Lecture 5: November,27, 2000

Lecturer: Racheli Zakarin *Scribe: Naomi Keren and Guy Kol*[1]

5.1 Biological Databases and Retrieval Systems

In recent years, biological databases have greatly developed, and became a part of the biologist's everyday toolbox (see eg. [4]). There are several reasons to search databases, for instance:

1. When obtaining a new DNA sequence, one needs to know whether it has already been deposited in the databanks fully or partially, or whether they contain any *homologous sequences* (sequences which are descended from a common ancestor).

2. Some of the databases contain annotation which has already been added to a specific sequence. Finding annotation for the searched sequence or its *homologous sequences* can enhance the progress of the research .

3. To find similar non-coding DNA stretches in the database: for instance Repeat elements or regulatory sequences.

4. There are other uses for specific purpose, like locating false priming sites for a set of PCR oligonucleotides.

5. Given a putative coding ORF, we can search for *homologous proteins* - proteins similar in their sequence and therefore also intheir presumed folding or structure or function.

5.1.1 Available Databases

What kinds of databases are there?

- There are two main classes of databases: DNA (nucleotide) databases and protein databases. The growth rate of DNA databases is much higher because most of the DNA is not coding for proteins and because DNA sequencing is the most prominent source of these data.

- The following graph illustrates the semi-exponential growth of DNA databases along the years.

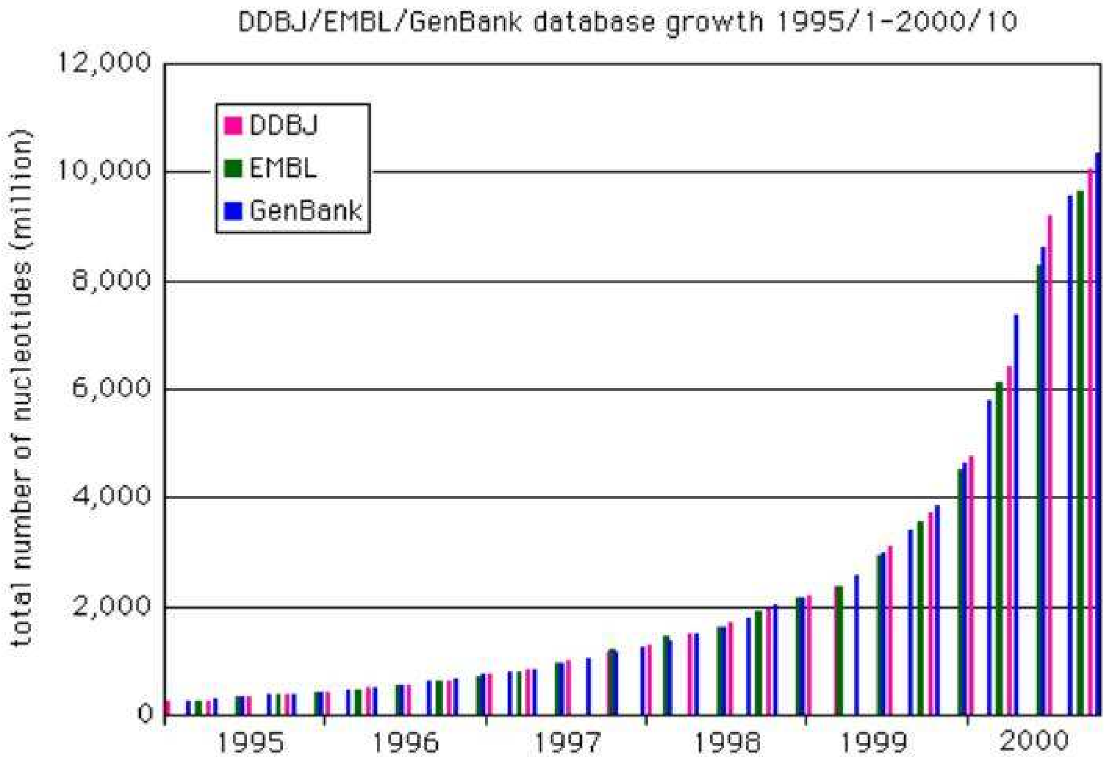

Figure 5.1: The DNA DB Growth

DNA Databases (Nucleotide Sequences)

- The large databases are: Genbank (US), EMBL (Europe - UK), DDBJ (Japan). These databases are quite similar regarding their contents and are updating one another periodically. This was is a result of the International Nucleotide Sequence Database Collaboration .

- Genomic databases: Human (GDB), mouse (MGB), yeast (SGB), etc...

- Special databases:

- ESTs (Expressed Sequence Tags): Short fragments of mRNA samples that were taken from a variety of tissues and organisms. These samples are amplified and sequenced. The sequencing is done in one read thus ESTs are a non-accurate source of information. There are about 6 million sequenced ESTs (more then 1/3 clonned from human) .

- STSs (Sequence-Tagged Sites): Short genomic samples that serves as a genomic markers.

