CMPS 6630: Introduction to Computational Biology and Bioinformatics

Gene Prediction
Now What?
Suppose we want to annotate a genome according to genetic traits.

Given a genome, where are the genes?

Given a gene, where on the genome did it come from?
Finding Genes

Given a strand of mRNA, can we just look for Met and “STOP” codons?
Finding Genes

Given a strand of mRNA, can we just look for Met and “STOP” codons?

<table>
<thead>
<tr>
<th>U</th>
<th>C</th>
<th>A</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>UUU = Phe</td>
<td>UCU = Ser</td>
<td>UAU = Tyr</td>
<td>UGU = Cys</td>
</tr>
<tr>
<td>UUC = Phe</td>
<td>UCC = Ser</td>
<td>UAC = Tyr</td>
<td>UGC = Cys</td>
</tr>
<tr>
<td>UUA = Leu</td>
<td>UCA = Ser</td>
<td>UAA = Stop</td>
<td>UAG = Stop</td>
</tr>
<tr>
<td>UUG = Leu</td>
<td>UCG = Ser</td>
<td>UGA = Stop</td>
<td></td>
</tr>
<tr>
<td>UAU = Tyr</td>
<td>UAC = Tyr</td>
<td>UAA = Stop</td>
<td>UGA = Stop</td>
</tr>
<tr>
<td>UAG = Stop</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UGU = Cys</td>
<td>UGC = Cys</td>
<td>UGA = Stop</td>
<td></td>
</tr>
<tr>
<td>UGG = Trp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>U</td>
<td>C</td>
<td>A</td>
<td>G</td>
</tr>
<tr>
<td>CUU = Leu</td>
<td>CUC = Leu</td>
<td>CAU = His</td>
<td>CGU = Arg</td>
</tr>
<tr>
<td>CUC = Leu</td>
<td>CCA = Pro</td>
<td>CAC = His</td>
<td>CGG = Arg</td>
</tr>
<tr>
<td>CUA = Leu</td>
<td>CCC = Pro</td>
<td>CAA = Gln</td>
<td>CAG = Gln</td>
</tr>
<tr>
<td>CUG = Leu</td>
<td>CCG = Pro</td>
<td>CAG = Gln</td>
<td></td>
</tr>
<tr>
<td>CAU = His</td>
<td>AAC = Asp</td>
<td>AAA = Lys</td>
<td>AGU = Ser</td>
</tr>
<tr>
<td>CAC = His</td>
<td>AAT = Asp</td>
<td>AAA = Lys</td>
<td>AGC = Ser</td>
</tr>
<tr>
<td>CAA = Gln</td>
<td>AAC = Asp</td>
<td>AAA = Lys</td>
<td>AGC = Ser</td>
</tr>
<tr>
<td>CGA = Arg</td>
<td>AAG = Lys</td>
<td>AGG = Arg</td>
<td></td>
</tr>
<tr>
<td>CGC = Arg</td>
<td>AAG = Lys</td>
<td>AGG = Arg</td>
<td></td>
</tr>
<tr>
<td>CGG = Arg</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G</td>
<td>C</td>
<td>A</td>
<td>G</td>
</tr>
<tr>
<td>GUU = Val</td>
<td>GCC = Ala</td>
<td>GAU = Asp</td>
<td>GGU = Gly</td>
</tr>
<tr>
<td>GUC = Val</td>
<td>GCA = Ala</td>
<td>GAC = Asp</td>
<td>GGC = Gly</td>
</tr>
<tr>
<td>GUA = Val</td>
<td>GCG = Ala</td>
<td>GAA = Glu</td>
<td>GGA = Gly</td>
</tr>
<tr>
<td>GUG = Val</td>
<td>GAG = Glu</td>
<td>GAG = Glu</td>
<td>GGG = Gly</td>
</tr>
</tbody>
</table>
Open Reading Frames

We could identify coding regions by searching for Met and STOPs. Suppose we are examining:

```
ACGGTGTTGGGTAGTAGTAGAAGTAGTATAG
```

| Arg | Cys | Try | STOP | Cys | Arg | Ser | Met |

When is a base-triple truly a STOP codon?
We could identify coding regions by searching for Met and STOPs. Suppose we are examining:

ACG GTG TTG GTAGT GTAGA AGTATG A

When is a base-triple truly a STOP codon?
Open Reading Frames

We could identify coding regions by searching for Met and STOPs. Suppose we are examining:

```
ACGGTGTTGGGTAGTGTAGAAGTATGAGCTAGAAGTATGAT
```

Frame shifts can change the protein sequence being coded.
“ORF” Detection

• Codons must have a functional pattern in a coding region; in a random sequence how often would we see a STOP?

