Pairwise Alignment and Database Searching

Anders Gorm Pedersen
Molecular Evolution Group
Center for Biological Sequence Analysis
Sequences are related

- Darwin: all organisms are related through descent with modification
- => Sequences are related through descent with modification
- => Similar molecules have similar functions in different organisms

Phylogenetic tree based on ribosomal RNA: three domains of life
Sequences are related, II

Phylogenetic tree of globin-type proteins found in humans
Why compare sequences?

• Determination of evolutionary relationships

Protein 1: binds oxygen

Sequence similarity

Protein 2: binds oxygen?
Dotplots: visual sequence comparison

1. Place two sequences along axes of plot

2. Place dot at grid points where two sequences have identical residues

3. Diagonals correspond to conserved regions
### Pairwise alignments

43.2% identity;  
Global alignment score: 374

<table>
<thead>
<tr>
<th></th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>60</td>
<td>70</td>
<td>80</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>120</td>
<td>130</td>
<td>140</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Percent identity is not a good measure of alignment quality

100.000% identity in 3 aa overlap

SPA
:::
SPA
Pairwise alignments: alignment score

43.2% identity;

Global alignment score: 374
Alignment scores: match vs. mismatch

Simple scoring scheme (too simple in fact...):

Matching amino acids: 5
Mismatch: 0

Scoring example:

K A W S A D V
: : : : : :
K D W S A E V
5+0+5+5+5+0+5 = 25
Pairwise alignments: conservative substitutions

43.2% identity;      Global alignment score: 374

10  20  30  40  50
alpha  V-LS-PAD-KTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTTKTYFPFH-DLS------HGSA
beta  VHL-TPEEK-SAVTALWGKV--NVDEVGGEALGRLLVYYPWTQRFESFGDLSTPDAVMGNP

60  70  80  90  100  110
alpha  QVKGHGKKVADALTNAVAHVDDMPNALSALSDLHAAHKLHRVPVNFKLSSHCLLVTLAAHL
beta  KVKAHGKKVGLGAFSDGLAHLDNLKGFATLSELHCDDKHLVDPENFRLLGNVLVCVLAHHF

120  130  140
alpha  PAEFTPAVHASLDKFLASVSTVLTSKYR
beta  GKEFTPPVQAAYQKVAGVANALAHKYH
Amino acid properties

Serine (S) and Threonine (T) have similar physicochemical properties

Aspartic acid (D) and Glutamic acid (E) have similar properties

=> Substitution of S/T or E/D occurs relatively often during evolution

=> Substitution of S/T or E/D should result in scores that are only moderately lower than identities
# Protein substitution matrices

**BLOSUM50 matrix:**

- Positive scores on diagonal (identities)
- Similar residues get higher (positive) scores
- Dissimilar residues get smaller (negative) scores

