MCB 372

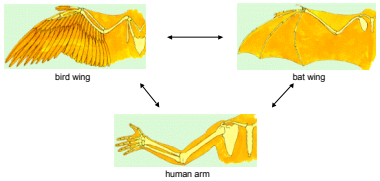
BLAST, unix, Perl

J. Peter Gogarten
Office: *BPB 404*
phone: *860 486-4061,*
Email: *gogarten@uconn.edu*

Theodosius Dobzhansky:

"Nothing in biology makes sense except in the light of evolution"

Homology

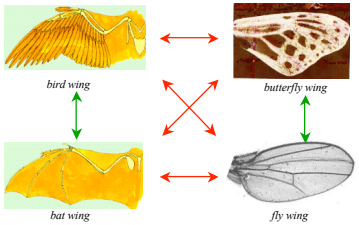


by Bob Friedman

homology *vs* analogy

A priori sequences could be similar due to convergent evolution

Homology (shared ancestry) *versus* Analogy (convergent evolution)



Related proteins

Present day proteins evolved through substitution and selection from ancestral proteins.
Related proteins have similar sequence AND similar structure AND similar function.

In the above mantra "similar function" can refer to:

- identical function,
- similar function, e.g.:
  - identical reactions catalyzed in different organisms; or
  - same catalytic mechanism but different substrate (malic and lactic acid dehydrogenases);
  - similar subunits and domains that are brought together through a (hypothetical) process called domain shuffling, e.g. nucleotide binding domains in hexokinse, myosin, HSP70, and ATPsynthases.

homology

Two sequences are homologous, if there existed an ancestral molecule in the past that is ancestral to both of the sequences

Homology is a "yes" or "no" character (don't know is also possible). Either sequences (or characters share ancestry or they don't (like pregnancy). Molecular biologist often use homology as synonymous with similarity of percent identity. One often reads: sequence A and B are 70% homologous. To an evolutionary biologist this sounds as wrong as 70% pregnant.

Types of Homology
Orthology: bifurcation in molecular tree reflects speciation
Paralogy: bifurcation in molecular tree reflects gene duplication

Sequence Similarity *vs* Homology

The following is based on observation and not on an *a priori* truth:

If two (complex) sequences show significant similarity in their primary sequence, they have shared ancestry, and probably similar function.
(although some proteins acquired radically new functional assignments, lysozyme -> lense crystalline).

The Size of Protein Sequence Space

(back of the envelope calculation)

Consider a protein of 600 amino acids.
Assume that for every position there could be any of the twenty possible amino acid.
Then the total number of possibilities is 20 choices for the first position times 20 for the second position times 20 to the third .... = 20 to the 600 = $4*10^{780}$ different proteins possible with lengths of 600 amino acids.

For comparison the universe contains only about 10^{89} protons and has an age of about $5*10^{17}$ seconds or $5*10^{29}$ picoseconds.

If every proton in the universe were a super computer that explored one possible protein sequence per picosecond, we only would have explored $5*10^{118}$ sequences, i.e. a negligible fraction of the possible sequences with length 600 (one in about 10^{662}).

no similarity *vs* no homology

If two (complex) sequences show significant similarity in their primary sequence, they have shared ancestry, and probably similar function.

THE REVERSE IS NOT TRUE:

PROTEINS WITH THE **SAME OR SIMILAR FUNCTION DO NOT ALWAYS SHOW SIGNIFICANT SEQUENCE SIMILARITY**
for one of two reasons:

a) they evolved independently
(e.g. different types of nucleotide binding sites);
or
b) they underwent so many substitution events that there is no readily detectable similarity remaining.

Corollary: PROTEINS WITH SHARED ANCESTRY DO NOT ALWAYS SHOW SIGNIFICANT SIMILARITY.

homology

Two sequences are homologous, if there existed an ancestral molecule in the past that is ancestral to both of the sequences

Types of Homology

Orthologs: "deepest" bifurcation in molecular tree reflects speciation.
These are the molecules people interested in the taxonomic classification of organisms want to study.