• Given a window of DNA in a genome, can we assess the likelihood that it is a coding region (given a particular frameshift)?

• In known genes, Arg is 12x more likely to be coded by CGC than AGG.
Using known mRNA, we can compute the likelihood that a stretch of DNA is coding (given the frame shift).
Using known mRNA, we can compute the likelihood that a stretch of DNA is coding (given the frame shift).

<table>
<thead>
<tr>
<th>U</th>
<th>C</th>
<th>A</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>UUU Phe</td>
<td>UCU Ser</td>
<td>UAU Tyr</td>
<td>UGU Cys</td>
</tr>
<tr>
<td>UUC Phe</td>
<td>UCC Ser</td>
<td>UAC Tyr</td>
<td>UGC Cys</td>
</tr>
<tr>
<td>UUA Leu</td>
<td>UCA Ser</td>
<td>UAA Stp</td>
<td>UGA Stp</td>
</tr>
<tr>
<td>UUG Leu</td>
<td>UCG Ser</td>
<td>UAG Stp</td>
<td>UGG Trp</td>
</tr>
<tr>
<td>CUU Leu</td>
<td>CCU Pro</td>
<td>CAU His</td>
<td>CGU Arg</td>
</tr>
<tr>
<td>CUC Leu</td>
<td>CCC Pro</td>
<td>CAC His</td>
<td>CGC Arg</td>
</tr>
<tr>
<td>CUA Leu</td>
<td>CCA Pro</td>
<td>CAA Gln</td>
<td>CGA Arg</td>
</tr>
<tr>
<td>CUG Leu</td>
<td>CCG Pro</td>
<td>CAG Gln</td>
<td>CGG Arg</td>
</tr>
<tr>
<td>AUU Ile</td>
<td>ACU Thr</td>
<td>AAU Asn</td>
<td>AGU Ser</td>
</tr>
<tr>
<td>AUC Ile</td>
<td>ACC Thr</td>
<td>AAC Asn</td>
<td>AGC Ser</td>
</tr>
<tr>
<td>AUA Ile</td>
<td>ACA Thr</td>
<td>AAA Lys</td>
<td>AGA Arg</td>
</tr>
<tr>
<td>AUG Met</td>
<td>ACG Thr</td>
<td>AAG Lys</td>
<td>AGG Arg</td>
</tr>
<tr>
<td>GUU Val</td>
<td>GCU Ala</td>
<td>GAU Asp</td>
<td>GGU Gly</td>
</tr>
<tr>
<td>GUC Val</td>
<td>GCC Ala</td>
<td>GAC Asp</td>
<td>GGC Gly</td>
</tr>
<tr>
<td>GUA Val</td>
<td>GCA Ala</td>
<td>GAA Glu</td>
<td>GGA Gly</td>
</tr>
<tr>
<td>GUG Val</td>
<td>GCG Ala</td>
<td>GAG Glu</td>
<td>GGG Gly</td>
</tr>
</tbody>
</table>
Suppose we have known codon frequencies $F^* = f_1^*, f_2^*, \ldots, f_{61}^*$. For our unknown sequence, calculate codon usages $F_0, F_1, F_2$ for each possible frame shift. Compute $\arg \max \delta(F_i, F^*)$, for an appropriately chosen cost function $\delta$ (Euclidean, KL-distance, etc).
Gene Splicing

Sharp and Roberts (1977) hybridized the mRNA for a viral protein to its corresponding “gene” and showed that transcription can be “spliced”.

So given a genomic sequence, we need to identify fragmented exonic components (with or without mRNA).
About 5% of a genomic sequence is exonic, while the rest is intronic (some say it is “junk”). Prokaryotes don’t have exons!
Splicing Signals

We can attempt to perform a “spliced alignment” by using a known homologous gene.

Statistical methods for gene detection attempt to detect the “transition” between splice sites by comparing the distributions of codons on either side of an AG or GT pair.
Given a DNA sequence $G$, a target sequence $T$, and a candidate set of exons $\mathcal{B}$, which chain of non-overlapping exons $\Gamma^*$ maximizes the alignment score $\text{cost}(\Gamma^*, T)$.
Suppose we had a candidate chain $\Gamma$ ending in a block $B$.