- HTGS (High Throughput Genomic Sequences): Sequences obtained in the course of sequencing the whole genome. The records of this databases are classified according to their phase 3 towards sequence completion. Phase 0 - Single-few pass reads of a single clone(not contigs). Phase 1 - Unfinished, may be unordered, unoriented contigs, with gaps. Phase 2 - Unfinished, ordered, oriented contigs, with or without gaps. Phase 3 - Finished, no gaps (with or without annotations).

Database Searching of Proteins (Amino Acid Sequence)

- The large databases are: Swiss-Prot (high level of annotation), PIR (protein identification resource).

- Translated databases, like SPTREMBL (translated EMBL), GenPept (translation of coding regions in GenBank) .

- Special databases, like PDB (proteins with their resolved 3D structure).

5.1.2 How to Perform Database-Searching?

As the amount of biological relevant data is increasing so rapidly, knowing how to access and search this information is essential. The two main ways of searching are: Text based searched, quering the annotation and Sequence based.

Text Based Searching

There are three data retrieval systems of particular relevance to molecular biologist: Sequence Retrieval System (SRS), Entrez, DBGET.These systems allow text searching in a multitude of molecular biology database and provide links to relevant information for entries that match the search criteria. The three systems differ in the databases they search and the links they have to other information.

SRS The Sequence Retrieval System is a homogeneous interface to over 80 biological databases that had been developed at the European Bioinformatics Institute (EBI) at Hinxton, UK (see also SRS help [9]). It includes databases of sequences, metabolic pathways, transcription factors, application results (like BLAST, SSEARCH, FASTA), protein 3-D structures, genomes, mappings, mutations, and locus specific mutations.
The web page listing all the databases contains a link to a description page about each database including the date on which it was last updated. You select one or more of the databases to search before entering your query.
After getting results you choose an alignment algorithm (like CLUSTALW, PHYLIP) enter parameters, and run it.
The SRS [8] is highly recommended for use.

Entrez [18] is a molecular biology database and retrieval system, developed by the National Center for Biotechnology information (NCBI) (see Entrez help [19]). It is entry point for exploring distinct but integrated databases. Of the three text-based database systems, Entrez is the easiest to use, but also offers more limited information to search.

DBGET [13] is an integrated database retrieval system, developed at the university of Tokyo (see DBGET help [14]). Provides access to 20 databases, one at a time. Having more limited options, the DBGET is less recommended than the two others.

Sequence Based Searching

DNA is made of 4 nucleotides : A,C,G,T, while proteins are build out of 20 amino acids which means that aligning two unrelated DNA sequences will result in 25% of random matching. Using proteins will result in weaker random similarity and thus fewer false positives. A major issue of concerns is DNA vs. protein searches: a coding nucleotide sequence, can be translated into a protein sequence. (The other direction is, of-course, ambiguous, because the genetic code is degenerated.) so suppose we have a nucleotide sequence. Should we search the DNA databases only? Or should we translate it into a protein and search protein databases? In the other hand, translating causes loss of information but on a second thought protein sequences are more evolutionary conserved than DNA sequences.

What about very different DNA seqeunces that code for similar protein sequence? we would like to find those too. It's better to use the protein for searching in this case too. Usually, we should use proteins for database similarity searches when possible.The reasons for this conclusion are:

- When comparing DNA sequences, we get significantly more random matches than we get with proteins.
- The DNA databases are much larger, and grow faster than Protein databases. A larger database means more random hits.
- For DNA we usually use identity matrices while for protein we have more sensitive matrices like PAM and BLOSUM, which allow better search results.
- Conservation at the protein level is higher than at the DNA level.

As stated, a primary goal of sequence search is to find a sequence which is seams homologous to the query sequence, such a homologous sequence shares sequence similarity with the query sequence. The similarity is derived from common ancestry and conservation throughout evolution. Homologous proteins are similar in their structure. This is the basis for *homology modeling* structure determination through the structure of similar proteins.

The main goal in searching is finding relevant information and avoiding non relevant information, therefore define:

- *Sensitivity*: The ability to detect "true positive" matchs . The most sensitive search finds all true matchs, but might have lots of "false positives".
- *Specificity*: The ability to reject "false positive" matches. The most specific search will return only true matches, but might have lots of "false negatives".

  When one chooses which algorithm to use, there is a trade off between these two figures of merit. It is quiet trivial to create an algorithm which will optimize one of these properties, the problem is to create Sequenced-Based algorithm that will perform well with respect to both of them.

5.2 Tools for Searching

There are three main search tools: FastA (better for nucleotides than for proteins), BLAST (better for proteins than for nucleotides) and SW-search (more sensitive than FastA or BLAST).

