- TAGTCG \\
ACA -- -- AGTG -- & ATGCTAGTCG \\
ACAGC -- AGTG -- & ATGCTAG -- CG \\
A -- GC -- AGTG -- & GATGCTAGTCG \\
A -- AG -- -- TG -- & ATGCTAGTCG \\
ACAGCGA -- TGCTAGGGT -- -- CG 
\end{align*}
\]
Phylogeny by parsimony pipeline

1. genome sequences of the species
2. For all pairs of species, find the homologous genes
3. Select interesting homologs
4. Compute multiple alignment for each homolog family
5. Build the phylogenetic tree based on the aligned columns
Part III

SIGNALS IN DNA
• Genes
• Promoter regions
• Binding sites for regulatory proteins (*transcription factors, enhancer modules, motifs*)
Typical gene

![AMY1 gene image](http://en.wikipedia.org/wiki/File:AMY1gene.png)
Genome analysis pipeline

DNA in vitro → Sequencing and assembly → AGCT...

DNA in silico → Gene prediction → annotation

RNA in vitro → Sequencing and assembly → ...AUG...

Gene regulation studies: systems biology

DNA and annotation of other species
Gene regulation

- Let us assume that gene prediction is done.
- We are interested in signals that influence gene regulation:
  - How much mRNA is transcribed, how much protein is translated?
  - How to measure those?
    - 2D gel electrophoresis (traditional technique to measure protein expression)
    - Microarrays (the standard technique to measure RNA expression)
    - RNA-sequencing (a new technique to measure RNA expression, useful for many other purposes as well, including gene prediction)
Microarrays and gene expression

- Idea:

A probe specific to the gene: e.g. complement of short unique fragment of cDNA

It is possible to make a series of microarray experiments to obtain a time series expression profile for each gene.

*Cluster* similarly behaving genes.
Analysis of clustered genes

• Similarly expressing genes may share a common transcription factor located upstream of the gene sequence.
  ○ Extract those sequences from the clustered genes and search for a common motif sequence.
  ○ Some basic techniques for motif discovery covered in the Algorithms for Bioinformatics course (period I) and more advanced ones in Algorithms in Molecular Biology course (next year).

• We concentrate now on the structure of upstream region, representation of motifs, and the simple tasks of locating the occurrences of already known motifs.
Promoter sequences

- Immediately before the gene.
- Clear structure in prokaryotes, more complex in eukaryotes.
- An example from *E. coli* is shown in next slide (from Deonier et al. book).
Table 9.2. A sample of *E. coli* promoter sequences. These sequences have been aligned relative to the transcriptional start site at position +1 (boldface large letter). Sequences from −40 to +11 are shown. Close matches to consensus −35 and −10 hexamers are underlined. See also Appendix C.3 for additional examples and sources of the data.

<table>
<thead>
<tr>
<th>ORF83P1</th>
<th>ada</th>
<th>amnP4</th>
<th>araFGH</th>
<th>aroG</th>
<th>atpI</th>
<th>caiT</th>
<th>clpAP1</th>
<th>crpP2-I</th>
</tr>
</thead>
<tbody>
<tr>
<td>CTCTGCTGGCATTCACAATGCGCAGGGGTAAACGTTTCCCTGTAAGCACC</td>
<td>GTTGGTTTTTGGTGATGTGGACCGGGCAGCCTAAGGCTATCTTTAACCA</td>
<td>TTCACATTTCTGTGACATACATATCGATGTCCGTAAATTGTATGGAACAGG</td>
<td>CTCTCTATGGAAGATTAATTTTCCTCGCTAAAACATATGTCAACACAGTCACT</td>
<td>CCCCCTTTACACATTCTGACGGGAAGATATAGATGGGAAGTATTGCATTCAC</td>
<td>TATGTGGAAAACACGGGGGCGCACCCTATAATTGTAGCCGCTTTTTGTAT</td>
<td>AATCACAGAATACAGGTTTATCGAATACCATTTATGAGTGATAGCCATTAACCG</td>
<td>TTATGGACGTGTTACAAATCTTTTCTTATGATGTAGAACGTGCAACCGC</td>
<td>GTGGTGACCTTCTGGCGATGAACGTTGTACACTTCTGTGGCGATGG</td>
</tr>
</tbody>
</table>
Representing signals in DNA

- **Consensus sequence:**
  - -10 site in E coli: TATAAT
  - GRE half-site consensus: AGAACA

- **Simple regular expression:**
  - A(C/G)AA(C/G)(A/T)