Paralogs: "deepest" bifurcation in molecular tree reflects gene duplication. The study of paralogs and their distribution in genomes provides clues on the way genomes evolved. Gen and genome duplication have emerged as the most important pathway to molecular innovation, including the evolution of developmental pathways.

Xenologs: gene was obtained by organism through horizontal transfer. The classic example for Xenologs are antibiotic resistance genes, but the history of many other molecules also fits into this category: inteins, selfsplicing introns, transposable elements, ion pumps, other transporters,

Synologs: genes ended up in one organism through fusion of lineages. The paradigm are genes that were transferred into the eukaryotic cell together with the endosymbionts that evolved into mitochondria and plastids

(the -logs are often spelled with "ue" like in orthologues)

see Fitch's article in TTG 2000 for more discussion.

Uses of Blast in bioinformatics

The Blast web tool at NCBI is limited:
- custom and multiple databases are not available
- tBlastN (gene prediction) not available
- "time-out" before long searches are completed

What if researcher wants to use tBlastN to find all olfactory receptors in the mosquito? Or, if you want to check the presence of a (pseudo)gene in a preliminary genome assembly?
Answer: Use Blast from command-line

Also: The command-line allows the user to run commands repeatedly

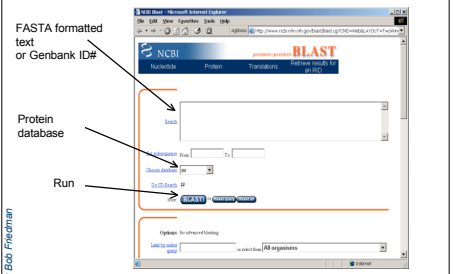
by Bob Friedman

Types of Blast searching

- blastp compares an amino acid query sequence against a protein sequence database
- blastn compares a nucleotide query sequence against a nucleotide sequence database
- blastx compares the six-frame conceptual protein translation products of a nucleotide query sequence against a protein sequence database
- tblastn compares a protein query sequence against a nucleotide sequence database translated in six reading frames
- tblastx compares the six-frame translations of a nucleotide query sequence against the six-frame translations of a nucleotide sequence database.

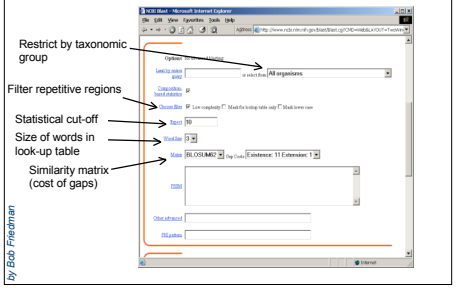
by Bob Friedman

Routine BlastP search



by Bob Friedman

BlastP parameters



by Bob Friedman

Establishing a significant "hit"

Blast's E-value indicates statistical significance of a sequence match
Karlin S, Altschul SF (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. PNAS 87:2264-8

E-value is the Expected number of sequence (HSPs) matches in database of *n* number of sequences
- database size is arbitrary
- multiple testing problem
- E-value calculated from many assumptions
- so, E-value is not easily compared between searches of different databases

Examples:
E-value = 1 = expect the match to occur in the database by chance 1x

E-value = .05 = expect 5% chance of match occurring

E-value = $1x10^{-20}$ = strict match between protein domains

by Bob Friedman

When are two sequences significantly similar? PRSS

One way to quantify the similarity between two sequences is to

1. compare the actual sequences and calculate an alignment score
2. randomize (scramble) one (or both) of the sequences and calculate the alignment score for the randomized sequences.
3. repeat step 2 at least 100 times
4. describe distribution of randomized alignment scores
5. do a statistical test to determine if the score obtained for the real sequences is significantly better than the score for the randomized sequences

z-values give the distance between the actual alignment score and the mean of the scores for the randomized sequences expressed as multiples of the standard deviation calculated for the randomized scores.

For example: a z-value of 3 means that the actual alignment score is 3 standard deviations better than the average for the randomized sequences. z-values > 3 are usually considered as suggestive of homology, z-values > 5 are considered as sufficient demonstration.

PRSS continued

To illustrate the assessment of similarity/homology we will use a program from Pearson's FASTA package called PRSS.
This and many other programs by Bill Pearson are available from his web page at ftp://ftp.virginia.edu/pub/fasta/.