5.2.1 FastA

FastA is a sequence comparison software that uses the method of Pearson and Lipman [5]. The basic FastA algorithm assumes a query sequence and a database over the same alphabet. It searches a DNA sequence in a DNA database or a protein sequence in a protein database. Practically, FastA is a family of programs, allowing also queries of DNA vs. a protein database, or vice versa. In these variants there is further distinction, which regards the location of gaps: one may assume that gaps occur only in the codon frames corresponding to amino-acid insertion; alternatively, one can assume gap location to be arbitrary, accounting for insertion/deletion of nucleotides. The different FastA variants are summarized in the following table:

FastA variant	Query sequence	Database sequence	Gap location
FastA	DNA	DNA	Not an issue
	Protein	Protein	
FastX	DNA	Protein	Between codons
FastY			Arbitrary
tFastA	Protein	DNA	Not an issue
tFastX	Protein	Translated DNA	Between codons
tFastX			Arbitrary
FastS	Linked peptides	Protein	Not an issue
FastF	Mixed peptides	Protein	Not an issue

Figure 5.2: Vairants of FastA algorithm

Under different circumstances it is favorable to use different programs:

- To identify an unknown protein sequence use either: FastA3, Ssearch3 or tFastX3
- To identify structural DNA sequence: (repeated DNA, structural RNA) use FastA3, first with $ktup = 6$ and than with $ktup = 3$.
- To identify an EST use FastX3 (check whether the EST codes for a protein homologous to a known protein).
- Use $ktup = 1$ for oligonucleotides (length < 20).
- Search speed and selectivity are controlled with the *ktup* (word-size) parameter. Searches with $ktup = 1$ are slower, but more sensitive, while $ktup = 2$ is faster but less effective.
- For proteins, the default is $ktup = 2$. For DNA, the default is $ktup = 6$. Use $ktup = 3$ or $ktup = 4$ more sensitivity.

FastA3 (Fastx3, etc.) is the current version of FastA. Tha main important over version 2 (which is still in common use) is a much better normalization method for alignment scores. FastA is available directly via the FastA3 server, [11] or it can be accessed through one of the retrieval systems e.g. the GenWeb mirror site at the Weizmann Lustitude.[7]

FastA - Steps

- Hashing: FastA locates regions of the query sequence and matching regions in the database sequences that have high densities of exact word matches. (without gaps) The length of the matched word is called the *ktup* parameter.

- Scoring: The ten highest scoring regions are rescored using a scoring matrix. The score for such a pair of regions is saved as the $init_1$ *score.*

- Introduction of Gaps: FastA determines if any of the initial regions from different diagonals may be joined together to form an approximate alignment with gaps. Only non-overlapping regions may be joined. The score for the joined regions is the sum of the scores of the initial regions minus a joining penalty for each gap. The score of the highest scoring region, at the end of this step, is saved as the $init_n$ *score.*

- Alignment: After computing the initial scores, FastA determines the best segment of similarity between the query sequence and the search set sequence, using a variation of the Smith-Waterman algorithm. The score for this alignment is the *opt score.*

- Random Sequence Simulation: In order to evaluate the significance of such alignment FastA empirically estimates the score distribution from the alignment of many random pairs of sequences. More precisely, the characters of the query sequences are reshuffled (to maintain bias due to length and character composition) and searched against a random subset of the database. This empirical distribution is extrapolated, assuming it is an extreme value distribution, and each alignment to the real query is assigned a Z-score and an E-score.

  The Fasta program can be used through an interface in WWW2.EBI.ac.uk/fasta3.[11]

FastA Output

An Example of the FastA histogram is shown below:
The X axis is the score,growing from top to bottom and printed on the left column . The y axis shows the number of matching database records having the score. The observed distribution of the score is plotted by bars of "=" signs and printed in second column. The expected random distribution of the score is plotted by "*" signs and printed on the

```
       opt      E()
< 20 82817     0:============================================================
  22     0     0:           one = represents 1381 library sequences
  24     0     0:
  26     0    11:*
  28     4   119:*
  30   224   725:*
  32   976  2804:= *
  34  3145  7603:===  *
  36  5406 15615:====       *
  38 11674 25805:=========         *
  40 33028 35996:========================  *
  42 58047 44001:===============================*===========
  44 70768 48537:===================================*================
  46 61240 49436:===================================*=========
  48 42577 47329:===============================   *
  50 23779 43188:==================             *
  52 12532 37970:==========                 *
  54  9284 32433:=======                *
  56 10961 27091:========           *
  58 16505 22241:============    *
  60 20473 18017:=============*=
  62 26145 14444:==========*========
  64 16445 11487:========*===
  66  8383  9079:======*
  68  4968  7142:==== *
  70  2049  5597:==  *
  72  1245  4373:=  *
  74   934  3410:= *
  76  1146  2654:=*
  78  1177  2063:=*
  80  1422  1602:=*
  82  1463  1225:*=
  84  1054   971:*
  86   800   751:*
  88   435   581:*          inset = represents 6 library sequences
  90   323   450:*
  92   154   348:*         :==========================             *
  94   141   269:*         :========================               *
  96   120   208:*         :====================              *
  98   134   161:*         :=======================   *
 100   100   125:*         :=================   *
 102   291    96:*         :===============*========================
 104   123    75:*         :============*========
 106    77    58:*         :=========*===
 108   207    45:*         :=======*===========================
 110    30    35:*         :=====*
 112    28    27:*         :====*
 114    14    21:*         :===*
 116     7    16:*         :==*
 118     8    12:*         :=*
>120    25    10:*         :=*===
171737814 residues in 532888 sequences
 statistics extrapolated from 60000 to 526287 sequences
```