|   | A | R | N | D | C | Q | E | G | H | I | L | K | M | F | P | S | T | W | Y | V |
| A | 5 | -2 | 7 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| R | -2 | 7 | -1 | -1 | 7 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| N | -1 | -1 | 7 | -2 | -2 | 2 | 8 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| D | -2 | -2 | 2 | 8 | -1 | -4 | -2 | -4 | 13 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| C | -1 | -4 | -2 | -4 | 13 | -1 | -4 | -2 | -4 | -4 | 2 | 8 | -1 | 3 | 0 | -1 | -3 | 2 | 1 | -2 | 0 | -3 | -3 | 6 |
| Q | -1 | 1 | 0 | 0 | -3 | 7 | -1 | 0 | 0 | 2 | -3 | 2 | 6 | 0 | -3 | 0 | -1 | -3 | -2 | -3 | 8 |   |   |   |
| E | -1 | 0 | 0 | 2 | -3 | 2 | 6 | -2 | 0 | 1 | -1 | -3 | 1 | 0 | -2 | 10 |   |   |   |   |   |   |   |   |
| G | 0 | -3 | 0 | -1 | -3 | -2 | -3 | 8 | -2 | 0 | 1 | -1 | -3 | 1 | 0 | -2 | 10 |   |   |   |   |   |   |   |   |
| H | -2 | 0 | 1 | -1 | -3 | 1 | 0 | -2 | 10 | -1 | -4 | -3 | -4 | -2 | -3 | -4 | -4 | 13 | 5 |   |   |   |   |   |   |
| I | -2 | -3 | -4 | -2 | -2 | -3 | -4 | -3 | 2 | 5 | -2 | -3 | -4 | -2 | -2 | -3 | -4 | -3 | 2 | 5 |   |   |   |   |   |   |
| L | -1 | 3 | 0 | -1 | -3 | 2 | 1 | -2 | 0 | -3 | -3 | 6 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| K | -1 | 2 | 1 | -2 | 0 | -3 | -3 | 6 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| M | -1 | -2 | -2 | -4 | -2 | 0 | -2 | -3 | 1 | 2 | 3 | -2 | 7 | -3 | -3 | -4 | -5 | -2 | -2 | -3 | -4 | -1 | -3 | -4 | 10 |
| F | -3 | -3 | -4 | -5 | -2 | -4 | -3 | -4 | -1 | 0 | 1 | -4 | 0 | 8 | -3 | -3 | -4 | -5 | -5 | -1 | -3 | -3 | -2 | -3 | -1 | 15 |
| P | -1 | -3 | -2 | -1 | -4 | -1 | -1 | -3 | -4 | -1 | -2 | -2 | -3 | -4 | -1 | -3 | -4 | -1 | -3 | -4 | 10 |   |   |   |   |   |   |
| S | 1 | -1 | 1 | 0 | -1 | 0 | -1 | -3 | -3 | 0 | -2 | -3 | -1 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| T | 0 | -1 | 0 | -1 | -1 | -1 | -2 | -2 | -1 | -1 | -1 | -1 | -2 | -1 | 2 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| W | -3 | -3 | -4 | -5 | -5 | -1 | -3 | -3 | -3 | -2 | -3 | -1 | 1 | -4 | -4 | -3 | 15 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Y | -2 | -1 | -2 | -3 | -1 | -2 | -3 | 2 | -1 | -1 | -2 | 0 | 4 | -3 | -2 | -2 | 2 | 8 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| V | 0 | -3 | -3 | -4 | -1 | -3 | -3 | -4 | -4 | 4 | 1 | -3 | 1 | -1 | -3 | -2 | 0 | -3 | -1 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

**BLOSUM50 Matrix Explanation:**

- **Positive scores on diagonal** (identities)
- **Similar residues get higher (positive) scores**
- **Dissimilar residues get smaller (negative) scores**
Pairwise alignments: insertions/deletions

43.2% identity;  
Global alignment score: 374

10    20    30    40    50
alpha  V-LS\text{PADKTNVKA}\text{AWGKVGA}\text{HAGEYGAEE}\text{ALERMFLSFP}TTKTYFPHF-DLS-----HGSA
beta   VHL\text{TPEEKSAVTALWGKV--NVDEVGGEALGRLLV}VYPWTQRFFESFGDLST\text{TPDAVMGNP}

10    20    30    40    50    60    70    80    90   100   110
alpha  QVK\text{GHGKKVADALTNAVHAVDDMPNALSALSD}L\text{HAKLRLVDPVNFKLLSHCLLVTЛАHL}
beta   KVK\text{AHGKKVLGAFSDGLAHLDNLGKFATLSELHCDKLHVDPENFRLLGNVLVCLAHHF}

120   130   140
alpha  P\text{AEFTPAVHASL}DKFL\text{ASVSTVLTSKYR}
beta   G\text{KEFTP}PQ\text{AAYQKVAGVANALAHKYH}
Alignment scores: insertions/deletions

Affine gap penalties:
Multiple insertions/deletions may be one evolutionary event =>
Separate penalties for gap opening and gap elongation
Compute 4 alignment scores: two different alignments using two different alignment matrices (and the same gap penalty system)

Score 1: Alignment 1 + BLOSUM-50 matrix + gaps
Score 2: Alignment 1 + BLOSUM-Trp matrix + gaps
Score 3: Alignment 2 + BLOSUM-50 matrix + gaps
Score 4: Alignment 2 + BLOSUM-Trp matrix + gaps