- **Positional weight matrix (PWM):**

  \[
  \begin{bmatrix}
  A & 1.00 & 0.00 & 1.00 & 1.00 & 0.00 & 0.86 \\
  C & 0.00 & 0.14 & 0.00 & 0.00 & 0.86 & 0.00 \\
  G & 0.00 & 0.86 & 0.00 & 0.00 & 0.14 & 0.00 \\
  T & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 0.14 \\
  \end{bmatrix}
  \]

  **GRE half-sites:**
  - AGAACA
  - ACAACA
  - AGAACA
  - AGAAGA
  - AGAACA
  - AGAACT
  - AGAACA

  **Consensus:** AGAACA
Position-specific scoring matrix (PSSM)

- PSSM is a log-odds normalized version of PWM. \(^1\)
- Calculated by \(\log(p_{ai}/q_a)\), where
  - \(p_{ai}\) is the frequency of \(a\) at column \(i\) in the samples.
  - \(q_a\) is the probability of \(a\) in the whole organism (or in some region of interest).
- Problematic when some values \(p_{ai}\) are zero.
- Solution is to use pseudocounts:
  - add 1 to all the sample counts where the frequencies are calculated.

\(^1\) In the following \(\log\) denotes base 2 logarithm.
PWM versus PSSM

Counts

\[
\begin{bmatrix}
7 & 0 & 7 & 7 & 0 & 6 \\
0 & 1 & 0 & 0 & 6 & 0 \\
0 & 6 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

PSSM (position-specific scoring matrix)

\[
\begin{bmatrix}
1.54 & -1.46 & 1.54 & 1.54 & -1.46 & 1.35 \\
-1.46 & -0.46 & -1.46 & -1.46 & 1.35 & -1.46 \\
-1.46 & 1.35 & -1.46 & -1.46 & -0.46 & -1.46 \\
-1.46 & -1.46 & -1.46 & -1.46 & -1.46 & -0.46 \\
\end{bmatrix}
\]

assuming \( q_a = 0.25 \) for all \( a \)
Many known transcription factor binding site PWM:s can be found from JASPAR database (http://jaspar.cgb.ki.se/).

PWM:s are visualized as sequence logos, where the height of each nucleotide equals its proportion of the relative entropy (expected log-odds score) in that column.

1. \[ E(S_i) = \sum_a p_{ai} \log(p_{ai} / q_a) \]
2. Height of \( a \) at column \( i \) is \( p_{ai}E(S_i) \)
Example sequence logo

\[
\begin{bmatrix}
1.54 & -1.46 & 1.54 & 1.54 & -1.46 & 1.35 \\
-1.46 & -0.46 & -1.46 & -1.46 & 1.35 & -1.46 \\
-1.46 & 1.35 & -1.46 & -1.46 & -0.46 & -1.46 \\
-1.46 & -1.46 & -1.46 & -1.46 & -1.46 & -0.46
\end{bmatrix}
\]

2 bits

A G A A C A
Searching PSSMs

- As easy as naive exact text search (see next slide).
- Much faster methods exist. For example, one can apply branch-and-bound technique on top of suffix tree (discussed later during the course).

**Warning:**
- Good hits for any PSSM are too easy to find!
- Search domain must be limited by other means to find anything statistically meaningful with PSSMs only.
  - Typically used on upstream regions of genes clustered by gene expression profiling.
```python
#!/usr/bin/env python
import sys
import time

# naive PSSM search
matrix = {'A':[1.54,-1.46,1.54,1.54,-1.46,1.35],
          'C':[-1.46,-0.46,-1.46,-1.45,1.35,-1.46],
          'G':[-1.46,1.35,-1.46,-1.46,-0.46,-1.46],
          'T':[-1.46,-1.46,-1.46,-1.46,-1.46,-0.46]}
count = {'A':0,'C':0,'G':0,'T':0}
textf = open(sys.argv[1],'r')
text = textf.read()
m=len(matrix['A'])
bestscore = -m*2.0
t1 = time.time()
for i in range(len(text)-m+1):
    score = 0.0
    for j in range(m):
        if text[i+j] in matrix:
            score = score + matrix[text[i+j]][j]
            count[text[i+j]] = count[text[i+j]]+1
        else:
            score = -m*2.0
        if score > bestscore:
            bestscore = score
            bestindex = i
    t2 = time.time()
totalcount = count['A']+count['C']+count['G']+count['T']
expectednumberofhits = 1.0*(len(text)-m+1)
for j in range(m):
    expectednumberofhits = expectednumberofhits*float(count[text[bestindex+j]])/float(totalcount)
print 'best score: ' + str(bestscore) + ' at index: ' +str(bestindex)
print 'best hit: ' + text[bestindex:bestindex+m]
print 'computation took: ' + str(t2-t1) + ' seconds'
print 'expected number of hits: ' + str(expectednumberofhits)
```

pssm.py hs_ref_chrY_nolinebreaks.fa
best score 8.67 at index 397
best hit: AGAACA
computation took 440.56187582 seconds
expected number of hits: 18144.7627936

no sense in this search!
Refined motifs

- Our example PSSM (GRE half-site) represents only half of the actual motif: the complete motif is a palindrome with consensus:
  - AGAACAnnnTGTTCT

```
./pssmpalindrome.py hs_ref_chrY_nolinebreaks.fa
best score 17.34 at index 17441483
best hit: AGAACAGGCTGTTCT
computation took 1011.4800241 seconds
expected number of hits: 5.98440033042
total number of maximum score hits: 2
```