third column. For example, a score of 34 was attained by 1045 sequences when the query was searched, compared to 1564 expected sequences with a random sequence search. This difference is visible at the corresponding histogram line: the '*' is not at the top of the '=' bar. Note, that on the bottom right a zoom is plotted, providing a closer look at the significant region.
following the graphic output of FastA.

```
The best scores are:                                              opt bits E(526287)
SWALL:Q9LXK2 Q9LXK2 HYPOTHETICAL 9.3 KDA PROTEIN.   (  87)   35   26      4.1
SWALL:Q9K3J8 Q9K3J8 PUTATIVE INTEGRAL MEMBRANE PR   ( 195)   35   26      6.6
SWALL:CFI2_MEDSA P28013 CHALCONE--FLAVONONE ISOME   ( 197)   35   26      6.7
SWALL:CFI1_MEDSA P28012 CHALCONE--FLAVONONE ISOME   ( 222)   35   27      7.2
SWALL:GR14_NEOCA Q25540 ANTIGEN NC14.1 (FRAGMENT)   ( 127)   34   26      8.3
SWALL:049070 049070 DNAJ HOMOLOGUE.                 ( 284)   35   27      8.3
SWALL:BAB11149 BAB11149 DNAJ HOMOLOGUE.             ( 284)   35   27      8.3
```

Figure 5.4: FastA output: The first lines contain general information about the search parameters. Score lines are made of nine rows: 1-3 details the name and the annotation of the hit, 4-9 are the FastA scores.

Each Line describes one database sequence matching the query, printed in decreasing order of statistical significance. Each contains the name of the record, its database ID, a short description of it, the sequence length and the five FastA scores. Although the statistical significance of a result can be evaluated directly from the E-score and Z-score, interpreting the row scores $(init_1, init_n andopt)$may be useful:

- When $init_1 = init_n = optthereis100\%$ homology over the matched stretch.
- When $init_n > init_1$ more than one matching region was found in the database sequence with poorly matching separating regions.
- When $opt > init_n$ the matching regions are greatly improved by adding gaps in one or both of the sequences.

5.2.2 BLAST - Basic Local Alignment Search Tool

Blast programs use a heuristic search algorithm. The programs use the statistical methods of Karlin and Altschul [2]. BLAST programs were designed for fast database searching, with minimal sacrifice of sensitivity for distantly related sequences. The programs search databases in a special compressed format. To use your own private database with BlAST, you need to convert it to the BLAST format. Direct pointer: The BLAST at NCBI: [16] It also can be run through one of the retrieval systems (recommended). For example: GeneWeb mirror site at Weizmann Institute: [7]

BLAST is a family of programs, following is a table of the BALST variants:

BLAST variant	Query sequence	Database sequence	Comments
BLASTN	DNA	DNA	
BLASTP	Protein	Protein	
BLASTX	DNA	Protein	The comparison made by translating the query sequence in the six possible frames and comparing each against the database (3 reading frames from each strand of the DNA) searching
TBLASTN	Protein	DNA	Comparison in the six possible frames of the database
TBLASTX	Protein encoded in a DNA query	Protein encoded in DNA query	Comparison in the 6×6 possible frames of both query and database sequences (combinations of frames may have different scores)

Figure 5.5: Variants of BLAST family

The BLAST program compares the query to each sequence in database using heuristic rules to speed up the pairwise comparison. It first creates *sequence abstraction* by listing exact and similar words. This is done in advance for each sequence in the database on the run for a certain query. BLAST finds similar words between the query and each database sequence. It then extends such words to obtain *high-scoring sequence pairs (HSPs)*. It also calculates statistics analytically like FastA does.

BLAST 2.0 is a new version with new capabilities(Also called *advanced BLAST*). It can perform gapped alignments) such as Gapped-BLST and PSI-BLAST (detail below). The **GAPPED BLAST** algorithm allows gaps to be introduced into the alignments. That means that similar regions are not broken into several segments (as in the older versions). This method reflects biological relationships much better than ordinary BLAST.

Blast interfaces like www.ncbi.nlm.nih.gov/BLAST can be used to run BLAST of a chosen query against the NCBI databases .

BLAST Output

following is the BLAST graphical output:

This output is very similar to the FastA output: Each Line describes one database sequence matching the query. The local alignments themself are detailed further along the output (not shown in the above figure).

5.2.3 Significance of Scores

FastA scores

we want to know how meaningful is the score we got for a specific query against a specific databases run. One possible solution is to compare the score against scores obtained from a search of random query against the same database.

The FastA algorithm estimates the distribution of these scores by empirical simulation. Note that this random distribution is not Gaussian. Under reasonable assumptions it is *Extreme Value Distribution*. The easy case is where that the real query score distribution is very different from the random score distribution - and so it's easy to say whether the score is significant.

A more complex case is presented in the next figure. In this case the random and real distribution have a large common area along the score axis, this means many random records have a better or equivalent score to the query score and it is hard to distinguish really related records from random ones.