Note: fake matrix constructed for pedagogic purposes.
<table>
<thead>
<tr>
<th></th>
<th>Alignment 1</th>
<th>Alignment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOSUM-50</td>
<td>38</td>
<td>51</td>
</tr>
<tr>
<td>BLOSUM-Trp</td>
<td>118</td>
<td>91</td>
</tr>
</tbody>
</table>
### Protein substitution matrices

|     | A  | R  | N  | D  | C  | Q  | E  | G  | H  | I  | L  | K  | M  | F  | P  | S  | T  | W  | Y  | V  |   |
|-----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|
| A   | 5  | -2 | -1 | -2 | -1 | -1 | 0  | -3 | -2 | -1 | -4 | -2 | -2 | -3 | -4 | -1 | -3 | -4 | -4 | -4 | 5  |
| R   | -2 | 7  | 0  | 0  | 0  | 0  | 1  | -1 | -1 | -3 | -2 | -3 | -3 | -2 | -1 | -3 | -2 | -1 | -1 | -1 | 5  |
| N   | -1 | -1 | 7  | 8  | 13 | 7  | 2  | -2 | -1 | -3 | -2 | -3 | -2 | -2 | -4 | -1 | -3 | -2 | -2 | -2 | 5  |
| D   | -2 | -2 | 2  | 2  | 2  | 2  | 1  | -1 | 0  | -2 | 0  | -2 | 0  | -3 | -2 | -2 | -3 | -3 | -2 | -1 | 5  |
| C   | -1 | -4 | -2 | -4 | -1 | -3 | 1  | 0  | -2 | 10 | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  |
| Q   | -1 | 1  | 0  | 0  | -3 | 7  | -1 | 0  | 2  | -3 | 2  | 6  | 5  | 4  | 3  | 2  | 1  | 0  | 0  | -1 | 7  |
| E   | 0  | -3 | 0  | -1 | -3 | -2 | -1 | -3 | -3 | -3 | -3 | 6  | 6  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  |
| G   | -2 | 0  | 1  | -1 | -3 | -1 | -3 | -1 | -2 | 0  | -3 | -3 | 8  | 8  | 8  | 8  | 8  | 8  | 8  | 8  | 8  |
| H   | -1 | -4 | -3 | -4 | -2 | -3 | -4 | -4 | -4 | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  | 5  |
| I   | -2 | -3 | -4 | -4 | -2 | -3 | -4 | -4 | -4 | 2  | 5  | 2  | 5  | 2  | 5  | 2  | 5  | 2  | 5  | 2  | 5  |
| L   | -1 | 3  | 0  | -1 | -3 | 2  | 1  | -2 | 0  | -3 | -3 | 6  | 6  | 6  | 6  | 6  | 6  | 6  | 6  | 6  | 6  |
| K   | -1 | -2 | -2 | -4 | -2 | 0  | -2 | -3 | -1 | 2  | 3  | 2  | 7  | 8  | 8  | 8  | 8  | 8  | 8  | 8  | 8  |
| M   | -1 | -2 | -2 | -4 | -2 | 0  | -2 | -3 | -1 | 2  | 3  | 2  | 7  | 8  | 8  | 8  | 8  | 8  | 8  | 8  | 8  |
| F   | -3 | -3 | -4 | -5 | -2 | -4 | -3 | -4 | -1 | 0  | 1  | -4 | 0  | 8  | 8  | 8  | 8  | 8  | 8  | 8  | 8  |
| P   | -1 | -3 | -2 | -1 | -4 | -1 | -1 | -2 | -2 | -3 | -4 | -1 | -3 | -4 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |
| S   | 1  | -1 | 1  | 0  | -1 | 0  | -1 | 0  | -1 | -3 | -3 | 0  | -2 | -3 | -1 | 5  | 5  | 5  | 5  | 5  | 5  | 5  |
| T   | 0  | -1 | 0  | -1 | -1 | -1 | -2 | -2 | -1 | -1 | -1 | -1 | -1 | -2 | -1 | 2  | 5  | 5  | 5  | 5  | 5  | 5  |
| W   | -3 | -3 | -4 | -5 | -5 | -1 | -3 | -3 | -3 | -2 | -3 | -1 | 1  | -4 | -4 | -3 | 15 | 15 | 15 | 15 | 15 | 15 | 15 |
| Y   | -2 | -1 | -2 | -3 | -3 | -1 | -2 | -3 | 2  | -1 | -1 | -2 | 0  | 4  | -3 | -2 | 2  | 8  | 8  | 8  | 8  | 8  |
| V   | 0  | -3 | -3 | -4 | -1 | -3 | -3 | -4 | -4 | 4  | 1  | -3 | 1  | -1 | -3 | -2 | 0  | -3 | -1 | 5  | 5  | 5  | 5  |

#### BLOSUM50 matrix:

- Positive scores on diagonal (identities)
- Similar residues get higher (positive) scores
- Dissimilar residues get smaller (negative) scores
Protein substitution matrices: different types

- **Identity matrix**
  (match vs. mismatch)

- **Genetic code matrix**
  (how similar are the codons?)