We want: $\Gamma^* = \arg \max_B S(\text{length}(B), \text{length}(T), B)$

We can find the optimal spliced alignment using dynamic programming. (How quickly?)
Gelfand et al.

\[ S(i, j, k) = \max_{\text{all chains } \Gamma \text{ containing block } B_k} s(\Gamma^*(i), T(j)). \]

\( S(i, j, k) \) can be easily computed by dynamic programming as described below.

Let \( B(i) = \{k: \text{last}(k) < i\} \) be the set of blocks ending (strictly) before position \( i \) in \( G \). The following recurrence computes \( S(i, j, k) \) for \( 1 \leq i \leq n \), \( 1 \leq j \leq m \), and \( 1 \leq k \leq b \):

\[
S(i, j, k) = \begin{cases} 
S(i - 1, j - 1, k) + \Delta(g_i, t_j), & \text{if } i \neq \text{first}(k) \\
S(i - 1, j, k) + \Delta_{\text{indel}}, & \text{if } i \neq \text{first}(k) \\
\max_{l \in B(\text{first}(k))} S(\text{last}(l), j - 1, l) + \Delta(g_i, t_j), & \text{if } i = \text{first}(k) \\
\max_{l \in B(\text{first}(k))} S(\text{last}(l), j, l) + \Delta_{\text{indel}}, & \text{if } i = \text{first}(k) \\
S(i, j - 1, k) + \Delta_{\text{indel}}. & \text{if } i = \text{first}(k)
\end{cases}
\]

[1]

After computing the three-dimensional table \( S(i, j, k) \), the score of the optimal spliced alignment can be found as

\[
\max_k S(\text{last}(k), m, k).
\]
Suppose that we associate blocks of length $k$ in $G$ with two possible states, “intron” and “exon”.

Starting with a prior and conditional likelihoods for each block, can we find the most likely set of exons for $G$?
Hidden Markov Models

We are given $\Pr[s_i]$ and $\Pr[s_i \mid s_{i-1}]$ as input.

Our goal is to find $\arg \max_{(s_1, s_2, \ldots, s_{\frac{m}{k}})} \Pr[s_1, s_2, \ldots, s_{\frac{m}{k}}]$

This can be done using the Viterbi algorithm, which is essentially a dynamic programming method.
Combinatorial Gene Regulation

- A differential gene expression (e.g., microarray, HTS) experiment showed that when gene X is knocked out, 20 other genes are not expressed.

- How can one gene have such drastic effects?
Regulatory Proteins

- Gene X encodes regulatory protein, a.k.a. a transcription factor (TF)

- The 20 unexpressed genes rely on gene X’s TF to induce transcription

- A single TF may regulate multiple genes
Regulatory Regions

- Every gene contains a regulatory region (RR) typically stretching 100-1000 bp upstream of the transcriptional start site.

- Located within the RR are the Transcription Factor Binding Sites (TFBS), also known as motifs, specific for a given transcription factor.

- TFs influence gene expression by binding to a specific location in the respective gene’s regulatory region - TFBS.
Transcription Factor Binding Sites

• A TFBS can be located anywhere within the Regulatory Region.

• TFBS may vary slightly across different regulatory regions since non-essential bases could mutate
Motifs and Transcriptional Start Sites

ATCCCG  gene
TTCCGG  gene
ATCCCG  gene
ATGCCG  gene
ATGCCC  gene
Transcription Factors and Motifs

motif

Transcription Factor

No Terminator

TRANSCRIPTION BEGINS
Motif Logo

- Motifs can mutate on non important bases
- The five motifs in five different genes have mutations in position 3 and 5
- Representations called motif logos illustrate the conserved and variable regions of a motif

TGGGGGA
TGAGAGA
TGAGGGGA
TGAGAGA
TGAGGGGA
Motif Logos: An Example

(http://www-lmmb.ncifcrf.gov/~toms/sequencelogo.html)
Identifying Motifs

- Genes are turned on or off by regulatory proteins.
- These proteins bind to upstream regulatory regions of genes to either attract or block an RNA polymerase.
- Regulatory protein (TF) binds to a short DNA sequence called a motif (TFBS).
- So finding the same motif in multiple genes’ regulatory regions suggests a regulatory relationship amongst those genes.
Identifying Motifs: Complications

• We do not know the motif sequence

• We do not know where it is located relative to the genes start

• Motifs can differ slightly from one gene to the next

• How to discern it from “random” motifs?
DNA Motifs

Small conserved regions of DNA can regulate transcription, but how do we find them?