The widely accepted solution is to consider the E-value: the expectancy of the number of records with score that exceed our query score in a the random distribution. if the E-value is low enough will consider the score to be significant.

i.e an E-score of $\sim 10^{-50}$ suggests, with extremely high confidence, that the query is evolutionary related to the target matched in the database.

Another way of measuring the significance of a score considers the mean of the random score distribution. This distance is normalized by the standard deviation of that distribution to form the *Z-score*. Higher $Z - scores$ are better because the further the real score is from this mean (in standart deviation units) - the more significant it is.

Under reasonable assumptions the random score distribution for optimal ungaped local alignments can be proved to follow *extreme value distribution* (which proved to be significantly different from the normal distribution) [3]. In the current versions of FastA and

```
                                                                 Score    E
Sequences producing significant alignments:                      (bits)  Value

pir||S01220  NADH dehydrogenase (ubiquinone) (EC 1.6.5.3) ch...   260  4e-69
ref|NP_007108.1|ND2_10504  NADH dehydrogenase subunit 2 [Art...   238  3e-62
sp||NU2M_ARTSA_1  [Segment 1 of 2] NADH-UBIQUINONE OXIDOREDU...   160  7e-39
sp||NU2M_ARTSA_2  [Segment 2 of 2] NADH-UBIQUINONE OXIDOREDU...   105  3e-22
gb|AAF72121.1|AF216203_13  (AF216203) NADH dehydrogenase sub...    81  4e-15
ref|NP_062838.1|  NADH dehydrogenase subunit 2 [Loligo bleek...    79  3e-14
ref|NP_008185.1|ND2_10528  NADH dehydrogenase subunit 2 [Kat...    77  8e-14
gb|AAG33817.1|  (AF295564) NADH dehydrogenase subunit 2 [Bom...    74  5e-13
ref|NP_059475.1|  NADH dehydrogenase subunit 2 [Bombyx mori]...    74  5e-13
ref|NP_006902.1|ND2_10015  NADH dehydrogenase subunit 2 [Dro...    74  5e-13
gb|AAF77278.1|  (AF200832) NADH dehydrogenase subunit 2 [Dro...    73  2e-12
gb|AAF77252.1|  (AF200830) NADH dehydrogenase subunit 2 [Dro...    73  2e-12
gb|AAF77291.1|  (AF200833) NADH dehydrogenase subunit 2 [Dro...    73  2e-12
gb|AAF78613.1|AF260826_12  (AF260826) NADH subunit 2 [Cochli...    72  2e-12
gb|AAF77265.1|  (AF200831) NADH dehydrogenase subunit 2 [Dro...    72  2e-12
emb|CAA78734.1|  (Z15015) NADH dehydrogenase subunit 2 [Daph...    72  3e-12
ref|NP_008621.1|ND2_15068  NADH dehydrogenase subunit 2 [Dap...    72  3e-12
gb|AAF77226.1|  (AF200828) NADH dehydrogenase subunit 2 [Dro...    72  3e-12
gb|AAF77239.1|  (AF200829) NADH dehydrogenase subunit 2 [Dro...    72  3e-12
emb|CAB91051.1|  (AJ400907) NADH dehydrogenase subunit 2 [Dr...    72  3e-12
ref|NP_008069.1|ND2_10183  NADH dehydrogenase subunit 2 [Ano...    72  3e-12
ref|NP_008647.1|ND2_15072  NADH dehydrogenase subunit 2 [Cer...    71  5e-12
gb|AAF77369.1|  (AF200839) NADH dehydrogenase subunit 2 [Dro...    70  8e-12
gb|AAF63423.1|AF150756_5  (AF150756) NADH dehydrogenase subu...    70  1e-11
ref|NP_008277.1|ND2_10704  NADH dehydrogenase subunit 2 [Dro...    70  1e-11
ref|NP_007290.1|ND2_10712  NADH dehydrogenase subunit 2 [Loc...    69  2e-11
pir||T13781  NADH dehydrogenase (ubiquinone) (EC 1.6.5.3) ch...    68  4e-11
ref|NP_008686.1|ND2_15089  NADH dehydrogenase subunit 2 [Ano...    68  4e-11
emb|CAA64821.1|  (X95574) NADH dehydrogenase subunit 2 [Chor...    68  4e-11
prf||2015259A  NADH dehydrogenase:SUBUNIT=2 [Anopheles quadr...    68  4e-11
```

Figure 5.6: The BLAST Output: Each line is made of 4 columns: 1-2 details the name and the annotation of the hit, 3 is the bit score and 4 is the E-value of the hit.