- **Chemical properties matrix**
  (use knowledge of physicochemical properties to design matrix)

- **Empirical matrices**
  (based on observed pair-frequencies in hand-made alignments)
  - PAM series
  - BLOSUM series
  - Gonnet
Estimation of the BLOSUM 50 matrix

- BLOSUM matrices are computed based on gap-free alignments in the so-called BLOCKS database. BLOSUM 50 is computed by comparing sequences that are less than 50% identical. BLOSUM 80 is computed from sequences less than 80% identical, etc.

- All pairs of sequences in a block are compared, and the observed pair frequencies are noted (e.g., A aligned with A makes up 1.5% of all pairs. A aligned with C makes up 0.01% of all pairs, etc.)

- Expected pair frequencies are computed from single amino acid frequencies. (e.g., \( f_{A,C} = f_A \times f_C = 7\% \times 3\% = 0.21\% \)).

- For each amino acid pair the substitution scores are essentially computed as:

\[
\log \left( \frac{\text{Pair-freq(obs)}}{\text{Pair-freq(expected)}} \right) = S_{A,C} = \log \left( \frac{0.01\%}{0.21\%} \right) = -1.3
\]
Optimal alignment:

alignment having the highest possible score given a substitution matrix and a set of gap penalties
Pairwise alignment: the problem

The number of possible pairwise alignments increases explosively with the length of the sequences:

Two protein sequences of length 100 amino acids can be aligned in approximately $10^{60}$ different ways

Time needed to test all possibilities is same order of magnitude as the entire lifetime of the universe.
Pairwise alignment: the solution

"Dynamic programming"
(the Needleman-Wunsch algorithm)
Alignment depicted as path in matrix

TCGCA → TCGCA
TC-CA → T-CCA
Dynamic programming: computation of scores

Any given point in matrix can only be reached from three possible previous positions (you cannot “align backwards”).

=> Best scoring alignment ending in any given point in the matrix can be found by choosing the highest scoring of the three possibilities.
Dynamic programming: computation of scores

Any given point in matrix can only be reached from three possible positions (you cannot “align backwards”).

=> Best scoring alignment ending in any given point in the matrix can be found by choosing the highest scoring of the three possibilities.

\[
\text{score}(x,y) = \max \left\{ \text{score}(x,y-1) - \text{gap-penalty} \right\}
\]
Any given point in matrix can only be reached from three possible positions (you cannot “align backwards”).

=> Best scoring alignment ending in any given point in the matrix can be found by choosing the highest scoring of the three possibilities.

\[
\text{score}(x, y) = \max \begin{cases} 
\text{score}(x, y-1) - \text{gap-penalty} \\
\text{score}(x-1, y-1) + \text{substitution-score}(x, y)
\end{cases}
\]
Dynamic programming: computation of scores

Any given point in matrix can only be reached from three possible positions (you cannot “align backwards”).

=> Best scoring alignment ending in any given point in the matrix can be found by choosing the highest scoring of the three possibilities.

\[
\text{score}(x,y) = \max \begin{cases} 
\text{score}(x,y-1) - \text{gap-penalty} \\
\text{score}(x-1,y-1) + \text{substitution-score}(x,y) \\
\text{score}(x-1,y) - \text{gap-penalty} 
\end{cases}
\]
Dynamic programming: computation of scores

Any given point in matrix can only be reached from three possible positions (you cannot “align backwards”).

=> Best scoring alignment ending in any given point in the matrix can be found by choosing the highest scoring of the three possibilities.

Each new score is found by choosing the maximum of three possibilities. For each square in matrix: keep track of where best score came from.