Given a set of $n$ sequences, can we find a shared substring of length $k$?
Given a set of $n$ sequences, can we find a shared substring of length $k$?
DNA Motifs

Small conserved regions of DNA can regulate transcription, but how do we find them?

CCTGATAGACGCTATCTGGCTATCC\textcolor{red}{AGGTACTT}AGGTCTCTCTGTGC\textcolor{orange}{GAATCTATAGCGT}
AGTACTGCTGTACATTTGAT\textcolor{red}{CCATACGT}ACACCGGCAACCTGAAAC
ACG\textcolor{red}{CCGTGTTAC}GTACATTTGAT\textcolor{red}{CCATACGT}ACACCGGCAACCTGAAACGCTCACG
ACGTACATTTGAT\textcolor{red}{CCATACGT}ACACCGGCAACCTGAAACGCTCACG

AGGTACTT
CCATACGT
ACGT\textcolor{red}{TAGT}
ACG\textcolor{red}{TAGT}
ACGTCC\textcolor{red}{AT}
CCGTACCGG
DNA Motifs

Given a set of $n$ sequences of length $m$, what is the consensus substring of length $k$ ?

CCTGATAGACGCTATCTGGCTATCCAGGTACTTAGGTCCCTCTGTGCGAATCTATATGCCTG
AGTACTGGTGTAACATTTTGATCCATACGTACACCGGCAACCTGAAACAAACGCTCAGAACG
AAACGTATTGCACCCCTCTTTCTTCGTGGCTCTGGCCACACGGCGGTGAGCTAGTATAAGA
GTAAGTCATAGCTGTAACATTTACCTGCCACCATTTATTTACATCTTTACGTCCATATAC
ACGCTCATGGGCGGGGTATGCGTTTTGGTCGTCGTACGCCTCGATCGTTAGGTACCCTC
CCATACGTACGTACGTACGTACGTACGT

The consensus substring is: ACGTCCATATAC

weblogo.berkeley.edu
DNA Motifs

Given a set of \( n \) sequences of length \( m \), what is the consensus substring of length \( k \) ?

If we knew where each motif started, then we just need to compute:

\[
score(s_1, s_2, \ldots, s_n) = \sum_{i=0}^{k-1} \text{best}(\{g_j[s_j + i] \mid j = 1, 2, \ldots, n\})
\]

What if we tried all possible starting points?

\((m - k)^n\) possible pairings of substrings!
A Motif Finding Analogy

• The Motif Finding Problem is similar to the problem posed by Edgar Allan Poe (1809 – 1849) in his *Gold Bug* story
The Gold Bug Problem

- Given a secret message:

```plaintext
53++!305) 6*;4826)4+.4+);806*;48!8`60)85;)8*:+*8!83(88)5*!;
46(;88*96*?;8)++(;485);5!*2;++(;4956*2(5*-4)8`8*;4069285);)6
!8)4++;1(+9;48081;8:8+1;48!85;4)485!528806*81(+9;48;(88;4(+83
4;48)4++;161;:188;+?;
```

- Decipher the message encrypted in the fragment
Hints for The Gold Bug Problem

• Additional hints:
  • The encrypted message is in English
  • Each symbol corresponds to one letter in the English alphabet
  • No punctuation marks are encoded
The Gold Bug Problem: Symbol Counts

- Naive approach to solving the problem:
  - Count the frequency of each symbol in the encrypted message
  - Find the frequency of each letter in the alphabet in the English language
  - Compare the frequencies of the previous steps, try to find a correlation and map the symbols to a letter in the alphabet
Symbol Frequencies in the Gold Bug Message

**Gold Bug Message:**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>34</td>
</tr>
<tr>
<td>;</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
</tr>
<tr>
<td>)</td>
<td>16</td>
</tr>
<tr>
<td>+</td>
<td>15</td>
</tr>
<tr>
<td>*</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>(</td>
<td>9</td>
</tr>
<tr>
<td>!</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>:</td>
<td>3</td>
</tr>
<tr>
<td>?</td>
<td>2</td>
</tr>
<tr>
<td>`</td>
<td>1</td>
</tr>
<tr>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>]</td>
<td>1</td>
</tr>
<tr>
<td>.</td>
<td>1</td>
</tr>
</tbody>
</table>