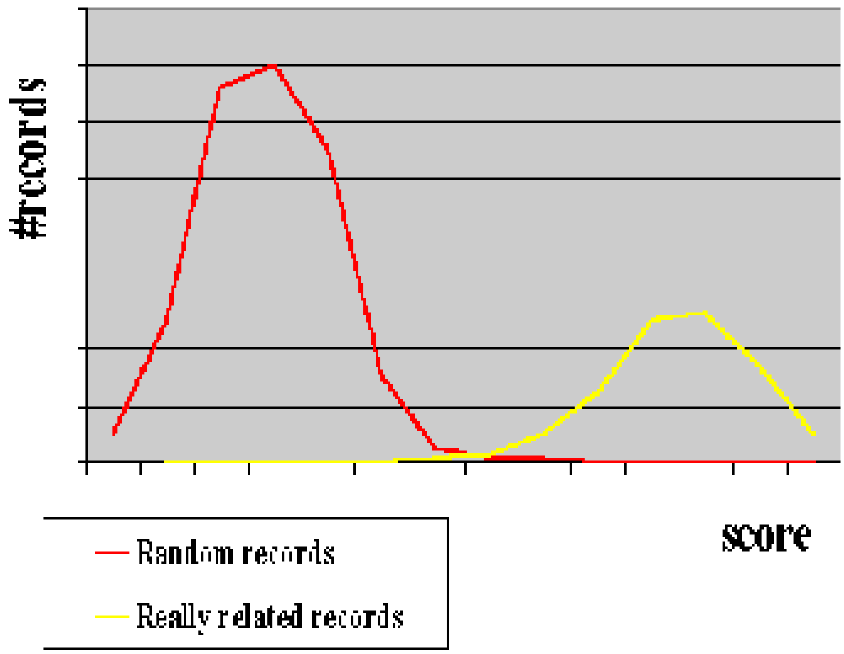

Figure 5.7: Easy Case: Illustration of an easy case of estimating significance. The score of really related records are distributed away from random records and thus can easily identified.

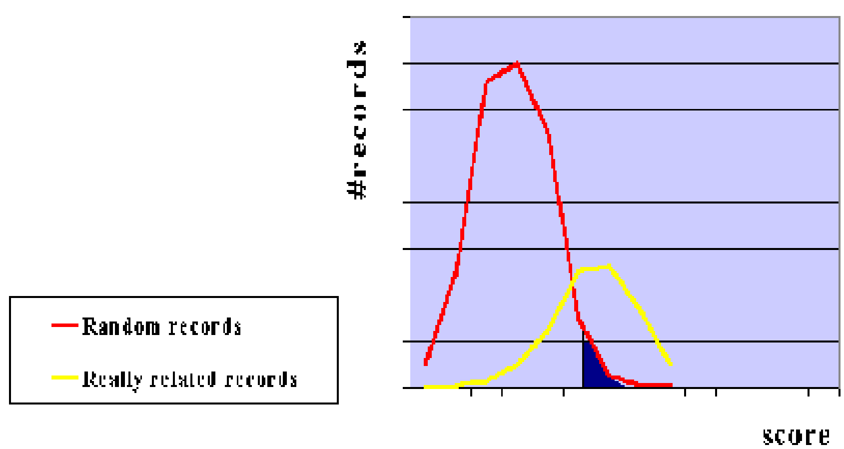

Figure 5.8: Complex Case: Illustration of a complex case of estimating significance. The dark area represents the number of random records (shuffeled query sequnce) that exceed the query score. In this case the common area betweeb the random plot and the real plot is large, which makes it hard to distinguish between the real and random ones

.

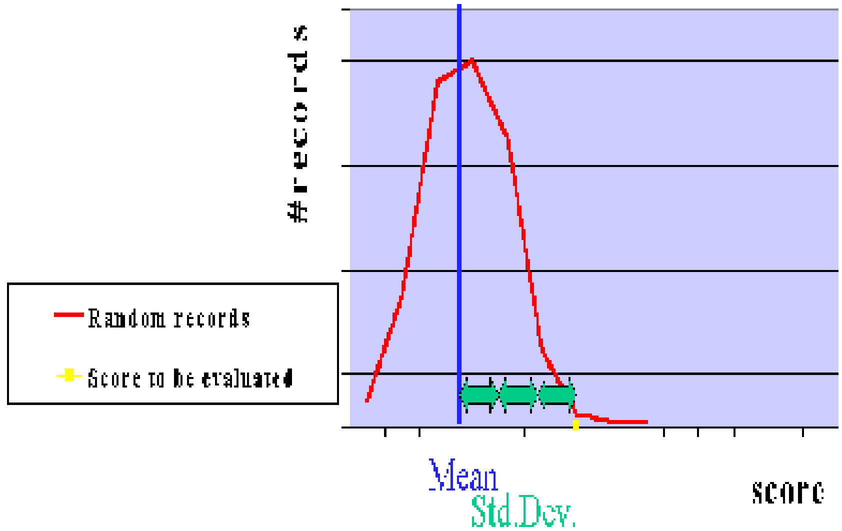

Figure 5.9: Fasta Z-score: The arrows represents the distance in Std.Dev units from the distribution mean. The Z-Score = (RawScore- mean)/ Std.Dev

BLAST search programs, the evaluation of statistical significance is based upon the extreme value distribution. These evaluations take the form of *E-scores*.
Once the E-value is known, one can count how many records observed with such E-value or lower, and compare this number with its theoretical distribution in the random model. The probablity of the observed number (*P-value*) messures it's significance: The smaller it is the more unsual is our result.
More formally: Define the random variable: Ye = The observed number of random records achieving E-value E or better(smaller). Ye is distributed Poisson with parameter E:

$Prob(Y_e = r) = \frac{exp(-E)E^r}{r!}$

Note that this model assumes an I.I.D trial for each databse record.
Under these assumptions, the alignment score behaves like a *random walk* when the alignment is elongated. The probablility if the highest score is such a random walk is at least s ,decreases exponentially with s. [1]

BLAST Score

BLAST scores rely on extensive theory. We start by making the following assumptions:

- The BLAST score is scoring local ungapped alignments. The theory of scoring here is well understood.
- The database sequences are assumed to be evolutionary unrelated, i.e. independent of one another.

