

# MCB 5472

BLAST, unix, Perl  
continued

*J. Peter Gogarten*

Office: *BPB 404*

phone: *860 486-4061,*

Email: *[gogarten@uconn.edu](mailto:gogarten@uconn.edu)*

## Discussion of Old Assignment

1) On the computer that you plan to use for your project set up a connection (or connections) to bbcxsr1 that allows you

- (a) ssh to the server using a command line interface
- (b) allows you to drop and drag files from your computer to the server.

2) check that your vi editor on bbcxsr1 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) Read through pages 53-61 of the [Unix and Perl Primer for Biologists](#)

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

```
#!/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 made executable (using `chmod u+x *.pl`).

From Lab exercises:

Which option turns off the low complexity filter? **-F F**

Which option, and which setting, sets the word size to 2? **-W 2**

Which option allows to use two processors? **-a 2**

# 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](#)

<code>pwd</code>	<code>ps</code>
<code>ls</code>	<code>ps aux</code>
<code>ls -l</code>	<code>rm</code>
<a href="#"><code>chmod</code></a>	<code>more</code>
<code>chmod a+x blastall.sh</code>	<code>cat</code>
<code>chmod 755 *.sh</code>	<code>vi (text editor)</code>
<code>cd</code>	<code>ps</code>
<code>cd ..</code>	<code>ps aux</code>
<code>cd \$HOME</code>	<code>ssh</code>
<code>passwd</code>	<code>sftp</code>

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@bbcxsrv1), 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 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 transfers 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 replaces 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 <http://lionel.kr.hs-niederrhein.de/~dalitz/data/software/macosx/ConvertNewlines.zip>. 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 cursor  
esc leaves the edit mode  
i enters the edit mode and inserts before the cursor  
a 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 **\$**scalars, **@**rarrays, or **%**ashes

**Scalars:** floating point numbers, integers,  
non decimal integers, strings

Scalar variables 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  
"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)
```

Do “Hello world” example (class 1) using a variable!



# 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 biologists often use homology as synonymous with similarity or 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

# 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 =  $4 \times 10^{780}$  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  $5 \times 10^{17}$  seconds or  $5 \times 10^{29}$  picoseconds.

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

# 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).

# 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 [TIG 2000](#) for more discussion.

## Types of Blast searching

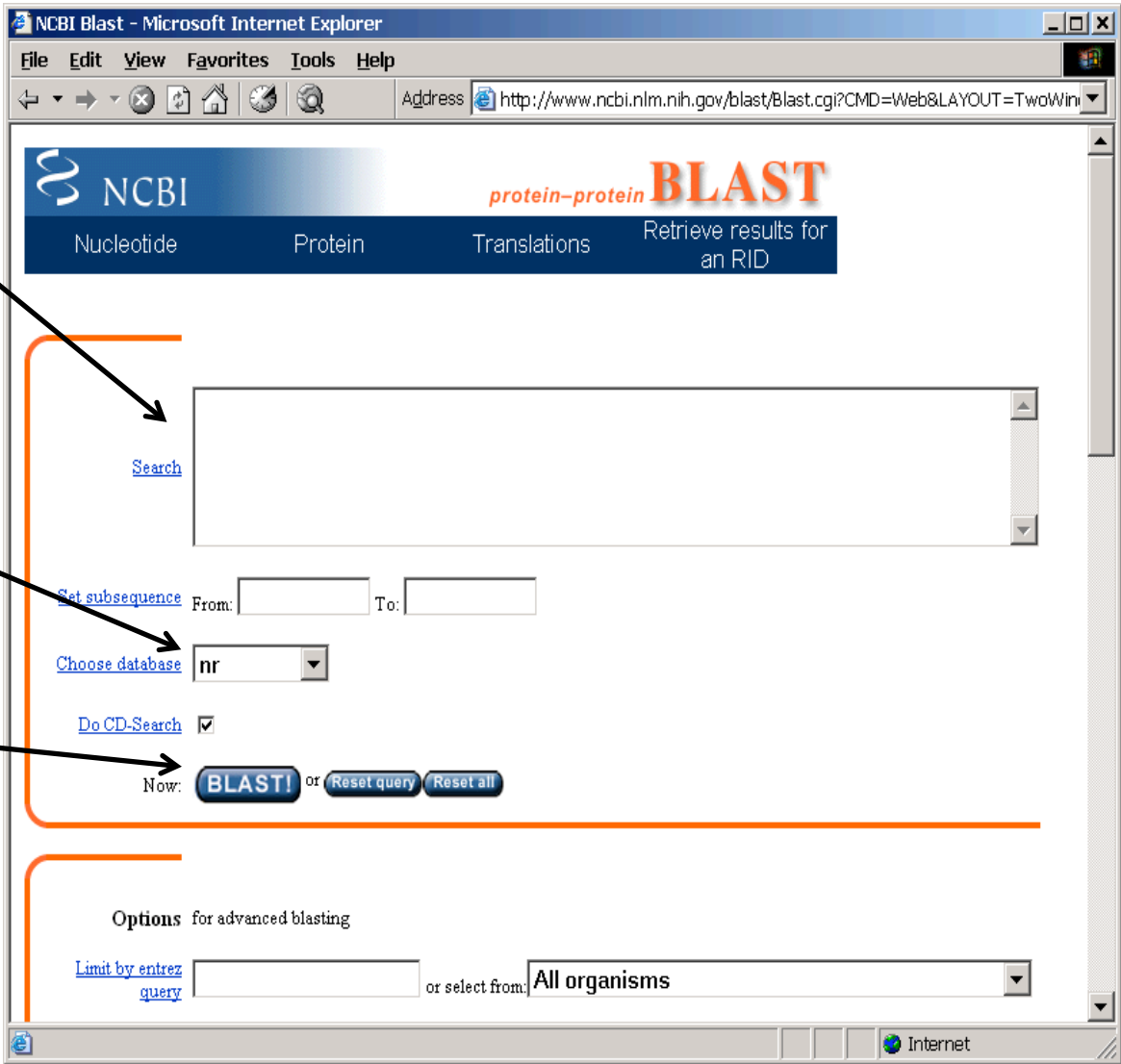
- `blastp` compares an amino acid query sequence against a protein sequence database
- `blastn` compares a nucleotide query sequence against a nucleotide sequence database
- `blastx` compares the six-frame conceptual protein translation products of a nucleotide query sequence against a protein sequence 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.