A web version is available here.

Sequences for an in class example are here (fl), here (B), here (A) and here (A2)

BLAST offers a similar service for pairwise sequence comparison bl2seq, however, the statistical evaluation is less straightforward.

To force the bl2seq program to report an alignment increase the E-value.

E-values and significance

Usually E values larger than 0.0001 are not considered as demonstration of homology.

For small values the E value gives the probability to find a match of this quality in a search of a databank of the same size by chance alone.

E-values give the expected number of matches with an alignment score this good or better,
P-values give the probability of to find a match of this quality or better.
P values are [0,1], E-values are [0,infinity).
For small values E=P

Problem: If you do 1000 blast searches, you expect one match due to chance with a P-value of 0.0001

"One should" use a correction for multiple tests, like the Bonferroni correction.

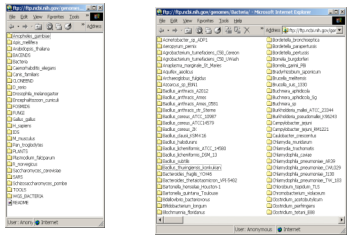
Blast databases

- EST - Expression Sequence Tags; cDNA
- GSS - Genome Survey Sequence; single-pass genomic sequences
- HTGS - unfinished High Throughput Genomic Sequences
- chromosome - complete chromosomes, complete genomes, contigs
- NR - non-redundant DNA or amino acid sequence database
- NT - NR database excluding EST, STS, GSS, HTGS
- PDB - DNA or amino acid sequences accompanied by 3d structures
- STS - Sequence Tagged Sites; short genomic markers for mapping
- Swissprot - well-annotated amino-acid sequences
- TaxDB - taxonomy information
- WGS_xx - whole genome shotgun assemblies

- Also, to obtain organism-specific sequence set: ftp://ftp.ncbi.nih.gov/genomes/

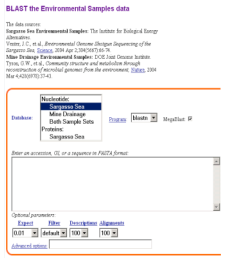
by Bob Friedman

More databases



by Bob Friedman

And more databases

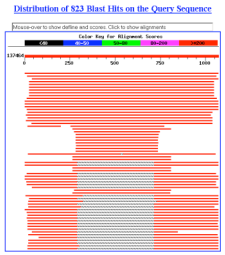


by Bob Friedman

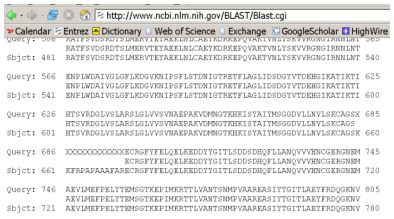
Example of web based BLAST

program: BLASTP
sequence: vma1
gi:137464

BLink provides similar information



Effect of low complexity filter



BUT the most common sequences are simple repeats

Custom databases

Custom databases can include private sequence data, non-redundant gene sets based on genomic locations, merging of genetic data from specific organisms

It's also faster to search only the sequence data that is necessary

Can search against sequences with custom names

by Bob Friedman

Formatting a custom database

Format sequence data into Fasta format

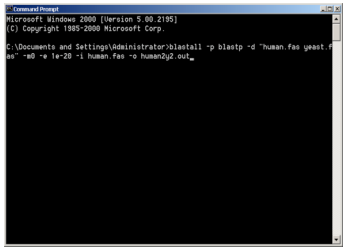
Example of Fasta format:
>sequence 1
AAATGCTTAAAAA
>sequence 2
AAATTGCTAAAAGA

Convert Fasta to Blast format by using FormatDB program from command-line:
formatdb -p F -o T -i name_of_fasta_file

(formatdb.log is a file where the results are logged from the formatting operation)

by Bob Friedman

BlastP search of custom database



by Bob Friedman

NR, GenBank, and EMBL

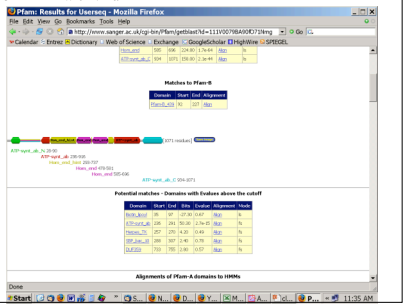
The European, Japanese and US sequence repository have agreed on the information that needs to be contained in a databank submission – the layout of the forms is different, but the content is the same. They share the information that one of the cooperating databanks receives.