- Exercise: modify pssm.py into pssmpalindrome.py
  ... or learn biopython to do the same in few lines of code
Discovering motifs

- **Principle:** discover over-represented motifs from the promotor / enhancer regions of co-expressing genes.

- **How to define a motif?**
  - Consensus, PWM, PSSM, palindrome PSSM, co-occurrence of several motifs (enhancer modules), ...
  - Abstractions of protein-DNA chemical binding.

- **Computational challenge in motif discovery:**
  - Almost as hard as (local) multiple alignment.
  - Exhaustive methods too slow.
  - Lots of specialized pruning mechanisms exist.

- **New sequencing technologies will help (ChIP-seq).**
Part IV

BIOLOGICAL WORDS
Biological words: k-mer statistics

To understand statistical approaches to gene prediction (later during the course), we need to study what is known about the structure and statistics of DNA.

- 1-mers: individual nucleotides (bases)
- 2-mers: dinucleotides (AA, AC, AG, AT, CA, ...)
- 3-mers: codons (AAA, AAC, ...)
- 4-mers and beyond
1-mers: base composition

- Typically DNA exists as *duplex* molecule (two complementary strands)

```
5’ –GGATCGAAGCTAAGGGCT–3’
3’ –CCTAGCTTTCGATTCCCGA–5’
```

Top strand: 7 G, 3 C, 5 A, 3 T
Bottom strand: 3 G, 7 C, 3 A, 5 T
Duplex molecule: 10 G, 10 C, 8 A, 8 T
Base frequencies: 10/36 10/36 8/36 8/36

fr(G + C) = 20/36, fr(A + T) = 1 – fr(G + C) = 16/36

These are something we can determine experimentally.
G+C content

- \( \text{fr}(G + C) \), or \( G+C \) content is a simple statistics for describing genomes
- Notice that one value is enough to characterise \( \text{fr}(A) \), \( \text{fr}(C) \), \( \text{fr}(G) \) and \( \text{fr}(T) \) for duplex DNA
- Is G+C content (= base composition) able to tell the difference between genomes of different organisms?
<table>
<thead>
<tr>
<th>Organism</th>
<th>G+C Content</th>
<th>Genome Size (Mb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mycoplasma genitalium</td>
<td>31.6%</td>
<td>0.585</td>
</tr>
<tr>
<td>Escherichia coli K-12</td>
<td>50.7%</td>
<td>4.693</td>
</tr>
<tr>
<td>Pseudomonas aeruginosa PAO1</td>
<td>66.4%</td>
<td>6.264</td>
</tr>
<tr>
<td>Pyrococcus abyssi</td>
<td>44.6%</td>
<td>1.765</td>
</tr>
<tr>
<td>Thermoplasma volcanium</td>
<td>39.9%</td>
<td>1.585</td>
</tr>
<tr>
<td>Caenorhabditis elegans</td>
<td>36%</td>
<td>97</td>
</tr>
<tr>
<td>Arabidopsis thaliana</td>
<td>35%</td>
<td>125</td>
</tr>
<tr>
<td>Homo sapiens</td>
<td>41%</td>
<td>3080</td>
</tr>
</tbody>
</table>
Consider a DNA sequence generated randomly, with probability of each letter being independent of position in sequence.

You could expect to find a uniform distribution of bases in genomes...

\[ \text{5'} \ldots \text{GGATCGAAGCTAAGGGCT} \ldots \text{3'} \]
\[ \text{3'} \ldots \text{CCTAGCTTTCGATTCCCGA} \ldots \text{5'} \]

This is not, however, the case in genomes, especially in prokaryotes.

- This phenomena is called GC skew.
When DNA is replicated, the molecule takes the replication fork form.

New complementary DNA is synthesised at both strands of the "fork".