- The alignment starts at specific positions along query and database record.
- The score matrix must give, on the average, a negative σ(a,b) score. Were this not the case, long alignments would tend to have high score independently of whether the segment aligned were related, and the statistical theory would break down.

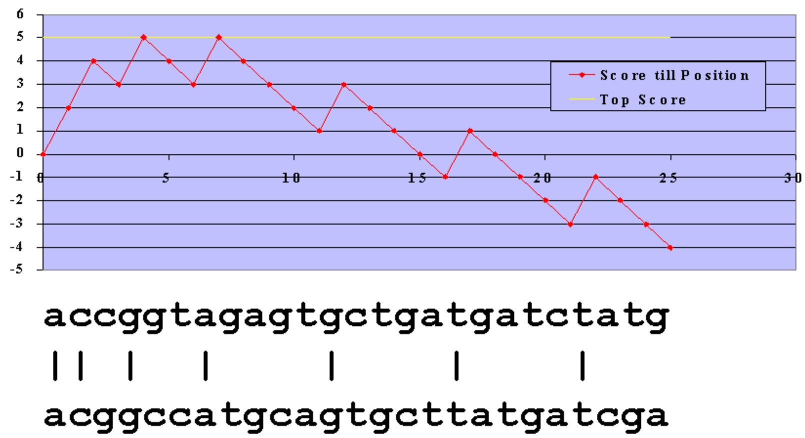

Figure 5.10: Random walk: The score for a match is +2 and the punishment for a missmatch is -1, As shown,the expectancy for the whole walk is negative. The probability that the Top Score will be larger than X decreases exponentially with x.

When searching a query of length m in a database of total length n one performs $m * n$ random walk experiment, each with exponentially decreasing probability of achieving a score S. Thus, the E-value for score s is: $KmnE^{-\lambda S}$. λ and K are constants:

- λ - scaling factor
- K - correction for dependency and bias of the scoring scheme.

Indeed the E-score is normalized by the length of the query and database: The same alignment would have different E-score if these length are different. Also the E-score is exponential, thus it is instructive to consider a normalization of the E-score into logarithmic scale, called the *Bit - score.*

The Bit-score B is computed from the E-score E by $E = mn2^{-B}$. Obviously, the Bit-score is linear in the raw score s: $B = \frac{\lambda S - ln(K)}{ln(2)}$.
In contrast to raw scores, that have little meaning without k and λ, the Bit-score is measured in standard units (see eg. [17]). Naturally, the meaning of the Bit-score depends on sizes of the query and the database.

Again, as mentioned before one can ask for the *P-value* (the probability of the observed number of records with a known E-value or lower).
Define the random variable Y to be the observed number of pairs achieveing E-value E or better(smaller).

Y is distributed Poisson with (E). The Probability of Y_e to be r is $\frac{exp(-E)E'}{r!}$, and the probability of $\boldsymbol{Y_e}$ to be 0 is equivilant to the probability that the (Best E-score $<$ E)=exp (-E). Specifically the chance of finding zero alignments with score $>=$ S is $\boldsymbol{e^{-E}}$ so the probability of finding at least one such alignment is $\boldsymbol{1 - e^{-E}}$. This is the P-value associated with the score S (see eg. [17]). Note that this model assumes an I.I.D trial for each database position.

5.2.4 PSI BLAST

- **PSI - BLAST** - (Position Specific Iterated BLAST) provides a new automatic "profile like" search. The program first performs a gapped BLAST search of the database. The significant alignments are then used by the program to construct a *position specific score matrix(PSSM)* . This matrix replaces the query sequence in the next round of database searching. The program may be iterated until no new significant alignments are found.

PSI BLAST Output

5.2.5 The Smith-Waterman Tool

Smith-Waterman (SW) searching method compares the query to each sequence in database using the full Smith-Waterman algorithm for pairwise comparisons [6]. It also uses search results to generate statistics. Since SW searching is exhaustive, it is the slowest method. We use a special hardware + software (Bioccelerator) to accelerate the application. A Bioccelerator can be found in the TAV bio-informatics department. Direct pointer: [10]
It also can be run through the Weizmann Institute site: [7]

Smith-Waterman Output

5.2.6 Comparison of the Programs

- Concept:
  SW and BLAST produce local alignments, while FastA is global alignment tool. BLAST can report more than one HSP per database entry, while FastA reports only one segment (match).