An example of a nucleotide sequence entry (GenBank format – note that the NCBI switched from a flatfile databank to an object oriented one that uses the asn format) is here.

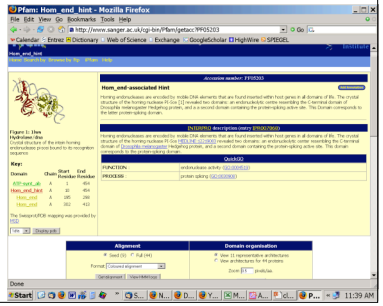
Other frequently used formats are described here.

else: bionet, Intelligenetics, genbank, a trip down memory lane ncbi, PIR, uniprot, genpept, pdb, Structural Classification Of Proteins (SCOP), PFAM and CDD

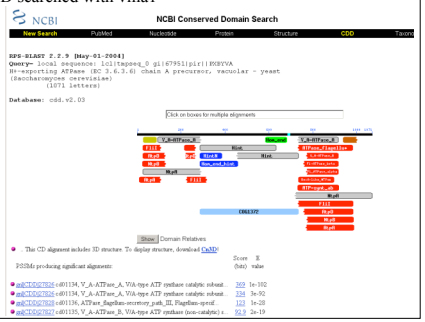
Search of PFAM with vma1



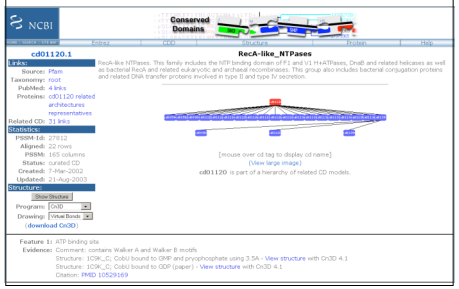
Search of PFAM with vma1, continued



CDD searched with vma1



CDD searched with vma1(cont.)



Command Line

The favored operating system flavor in computational biology is UNIX/LINUX.
The command line is similar to DOS.
Some of the frequently used commands are here

`pwd`	`ps`
`ls`	`ps aux`
`ls -l`	`rm`
`chmod`	`more`
`chmod a+x blastall.sh`	`cat`
`chmod 755 *.sh`	`vi (text editor)`
`cd`	`ps`
`cd ..`	`ps aux`
`cd $HOME`	`ssh`
`passwd`	`sftp`

For windows an "ok" ssh program is putty.
UConn also has a site license for the ssh program from ssh.com

UNIX

Basic UNIX commands
ls, cd, chmod, cp, rm, mkdir, more (or) less, vi, ps, kill -9, man
A brief listing is here

chmod is a particular pain in the ... .
Under unix every file has an owner and the owner, his group and everyone else have permissions to read, write and/or execute the file (or they don't). If you want to see which permissions are currently assigned to your files, type ls -l at the command prompt.
chmod a+x *.pl gives everyone execute permission for all files that end with .pl the * is a wildcard. (warning don't ever use rm in conjunction with *)
For more on chmod type "`man chmod`" or see here.
(In the OSX GUI you can control click at a file, and change permissions in the info box). Most ssh clients (FUGU and SSH) allow you to use a GUI to change file permissions (in FUGU ctrl click).

Unix - command line interface

If you tried to execute a command, and you made a mistake, for example, you mistyped a file name, you can recall the last command using the up arrow (down arrow for more recent).