New strand in 5’-3’ direction corresponding to replication fork movement is called leading strand and the other lagging strand.
DNA replication fork

- This process has specific starting points in genome (*origins of replication*)
- Observation: Leading strands have an excess of G over C
- This can be described by *GC skew* statistics
GC skew

- GC skew is defined as $(\#G - \#C) / (\#G + \#C)$
- It is calculated at successive positions in intervals (windows) of specific width

$$\frac{4 - 2}{4 + 2} = \frac{1}{3}$$
$$\frac{3 - 2}{3 + 2} = \frac{1}{5}$$
G-C content & GC skew

- G-C content & GC skew statistics can be displayed with a circular genome map

Chromosome map of *S. dysenteriae*, the nine rings describe different properties of the genome

http://www.mgc.ac.cn/ShiBASE/circular_Sd197.htm
GC skew often changes sign at origins and termini of replication.

(G+C content)

(GC skew)

(10kb window size)

Nie et al., BMC Genomics, 2006
i.i.d. model for nucleotides

- Assume that bases
  - occur independently of each other
  - bases at each position are identically distributed

- Probability of the base A, C, G, T occurring is $p_A$, $p_C$, $p_G$, $p_T$, respectively
  - For example, we could use $p_A=p_C=p_G=p_T=0.25$ or estimate the values from known genome data

- Joint probability is then just the product of independent variables
  - For example, $P(TG) = p_T p_G$
Refining the i.i.d. model

• i.i.d. model describes some organisms well (see Deonier et al. book) but fails to characterise many others

• We can refine the model by having the DNA letter at some position depend on letters at preceding positions

…TCGTGACGCCG?

Sequence context to consider
First-order Markov chains

- Let’s assume that in sequence X the letter at position t, $X_t$, depends only on the previous letter $X_{t-1}$ (first-order markov chain)
- Probability of letter b occurring at position t given $X_{t-1} = a$: $p_{ab} = P(X_t = b \mid X_{t-1} = a)$
- We consider homogeneous markov chains: probability $p_{ab}$ is independent of position t
We can estimate probabilities $p_{ab}$ ("the probability that b follows a") from observed dinucleotide frequencies.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$p_{AA}$</td>
<td>$p_{AC}$</td>
<td>$p_{AG}$</td>
<td>$p_{AT}$</td>
</tr>
<tr>
<td>C</td>
<td>$p_{CA}$</td>
<td>$p_{CC}$</td>
<td>$p_{CG}$</td>
<td>$p_{CT}$</td>
</tr>
<tr>
<td>G</td>
<td>$p_{GA}$</td>
<td>$p_{GC}$</td>
<td>$p_{GG}$</td>
<td>$p_{GT}$</td>
</tr>
<tr>
<td>T</td>
<td>$p_{TA}$</td>
<td>$p_{TC}$</td>
<td>$p_{TG}$</td>
<td>$p_{TT}$</td>
</tr>
</tbody>
</table>

...the values $p_{AA}$, $p_{AC}$, ..., $p_{TG}$, $p_{TT}$ sum to 1.
Estimating $p_{ab}$

- $p_{ab} = P(X_t = b \mid X_{t-1} = a) = \frac{P(X_t = b, X_{t-1} = a)}{P(X_{t-1} = a)}$

  - Probability of transition $a \rightarrow b$
  - Dinucleotide frequency
  - Base frequency of nucleotide $a$, $fr(a)$

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.146</td>
<td>0.052</td>
<td>0.058</td>
<td>0.089</td>
</tr>
<tr>
<td>C</td>
<td>0.063</td>
<td>0.029</td>
<td>0.010</td>
<td>0.056</td>
</tr>
<tr>
<td>G</td>
<td>0.050</td>
<td>0.030</td>
<td>0.028</td>
<td>0.051</td>
</tr>
<tr>
<td>T</td>
<td>0.086</td>
<td>0.047</td>
<td>0.063</td>
<td>0.140</td>
</tr>
</tbody>
</table>

$P(X_t = b, X_{t-1} = a)$

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.423</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>0.399</td>
<td>0.184</td>
<td>0.063</td>
</tr>
<tr>
<td>G</td>
<td></td>
<td></td>
<td>0.314</td>
<td>0.189</td>
</tr>
<tr>
<td>T</td>
<td></td>
<td></td>
<td></td>
<td>0.258</td>
</tr>
</tbody>
</table>

$P(X_t = b \mid X_{t-1} = a)$

0.052 / 0.345 ≈ 0.151
Simulating a DNA sequence

- From a transition matrix, it is easy to generate a DNA sequence of length n:
  - First, choose the starting base randomly according to the base frequency distribution
  - Then, choose next base according to the distribution $P(X_t \mid X_{t-1})$ until n bases have been chosen