Fill in scores one row at a time, starting in upper left corner of matrix, ending in lower right corner.

\[
\text{score}(x, y) = \max \left\{ \text{score}(x, y-1) - \text{gap-penalty}, \right.
\]
\[
\text{score}(x-1, y-1) + \text{substitution-score}(x, y), \right.
\]
\[
\text{score}(x-1, y) - \text{gap-penalty} \right\}
\]
Dynamic programming: example

\[ a[i,j] = \max \begin{cases} a[i-1,j-1] - 2 \\ a[i-1,j] + p(i,j) \\ a[i,j-1] - 2 \end{cases} \]

Gaps: -2
Dynamic programming: example

\[ a[i,j] = \max \begin{cases} 
  a[i-1,j-1] + p(i,j) \\
  a[i-1,j] - 2 \\
  a[i,j-1] - 2 
\end{cases} \]
Dynamic programming: example

\[ a[i, j] = \max \begin{cases} a[i, j-1] - 2 \\ a[i-1, j-1] + p(i, j) \\ a[i-1, j] - 2 \end{cases} \]
Dynamic programming: example

![Diagram of dynamic programming example]
Dynamic programming: example

Diagram showing the dynamic programming process with a sequence of characters T, C, G, C, and A, and their corresponding values at each step.
Dynamic programming: example

\[
a[i,j] = \max \begin{cases} 
a[i,j-1] -2 \\
a[i-1,j-1] + p(i,j) \\
a[i-1,j] -2
\end{cases}
\]
Dynamic programming: example

![Dynamic programming example diagram](image)
Dynamic programming: example

\[
\begin{array}{cccccccc}
 & & & & t[j] & & & \\
0 & 0 & -2 & -4 & -6 & -8 & -10 & \\
1 & -2 & 1 & -1 & -3 & -5 & -7 & \\
2 & -4 & -1 & 2 & 0 & -2 & -4 & \\
3 & -6 & -3 & 0 & 1 & 1 & -1 & \\
4 & -8 & -5 & -2 & -1 & 0 & 2 & \\
\end{array}
\]

\[
\begin{align*}
T & \quad C & \quad G & \quad C & \quad A \\
\vdots & \quad \vdots & \quad \vdots & \quad \vdots & \quad \vdots \\
T & \quad C & \quad - & \quad C & \quad A \\
\end{align*}
\]

\[
\frac{1 + 1 - 2 + 1 + 1}{1 + 1 - 2 + 1 + 1} = 2
\]
Global versus local alignments

Global alignment: align full length of both sequences. (The “Needleman-Wunsch” algorithm).

Local alignment: find best partial alignment of two sequences (the “Smith-Waterman” algorithm).
Local alignment overview

- The recursive formula is changed by adding a fourth possibility: zero. This means local alignment scores are never negative.

\[
\begin{align*}
score(x,y) &= \max \begin{cases}
score(x,y-1) - \text{gap-penalty} \\
score(x-1,y-1) + \text{substitution-score}(x,y) \\
score(x-1,y) - \text{gap-penalty} \\
0
\end{cases}
\end{align*}
\]

- Trace-back is started at the highest value rather than in lower right corner
- Trace-back is stopped as soon as a zero is encountered
### Local alignment: example

```
+------------+------------+------------+------------+------------+------------+------------+------------+------------+------------+------------+------------+
|   H        |   E        |   A        |   G        |   A        |   W        |   G        |   H        |   E        |   E        |
|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|
| 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 0          | 0          | 0          | 5          | 0          | 0          | 5          | 0          | 0          | 0          | 0          | 0          |
| 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          | 0          |
| 0          | 0          | 0          | 2          | 0          | 0          | 20         | 12         | 4          | 0          | 0          | 0          |
| 0          | 0          | 0          | 10         | 2          | 0          | 0          | 0          | 12         | 18         | 22         | 14         |
| 0          | 0          | 2          | 16         | 8          | 0          | 0          | 0          | 4          | 10         | 18         | 28         |
| 0          | 0          | 8          | 21         | 13         | 5          | 0          | 4          | 10         | 20         | 20         | 27         |
| 0          | 0          | 6          | 13         | 18         | 12         | 4          | 0          | 4          | 16         | 26         |
|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|

AWGHE
AW-HE
```
Substitution matrices and sequence similarity

- Substitution matrices come as series of matrices calculated for different degrees of sequence similarity (different evolutionary distances).

- "Hard" matrices are designed for similar sequences
  - Hard matrices are designated by high numbers in the BLOSUM series (e.g., BLOSUM80)
  - Hard matrices yield short, highly conserved alignments

- "Soft" matrices are designed for less similar sequences
  - Soft matrices have low BLOSUM values (45)
  - Soft matrices yield longer, less well conserved alignments
Alignments: things to keep in mind

“Optimal alignment” means “having the highest possible score, given substitution matrix and set of gap penalties”.