If you are tired typing long filenames, you can use the tab key to complete the line, provided there is only one way to complete the line. E.g: cd /Desktop could be replaced by cd /D<tab>
If there are two or more choices you hear a boing, if you hit <tab> again, you get a list of choices.

writing Perl scripts

Use unix/ linux /OsX if possible (talk with Tim if you want to use windows).
A) open a terminal window ; type "which perl <return>"
B) SSH to a unix machine (cluster@bbcxrv1), log in, type "which perl <return>"
C) to check the version type perl -v <return>The response of the system should tell you, where Perl is installed on your machine (you need to know this for the first line of your perl program, which tells the operating system how to interpret what follows. On most installations this is #!/usr/bin/perl).
WINDOWS: If you use a windows machine, you can use an ssh program to connect to the biotech cluster. A good ssh client is available at ftp://ftp.ssh.com/pub/ssh/- highly recommended. A reasonable text editor is available at http://www.context.cx/
MAC OsX: If you use a Mac under OS X, and you do not want to (only) use the PERL locally, you want to install both jellyfish (ssh terminal) and fugu (a secure file transfer program). Both are available at ftp://ftp.uconn.edu/pub/packages/ssh/mac/ or through the people who wrote the software - GOOGLE) Also, the bbcxsrv1 is available as a server using ssh or apl. You can connect to to it from the finder menu (-> GO -> Connect to Server) pasting the following into the menu box afp://bbcxsrv1.biotech.uconn.edu (select your account).
LINUX: Most editors on linux systems recognize Perl programs and provide context dependent coloring. Ssh and Konquerer work well for file transfer.

characters at the end of lines

File tranfers from Windows to UNIX and return:
End of Line characters are a problem. Under Windows DO NOT use notepad, it does not understand UNIX newline symbols '\n'.
Best write your programs under UNIX using vi or vim (or any other editor you are comfortable with)
2nd best is to use a text editor like textwrangler (very nice and free program for UNIX). Like vi and vim it provides context dependent coloring.
3rd best is to remove end of line symbols in a UNIX editor or use sed (Stream EDitor) after you transferred the file:
`sed s/.$// name_of_WINDOWS_infile > name_of_UNIX_outfile`
(This replases the last non letter character before the eol ($) with nothing)

Some versions of office allow to change files as UNIX textfiles, but ...

A related problem is encountered by Mac users. Most text editors will use MAC carriage returns at the end of the line. Most unix programs will not be able to handle these. In a terminal window you could use the following command to convert your file:
tr '\r' '\n' < name_of_the_Mac_file > name_of_the_unix_file
If you are working in a GUI environment, you also could use the convertNewLines.app program (install it in your application folder, drag the file you want to convert into the icon). The program is available here. This is very inconvenient, but there really is no easy solution, tough luck; and you better know about this incase something goes wrong.

vi

A short introduction to vi is at http://goforit.unk.edu/unix/unix11.htm -- however, if you run into problems google usually helps (e.g. google: vi replace unix gives you many pages of info on how to replace one string with another under vi)

`vi myprogram.pl` #starts the editor and loads the file myprogram.pl into the editor

The following should get you started:
The arrow keys move the cursor in the text (if you have a really dumb terminal you can use the letter hjkl to move the cursor)

`x` deletes the character under the cursoresc (i.e. the escape key) leaves the edit modei enters the edit mode and inserts before the cursora enters the edit mode and appends
`esc :` opens a command line (here you can start searches, and replacements)
`:w` #saves the file
`:w` new_name _of_file #writes the file into a new file.
`:wq` #saves the file and exits vi
`:q!` #exits vi without saving

customizing vi

One of the beauties of vi is that usually it provides context dependent coloring.
You need to tell vi which terminal you use.
One way to do so is to add a file called .vimrc to your home directory.

The following works under both, MAS OSX and using ssh via the secure shell program under windows:
`vi .vimrc` #opens vi to edit .vimrc (Files that start with a dot are not listed if you list a directory. List with `ls -a`)
`set term=xterm-color` #tells the editor that you use a terminal that conforms to some standard
`syn on` # tells the editor program that you want to use syntax dependent coloring.
`esc:wq`

This might seem a little inconvenient, but it really comes in handy to trouble shoot the program in the same environment where you want to run it.
(comment on textwrangler alternative, ssh is included inside the program)

PERL conventions and rules

Basic Perl Punctuation:
line ends with ";"
empty lines in program are ignored
comments start with #
first line points to path to interpreter:

```
#! /usr/bin/perl
# "#!" is known as "shebang";
```

keep one command per line for readability
use indentation do show program blocks.
Variables start with $calars, @rrays, or %ashes
Scalars: foating point numbers, integers,
non decimal integers, strings

Scalar variable are placeholders that can be assigned a scalar value (either number or string).