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.423</td>
<td>0.151</td>
<td>0.168</td>
<td>0.258</td>
</tr>
<tr>
<td>C</td>
<td>0.399</td>
<td>0.184</td>
<td>0.063</td>
<td>0.354</td>
</tr>
<tr>
<td>G</td>
<td>0.314</td>
<td>0.189</td>
<td>0.176</td>
<td>0.321</td>
</tr>
<tr>
<td>T</td>
<td>0.258</td>
<td>0.138</td>
<td>0.187</td>
<td>0.415</td>
</tr>
</tbody>
</table>

Look for R code in Deonier et al. book

$P(X_t = b \mid X_{t-1} = a)$
#!/usr/bin/env python

import sys, random

n = int(sys.argv[1])

tm = {'a': {'a': 0.423, 'c': 0.151, 'g': 0.168, 't': 0.258},
     'c': {'a': 0.399, 'c': 0.184, 'g': 0.063, 't': 0.354},
     'g': {'a': 0.314, 'c': 0.189, 'g': 0.176, 't': 0.321},
     't': {'a': 0.258, 'c': 0.138, 'g': 0.187, 't': 0.415}}

pi = {'a': 0.345, 'c': 0.158, 'g': 0.159, 't': 0.337}

def choose(dist):
    r = random.random()
    sum = 0.0
    keys = dist.keys()
    for k in keys:
        sum += dist[k]
        if sum > r:
            return k
    return keys[-1]

c = choose(pi)
for i in range(n - 1):
    sys.stdout.write(c)
    c = choose(tm[c])
sys.stdout.write(c)
sys.stdout.write("\n")

Example Python code for generating DNA sequences with first-order Markov chains.

Initialisation: use packages 'sys' and 'random', read sequence length from input.

Transition matrix tm and initial distribution pi.

Function choose(), returns a key (here 'a', 'c', 'g' or 't') of the dictionary 'dist' chosen randomly according to probabilities in dictionary values.

Choose the first letter, then choose next letter according to $P(x_t | x_{t-1})$. 
Simulating a DNA sequence