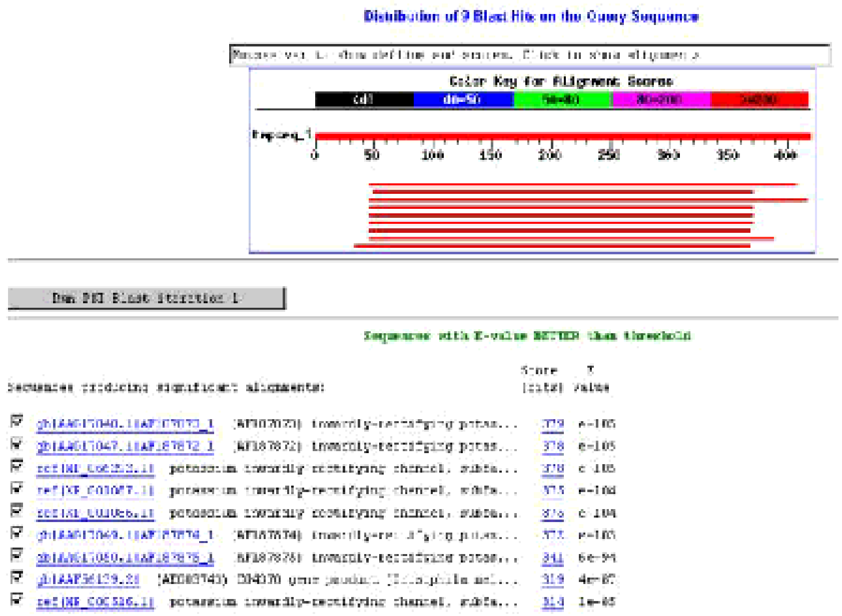

Figure 5.11: The Psi Blast Output: The area in each hit which align to the query is shown graphically in its right position. Hits are ordered in descending score order. Below the graphic description there are detailed lines for each hit. Those lines are identical to the BLAST output hit lines except the check boxes in the first columns. Those boxes allow the selection of the hits that will used to build the position specific score matrix for the next iteration.

Figure 5.12: Smith-Waterman Output: Each line contains information about a hit. The hit accsession in the database, a list of links to other important bioinformatic sites, the strand, the scores of the alignment, the length of the alignment and some documentation about the hit.

- Speed:
  BLAST $>$ FastA $\gg$ SW
  BLAST (package) is a highly efficient search tool.

- Sensitivity:
  SW $>$ FastA $>$ BLAST (old version!)
  FastA is more sensitive, missing less homologues (but the opposite can also happen - if there are no identical residues conserved, but this is infrequent). It also gives better separation between true homologues and random hits.

- Statistics:
  BLAST calculates probabilities, and it sometimes fails entirely if some of the assumptions used are invalid. FastA calculates significance 'on the fly' from the given dataset which is more relevant but can be problematic if the dataset is small.

5.2.7 Tips for DB Searches

- Use latest database version

- Run BLAST first, then depending on your results run a finer tool (FastA, Ssearch, SW, Blocks, etc..)

- Whenever possible, use the translated sequence.

- E $<$ 0.05 is statistically significant, usually biologically interesting. Check also
  0.05 $<$ E $<$ 10 because you might find interesting hits.

- Pay attention to abnormal composition of the query sequence, since it usually causes biased scoring.

- Split large query sequences ($>$ 1000 for DNA, $>$ 200 for protein).

- If the query has repeated segments, remove them and repeat the search.

5.3 Recommended Sites on the Web

A collection of bioinformatic links: [15]
Biocatalog of Programs: [12]

Bibliography

[1] A.Dembo and S.Karlin. Strong limit theorems of empirical functionals for large excedances of partial sums of i.i.d variables. *Annuals of Probability*, 19(4):1737–1755, 1991.

[2] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *J. Mol. Biol*, 215:403–410, 1990.

[3] S. Karlin and S. F. Altschul. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl. Acad. Sci. USA*, 87:2264–2268, 1990.

[4] F. Lewitter. Text-based database searching. *Trends Guide to Bioinformatics*, pages 3–5, 1998.

[5] R. W. Pearson and D. J. Lipman. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA*, 85:2444–2448, 1988.

[6] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *J. Mol. Biol.*, 147:195–197, 1981.

[7] `http://dapsas1.weizmann.ac.il\bcd\bcd_parent\bcd_bioccel\bioccel.html.`.

[8] `http://srs/ebi/ac/uk/`.

[9] `http://srs.ebi.ac.uk/srs6/man/mi_srswww.html`.

[10] `http://www2.ebi.ac.uk/bic_sw/`.

[11] `http://www2.ebi.ac.uk/fasta3/`.

[12] `http://www.ebi.ac.uk/biocat/biocat_form.html`.

[13] `http://www.genome.ad.jp/dbget/dbget2.html` .

[14] `http://www.genome.ad.jp/dbget/dbget_manual.html` .

[15] `http://www.ii.uib.no/~inge/list.html`.

[16] `http://www.ncbi.nlm.nih.gov/BLAST/`.

[17] `http://www/ncbi.nlm.nih.gov/blast/tutorial/Altschul-1/`.

[18] `http://www/ncbi.nlm.nih.gov/Entrez/`.

[19] `http://www/ncbi.nlm.nih.gov/Entrez/entrezhelp.html`.