This is NOT necessarily the biologically most meaningful alignment.

Specifically, the underlying assumptions are often wrong: substitutions are not equally frequent at all positions, affine gap penalties do not model insertion/deletion well, etc.

Pairwise alignment programs always produce an alignment - even when it does not make sense to align sequences.
BLAST

Anders Gorm Pedersen
&
Rasmus Wernersson
Database searching

Using pairwise alignments to search databases for similar sequences

Query sequence

Database
Most common use of pairwise sequence alignments is to search databases for related sequences. For instance: find probable function of newly isolated protein by identifying similar proteins with known function.

Most often, *local* alignment ("Smith-Waterman") is used for database searching: you are interested in finding out if ANY domain in your protein looks like something that is known.

Often, full Smith-Waterman is too time-consuming for searching large databases, so heuristic methods are used (fasta, BLAST).
## Database searching: heuristic search algorithms

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Uses heuristics to avoid calculating the full dynamic programming matrix</td>
<td>Uses rapid word lookup methods to completely skip most of the database entries</td>
</tr>
<tr>
<td>Speed up searches by an order of magnitude compared to full Smith-Waterman</td>
<td>Extremely fast</td>
</tr>
<tr>
<td></td>
<td>One order of magnitude faster than FASTA</td>
</tr>
<tr>
<td></td>
<td>Two orders of magnitude faster than Smith-Waterman</td>
</tr>
<tr>
<td>The statistical side of FASTA is still stronger than BLAST</td>
<td>Almost as sensitive as FASTA</td>
</tr>
</tbody>
</table>
BLAST flavors

- **BLASTN**
  - Nucleotide query sequence
  - Nucleotide database

- **BLASTP**
  - Protein query sequence
  - Protein database

- **BLASTX**
  - Nucleotide query sequence
  - Protein database
  - Compares all six reading frames with the database

- **TBLASTN**
  - Protein query sequence
  - Nucleotide database
  - "On the fly" six frame translation of database

- **TBLASTX**
  - Nucleotide query sequence
  - Nucleotide database
  - Compares all reading frames of query with all reading frames of the database
Searching on the web: BLAST at NCBI

Very fast computer dedicated to running BLAST searches

Many databases that are always up to date (e.g. NR and Human Genome)

Nice simple web interface

But you still need knowledge about BLAST to use it properly
When is a database hit significant?

- **Problem:**
  - Even unrelated sequences can be aligned (yielding a low score)
  - How do we know if a database hit is meaningful?
  - When is an alignment score sufficiently high?

- **Solution:**
  - Determine the range of alignment scores you would expect to get for random reasons (i.e., when aligning unrelated sequences).
  - Compare actual scores to the distribution of random scores.
  - Is the real score much higher than you’d expect by chance?
Distribution of random alignment scores

- Software simulation
Significance of alignment score expressed as E-value

Searching a database of **unrelated** sequences results in scores following an extreme value distribution.

The exact shape and location of the distribution depends on the exact nature of the database and the query sequence.
Significance of alignment score expressed as E-value

E-value: the number of random hits with score $\geq$ real score

Want E-values below 1 (the lower the better)
BLAST heuristics

- BLAST speeds up the search >100x by pre-screening the database sequences and only performing the full Dynamic Programming on "promising" sequences.

- Promising sequences: database sequences that have sub-strings ("words") which also occur in the query sequence (found rapidly using a so-called "suffix-tree")

- BLASTN and BLASTP use different criteria for overlap required for a sequence to be deemed promising
BLASTN

- **Heuristics:**
  - Perfect match “word” of at least size: 7, 11 (default) or 15.

- **Alignment matrix:**
  - Match: 1
  - Mismatch: -3

- **Notice:** All mismatches are equally penalized:
  - E.g. A:G == A:C == A:T
  - More advanced models for DNA evolution do exist.

Illustration:
- All sequences
- Subset to align
- Match >= word size
- Potential matches of length < word size (not seen by BLAST)
- Heuristics:
  - 2 x “Near match” within a window.
  - Default word length: 3 aa
  - Default window length: 40 aa

- Alignment matrix:
  - PAM and BLOSUM-series (default: BLOSUM 62)

- Notice: These alignment matrices incorporate knowledge about protein evolution.