**English Language:**

```
et a o i n s r h l d c u m f p g w y b v k x j q z
```

Most frequent → Least frequent
The Gold Bug Message Decoding: First Attempt

• By simply mapping the most frequent symbols to the most frequent letters of the alphabet:

sfiilfcsoorntaeuroaikoiotecrntaeleyrcooestvenpinelelefheeesnltaarhteenmrunwteonihtaesotsnlupnihtamrsnuhsnbaoeyentacrmuesotorl
eoaiitdhimtaecedtepeidtaelelestaoaeslsueecrnedhimtaetheetahiwfa
taeoaitdrdtpdeetiwt

• The result does not make sense
The Gold Bug Problem: l-tuple count

- A better approach:
  - Examine frequencies of l-tuples, combinations of 2 symbols, 3 symbols, etc.
  - “The” is the most frequent 3-tuple in English and “;48” is the most frequent 3-tuple in the encrypted text
  - Make inferences of unknown symbols by examining other frequent l-tuples
The Gold Bug Problem: the ;48 clue

• Mapping “the” to “;48” and substituting all occurrences of the symbols:

53++!305))6*the26)h+.h+te06*the!e`60)e5t]e*:++e!e3(ee)5*!t
h6(tee*96*?te)+the5)t5*!2:*+(th956*2(5*h)e`e*th0692e5)t)6!e
h++t1(+9the0elte:e+1the!e5th)he5!52ee06*e1(+9thet(eeth(+?3ht
he)h+t161t:1eet+?t
The Gold Bug Message Decoding: Second Attempt

• Make inferences:

53++!305)6*the26)h+.)h+)te06*the!e`60)\e5t]e*:+*e!e3(ee)5*!t
h6(tee*96*?te)**+(the5)\t5*!2:**+(th956*2(5*h)e`e*th0692e5)\t)6!e
)h++t1(+9the0e1te:e+1the!e5th)he5!52ee06*e1(+9thet(eeth(+/3ht
he)h+t161t:leet+?t

• “thet(ee)” most likely means “the tree”
• Infer “(“ = “r”

• “th(+?3h)” becomes “thr+?3h”
• Can we guess “+” and “?”?
The Gold Bug Problem: The Solution

After figuring out all the mappings, the final message is:

AGOODGLASSINTHEBISHOPSHOSTELINTHEDEVILSSEATWENYONEDEGREE
ESANDTHIRTEENMINUTENORtheEASTANDBYNORTHMAINBRANCHSEVENT
HLIMBEASTSIDEHOOTFROMTHELEFTYEOTHEDEATHSHEAdABEELEINE
FROMTHETREE THROUGH THE SHOT FIFTY FEET OUT
The Solution (cont’d)

- Punctuation is important:

A GOOD GLASS IN THE BISHOP’S HOSTEL IN THE DEVIL’S SEA,
TWENY ONE DEGREES AND THIRTEEN MINUTES NORTHEAST AND BY NORTH,
MAIN BRANCH SEVENTH LIMB, EAST SIDE, SHOOT FROM THE LEFT EYE OF
THE DEATH’S HEAD A BEE LINE FROM THE TREE THROUGH THE SHOT,
FIFTY FEET OUT.
Solving the Gold Bug Problem

• Prerequisites to solve the problem:

• Need to know the relative frequencies of single letters, and combinations of two and three letters in English

• Knowledge of all the words in the English dictionary is highly desired to make accurate inferences
DNA Motifs

Given a set of \( n \) sequences of length \( m \), what is the **consensus** substring of length \( k \)?

If we knew where each motif started, then we just need to compute:

\[
\text{score}(s_1, s_2, \ldots, s_n) = \sum_{i=0}^{k-1} \text{best}(\{g_j[s_j+i] \mid j = 1, 2, \ldots, n\})
\]

What if we tried all possible \( k \)-mers?

\[
4^k \cdot n(m - k) \text{ comparisons}
\]
Branch and Bound

Can we improve our sequential search strategy?

What if we looked at shorter segments?

If a $k$-mer has a “bad” prefix, then we can eliminate all $k$-mers with that prefix.
We can draw the search as a tree where each level corresponds to a prefix of the $k$-mer we want.

We must establish bounds on the “score” of a motif given its prefix.

**Idea:** If a particular $k$-mer cannot be improved upon by a different prefix, eliminate that subtree from the search.