Scalar variables begin with $

```
$n=3; #assigns the numerical value 3 to the variable $n.
#Variables are interpolated, for example if you print text

 $b = 4 + ($a - 3); # assign 3 to $a, then add 4 to that
 # resulting in $b getting 7
 $d = ($c = 5); # copy 5 into $c, and then also into $d
 $d = $c = 5; # the same thing without parentheses

 $a = $a + 5; # without the binary assignment operator
 $a += 5; # with the binary assignment operator

 $str = $str . " "; # append a space to $str
 $str .= " "; # same thing with assignment operator

 "hello" . "world" # same as "helloworld"
 'hello world' . "\n" # same as "hello world\n"
 "fred" . " " . "barney" # same as "fred barney"
 "fred" x 3 # is "fredfredfred"
 "barney" x (4+1) # is "barney" x 5, or # "barneybarney....."
 (3+2) x 4 # is 5 x 4, or really "5" x 4, which is "5555"
```

Note: '=' Does not denote a mathematical equations but assignments!

Numbers can be manipulated using the typical symbols:

```
2 + 3 # 2 plus 3, or 5
5.1 - 2.4 # 5.1 minus 2.4, or approximately 2.7;
3 * 12 # 3 times 12 = 36;
2**3 # 2 taken to the third power = 2*2*2 = 8
14 / 2 # 14 divided by 2, or 7;
10.2 / 0.3 # 10.2 divided by 0.3, or approximately 34;
10 / 3 # always floating point divide, so approximately 3.3333333...
```

Special characters:

```
\n #newline
\t #tab
```

Double quoted strings are interpolated by the Perl interpreter:

```
"hello world\n" # hello world, and a newline
"new \177" # new, space, and the delete character (octal 177)
"coke\tsprite" # a coke, a tab, and a sprite
```

The backslash can precede many different characters to mean different things (typically called a backslash escape).

Variable interpolation - single quoted strings are not interpolated:

```
'hello' # five characters: h, e, l, l, o
'don\'t' # five characters: d, o, n, single-quote, t
'' # the null string (no characters)
'silly\\me' # silly, followed by backslash, followed by me
'hello\n' # hello followed by backslash followed by n
'hello
there' # hello, newline, there (11 characters total)
```

Example demo SSH to bbcxsrv.biotech.uconn.edu

A) Move files to bbcxsrv p_abyssi.faa and t_maritima.faa (using ssh or enter afp://bbcxsrv.biotech.uconn.edu in finder -> Go -> connect to server)

(check options for blastall and formatdb)

```
formatdb -i p_abyssi.faa -o T -p T
blastall -i t_maritima.faa -d p_abyssi.faa -o
   blast.out -p blastp -e 10 -m 8 -a2

./extract_lines.pl blast.out
```

Perl script that only retains the first hit and gets rid of comment lines
sftp results
load into spreadsheet
sort data, do histogram …
the extract_lines.pl script is here (you can sftp it into your account, you'll need to chmod 755 extr*.pl afterwards)

Assignment for Wednesday

1) On the computer that you plan to use for your project set up a connection (or connections) to bbcxsrv1 that allows you
(a) ssh to the server using a commandline interface
(b) allows you to drop and drag files from your computer to the server.

2) check that your vi editor on bbcxsrv1 is set up to have context dependent coloring (do this, even if you don't plan to use vi on the server!).

3) if you do not want to use vi, install an editor on your computer that provides context dependent coloring.

4) Create first Perl Program- "Hello, world!" [make file executable using chmod 755 *.pl]

```
#!/usr/bin/perl -w
print ("Hello, world! \n");
```

What happens if you leave out the new line character?
You can run the program by typing ./program_name.pl, if the file containing the program is an executable.

5) Read chapter 1 of "Learning Perl". HERE (pay attention on pages 12-15)