- Now we can quickly generate sequences of arbitrary length...

```
ttctcacaataaaggatagtgattcttatttttctgttataaggatagtcatctaggatttttttcatgtatggtcctattttaagaagttgtaacgtaataaggcttactgcgagcattttttgttttaactgctaggtttttctttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttttt
``
Simulating a DNA sequence

Dinucleotide frequencies
Simulated  Observed

<table>
<thead>
<tr>
<th></th>
<th>Simulated</th>
<th>Observed</th>
</tr>
</thead>
<tbody>
<tr>
<td>aa</td>
<td>0.145</td>
<td>0.146</td>
</tr>
<tr>
<td>ac</td>
<td>0.050</td>
<td>0.052</td>
</tr>
<tr>
<td>ag</td>
<td>0.055</td>
<td>0.058</td>
</tr>
<tr>
<td>at</td>
<td>0.092</td>
<td>0.089</td>
</tr>
<tr>
<td>ca</td>
<td>0.065</td>
<td>0.063</td>
</tr>
<tr>
<td>cc</td>
<td>0.028</td>
<td>0.029</td>
</tr>
<tr>
<td>cg</td>
<td>0.011</td>
<td>0.010</td>
</tr>
<tr>
<td>ct</td>
<td>0.058</td>
<td>0.056</td>
</tr>
<tr>
<td>ga</td>
<td>0.048</td>
<td>0.050</td>
</tr>
<tr>
<td>gc</td>
<td>0.032</td>
<td>0.030</td>
</tr>
<tr>
<td>gg</td>
<td>0.029</td>
<td>0.028</td>
</tr>
<tr>
<td>gt</td>
<td>0.050</td>
<td>0.051</td>
</tr>
<tr>
<td>ta</td>
<td>0.084</td>
<td>0.086</td>
</tr>
<tr>
<td>tc</td>
<td>0.052</td>
<td>0.047</td>
</tr>
<tr>
<td>tg</td>
<td>0.064</td>
<td>0.063</td>
</tr>
<tr>
<td>tt</td>
<td>0.138</td>
<td>0.0140</td>
</tr>
</tbody>
</table>

n = 10000
Simulating a DNA sequence

- The model is able to generate correct proportions of 1- and 2-mers in genomes...
- ...but fails with \( k=3 \) and beyond.
3-mers: codons

- We can extend the previous method to 3-mers
- k=3 is an important case in study of DNA sequences because of genetic code
3-mers in *Escherichia coli* genome

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
<th>Observed</th>
<th>Expected</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>108924</td>
<td>0.02348</td>
<td>0.01492</td>
</tr>
<tr>
<td>AAC</td>
<td>82582</td>
<td>0.01780</td>
<td>0.01541</td>
</tr>
<tr>
<td>AAG</td>
<td>63369</td>
<td>0.01366</td>
<td>0.01537</td>
</tr>
<tr>
<td>AAT</td>
<td>82995</td>
<td>0.01789</td>
<td>0.01490</td>
</tr>
<tr>
<td>ACA</td>
<td>58637</td>
<td>0.01264</td>
<td>0.01541</td>
</tr>
<tr>
<td>ACC</td>
<td>74897</td>
<td>0.01614</td>
<td>0.01591</td>
</tr>
<tr>
<td>ACG</td>
<td>73263</td>
<td>0.01579</td>
<td>0.01588</td>
</tr>
<tr>
<td>ACT</td>
<td>49865</td>
<td>0.01075</td>
<td>0.01539</td>
</tr>
<tr>
<td>AGA</td>
<td>56621</td>
<td>0.01220</td>
<td>0.01537</td>
</tr>
<tr>
<td>AGC</td>
<td>80860</td>
<td>0.01743</td>
<td>0.01588</td>
</tr>
<tr>
<td>AGG</td>
<td>50624</td>
<td>0.01091</td>
<td>0.01584</td>
</tr>
<tr>
<td>AGT</td>
<td>49772</td>
<td>0.01073</td>
<td>0.01536</td>
</tr>
<tr>
<td>ATA</td>
<td>63697</td>
<td>0.01373</td>
<td>0.01490</td>
</tr>
<tr>
<td>ATC</td>
<td>86486</td>
<td>0.01864</td>
<td>0.01539</td>
</tr>
<tr>
<td>ATG</td>
<td>76238</td>
<td>0.01643</td>
<td>0.01536</td>
</tr>
<tr>
<td>ATT</td>
<td>83398</td>
<td>0.01797</td>
<td>0.01489</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
<th>Observed</th>
<th>Expected</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAA</td>
<td>76614</td>
<td>0.01651</td>
<td>0.01541</td>
</tr>
<tr>
<td>CAC</td>
<td>66751</td>
<td>0.01439</td>
<td>0.01591</td>
</tr>
<tr>
<td>CAG</td>
<td>104799</td>
<td>0.02259</td>
<td>0.01588</td>
</tr>
<tr>
<td>CAT</td>
<td>76985</td>
<td>0.01659</td>
<td>0.01539</td>
</tr>
<tr>
<td>CCA</td>
<td>86436</td>
<td>0.01863</td>
<td>0.01591</td>
</tr>
<tr>
<td>CCC</td>
<td>47775</td>
<td>0.01030</td>
<td>0.01643</td>
</tr>
<tr>
<td>CCG</td>
<td>87036</td>
<td>0.01876</td>
<td>0.01640</td>
</tr>
<tr>
<td>CCT</td>
<td>50426</td>
<td>0.01087</td>
<td>0.01589</td>
</tr>
<tr>
<td>CGA</td>
<td>70938</td>
<td>0.01529</td>
<td>0.01588</td>
</tr>
<tr>
<td>CGC</td>
<td>115695</td>
<td>0.02494</td>
<td>0.01640</td>
</tr>
<tr>
<td>CGG</td>
<td>86877</td>
<td>0.01872</td>
<td>0.01636</td>
</tr>
<tr>
<td>CGT</td>
<td>73160</td>
<td>0.01577</td>
<td>0.01586</td>
</tr>
<tr>
<td>CTA</td>
<td>26764</td>
<td>0.00577</td>
<td>0.01539</td>
</tr>
<tr>
<td>CTC</td>
<td>42733</td>
<td>0.00921</td>
<td>0.01589</td>
</tr>
<tr>
<td>CTG</td>
<td>102909</td>
<td>0.02218</td>
<td>0.01586</td>
</tr>
<tr>
<td>CTT</td>
<td>63655</td>
<td>0.01372</td>
<td>0.01537</td>
</tr>
</tbody>
</table>
3-mers in *Escherichia coli* genome

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
<th>Observed</th>
<th>Expected</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAA</td>
<td>83494</td>
<td>0.01800</td>
<td>0.01537</td>
</tr>
<tr>
<td>GAC</td>
<td>54737</td>
<td>0.01180</td>
<td>0.01588</td>
</tr>
<tr>
<td>GAG</td>
<td>42465</td>
<td>0.00915</td>
<td>0.01584</td>
</tr>
<tr>
<td>GAT</td>
<td>86551</td>
<td>0.01865</td>
<td>0.01536</td>
</tr>
<tr>
<td>GCA</td>
<td>96028</td>
<td>0.02070</td>
<td>0.01588</td>
</tr>
<tr>
<td>GCC</td>
<td>92973</td>
<td>0.02004</td>
<td>0.01640</td>
</tr>
<tr>
<td>GCG</td>
<td>114632</td>
<td>0.02471</td>
<td>0.01636</td>
</tr>
<tr>
<td>GCT</td>
<td>80298</td>
<td>0.01731</td>
<td>0.01539</td>
</tr>
<tr>
<td>GGA</td>
<td>56197</td>
<td>0.01211</td>
<td>0.01584</td>
</tr>
<tr>
<td>GGC</td>
<td>92144</td>
<td>0.01986</td>
<td>0.01636</td>
</tr>
<tr>
<td>GGG</td>
<td>47495</td>
<td>0.01024</td>
<td>0.01632</td>
</tr>
<tr>
<td>GGT</td>
<td>74301</td>
<td>0.01601</td>
<td>0.01582</td>
</tr>
<tr>
<td>GTA</td>
<td>52672</td>
<td>0.01135</td>
<td>0.01536</td>
</tr>
<tr>
<td>GTC</td>
<td>54221</td>
<td>0.01169</td>
<td>0.01586</td>
</tr>
<tr>
<td>GTG</td>
<td>66117</td>
<td>0.01425</td>
<td>0.01582</td>
</tr>
<tr>
<td>GTT</td>
<td>82598</td>
<td>0.01780</td>
<td>0.01534</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
<th>Observed</th>
<th>Expected</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAA</td>
<td>68838</td>
<td>0.01484</td>
<td>0.01490</td>
</tr>
<tr>
<td>TAC</td>
<td>52592</td>
<td>0.01134</td>
<td>0.01539</td>
</tr>
<tr>
<td>TAG</td>
<td>27243</td>
<td>0.00587</td>
<td>0.01536</td>
</tr>
<tr>
<td>TAT</td>
<td>63288</td>
<td>0.01364</td>
<td>0.01489</td>
</tr>
<tr>
<td>TCA</td>
<td>84048</td>
<td>0.01812</td>
<td>0.01539</td>
</tr>
<tr>
<td>TCC</td>
<td>56028</td>
<td>0.01208</td>
<td>0.01589</td>
</tr>
<tr>
<td>TCG</td>
<td>71739</td>
<td>0.01546</td>
<td>0.01586</td>
</tr>
<tr>
<td>TCT</td>
<td>55472</td>
<td>0.01196</td>
<td>0.01537</td>
</tr>
<tr>
<td>TGA</td>
<td>83491</td>
<td>0.01800</td>
<td>0.01537</td>
</tr>
<tr>
<td>TGC</td>
<td>95232</td>
<td>0.02053</td>
<td>0.01586</td>
</tr>
<tr>
<td>TGG</td>
<td>85141</td>
<td>0.01835</td>
<td>0.01582</td>
</tr>
<tr>
<td>TGT</td>
<td>58375</td>
<td>0.01258</td>
<td>0.01534</td>
</tr>
<tr>
<td>TTA</td>
<td>68828</td>
<td>0.01483</td>
<td>0.01489</td>
</tr>
<tr>
<td>TTC</td>
<td>83848</td>
<td>0.01807</td>
<td>0.01537</td>
</tr>
<tr>
<td>TTG</td>
<td>76975</td>
<td>0.01659</td>
<td>0.01534</td>
</tr>
<tr>
<td>TTT</td>
<td>109831</td>
<td>0.02367</td>
<td>0.01487</td>
</tr>
</tbody>
</table>
2nd order Markov Chains

- Markov chains readily generalise to higher orders
- In 2nd order markov chain, position t depends on positions t-1 and t-2
- Transition matrix:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>C</th>
<th>G</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AC</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CA</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Observation: cells prefer certain codons over others in highly expressed genes
- Gene expression: DNA is transcribed into RNA (and possibly translated into protein)

<table>
<thead>
<tr>
<th>Amino acid</th>
<th>Codon</th>
<th>Predicted</th>
<th>Gene class I</th>
<th>Gene class II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phe</td>
<td>TTT</td>
<td>0.493</td>
<td>0.551</td>
<td>0.291</td>
</tr>
<tr>
<td></td>
<td>TTC</td>
<td>0.507</td>
<td>0.449</td>
<td>0.709</td>
</tr>
<tr>
<td>Ala</td>
<td>GCT</td>
<td>0.246</td>
<td>0.145</td>
<td>0.275</td>
</tr>
<tr>
<td></td>
<td>GCC</td>
<td>0.254</td>
<td>0.276</td>
<td>0.164</td>
</tr>
<tr>
<td></td>
<td>GCA</td>
<td>0.246</td>
<td>0.196</td>
<td>0.240</td>
</tr>
<tr>
<td></td>
<td>GCG</td>
<td>0.254</td>
<td>0.382</td>
<td>0.323</td>
</tr>
<tr>
<td>Asn</td>
<td>AAT</td>
<td>0.493</td>
<td>0.409</td>
<td>0.172</td>
</tr>
<tr>
<td></td>
<td>AAC</td>
<td>0.507</td>
<td>0.591</td>
<td>0.828</td>
</tr>
</tbody>
</table>

Codon frequencies for some genes in E. coli
Consider an amino acid sequence $X = x_1x_2...x_n$

Let $p_k$ be the probability that codon $k$ is used in highly expressed genes

Let $q_k$ be the highest probability that a codon coding for the same amino acid as codon $k$ has

- For example, if codon $k$ is ”GCC”, the corresponding amino acid is Alanine (see genetic code table; also GCT, GCA, GCG code for Alanine)
- Assume that $p_{GCC} = 0.164$, $p_{GCT} = 0.275$, $p_{GCA} = 0.240$, $p_{GCG} = 0.323$
- Now $q_{GCC} = q_{GCT} = q_{GCA} = q_{GCG} = 0.323$
Codon Adaptation Index (CAI)

- CAI is defined as

\[ CAI = \left( \prod_{k=1}^{n} \frac{p_k}{q_k} \right)^{1/n} \]

- CAI can be given also in log-odds form:

\[ \log(\text{CAI}) = \frac{1}{n} \sum_{k=1}^{n} \log\left( \frac{p_k}{q_k} \right) \]
CAI: example with an E. coli gene

<table>
<thead>
<tr>
<th>M</th>
<th>A</th>
<th>L</th>
<th>T</th>
<th>K</th>
<th>A</th>
<th>E</th>
<th>M</th>
<th>S</th>
<th>E</th>
<th>Y</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATG</td>
<td>GCG</td>
<td>CTT</td>
<td>ACA</td>
<td>AAA</td>
<td>GCT</td>
<td>GAA</td>
<td>ATG</td>
<td>TCA</td>
<td>GAA</td>
<td>TAT</td>
<td>CTG</td>
</tr>
<tr>
<td><strong>1.00</strong></td>
<td>0.47</td>
<td>0.02</td>
<td>0.45</td>
<td><strong>0.80</strong></td>
<td><strong>0.47</strong></td>
<td><strong>0.79</strong></td>
<td><strong>1.00</strong></td>
<td>0.43</td>
<td><strong>0.79</strong></td>
<td>0.19</td>
<td>0.02</td>
</tr>
<tr>
<td>0.06</td>
<td>0.02</td>
<td><strong>0.47</strong></td>
<td>0.20</td>
<td>0.06</td>
<td>0.21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.28</td>
<td><strong>0.04</strong></td>
<td>0.04</td>
<td>0.28</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>0.20</strong></td>
<td>0.03</td>
<td>0.05</td>
<td>0.20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>0.89</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ATG</th>
<th>GCT</th>
<th>TTA</th>
<th>ACT</th>
<th>AAA</th>
<th>GCT</th>
<th>GAA</th>
<th>ATG</th>
<th>TCT</th>
<th>GAA</th>
<th>TAT</th>
<th>TTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCC</td>
<td>TTG</td>
<td><strong>ACC</strong></td>
<td>AAG</td>
<td>GCC</td>
<td>GAG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GCA</td>
<td>CTT</td>
<td>ACA</td>
<td>GCA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GCG</td>
<td>CTC</td>
<td>ACG</td>
<td>GCG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>CTG</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{bmatrix}
1.00 & 0.20 & 0.04 & 0.04 & 0.80 & 0.47 & 0.79 & 1.00 & 0.03 & 0.79 & 0.19 & 0.89 \ 
1.00 & 0.47 & 0.89 & 0.47 & 0.80 & 0.47 & 0.79 & 1.00 & 0.43 & 0.79 & 0.81 & 0.89 \\
\end{bmatrix}
\]
CAI: properties

- CAI = 1.0: each codon was the most frequently used codon in highly expressed genes
- Log-odds used to avoid numerical problems
  - What happens if you multiply many values <1.0 together?
- In a sample of E.coli genes, CAI ranged from 0.2 to 0.85
- CAI correlates with mRNA levels: can be used to predict high expression levels
Simple 1-, 2- and 3-mer models can describe interesting properties of DNA sequences:

- GC skew can identify DNA replication origins.
- It can also reveal genome rearrangement events and lateral transfer of DNA.
- GC content can be used to locate genes: human genes are comparably GC-rich.
- CAI predicts high gene expression levels.
Biological words: summary

- *k=3* models can help to identify correct *reading frames*
  - Reading frame starts from a start codon and stops in a stop codon
  - Consider what happens to translation when a single extra base is introduced in a reading frame
- Also word models for *k > 3* have their uses