# Routine BlastP search

FASTA formatted text  
or Genbank ID#

Protein database

Run



# BlastP parameters

Restrict by taxonomic group

Filter repetitive regions

Statistical cut-off

Size of words in look-up table

Similarity matrix (cost of gaps)

Options for advanced blasting

[Limit by entrez query](#)  or select from: **All organisms**

[Composition-based statistics](#)

[Choose filter](#)  Low complexity  Mask for lookup table only  Mask lower case

[Expect](#)

[Word Size](#)

[Matrix](#) **BLOSUM62** Gap Costs **Existence: 11 Extension: 1**

[PSSM](#)

[Other advanced](#)

[PHI pattern](#)



# 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 =  $1 \times 10^{-20}$  = strict match between protein domains

## 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  $[0,\text{infinity})$ .

For small values  $E=P$

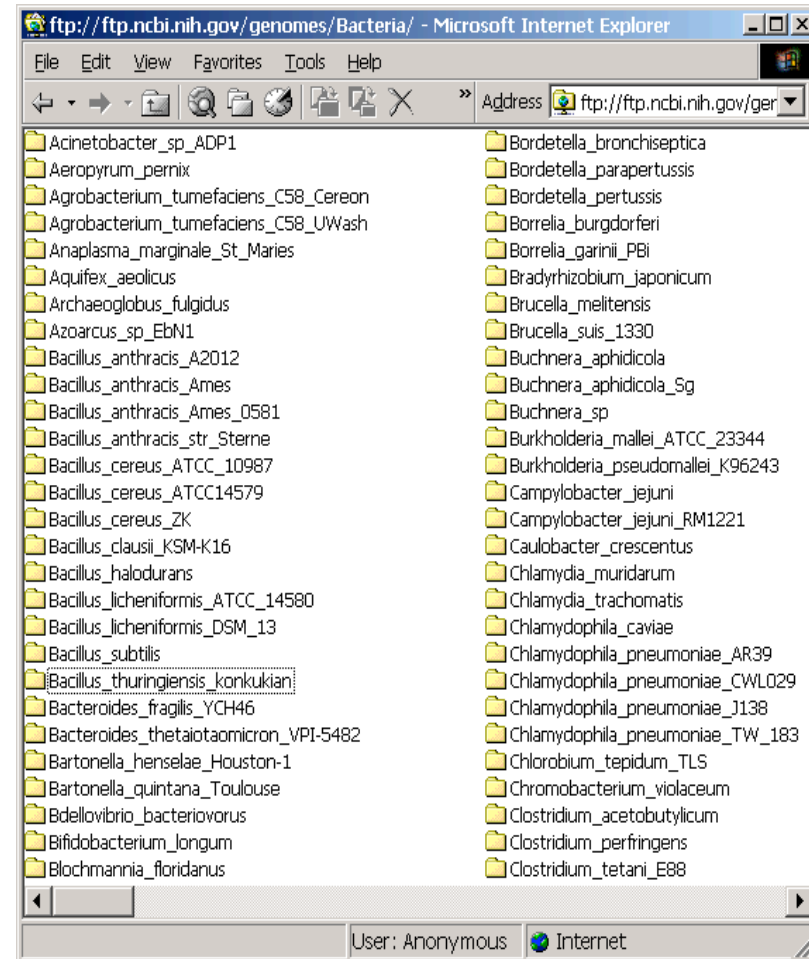
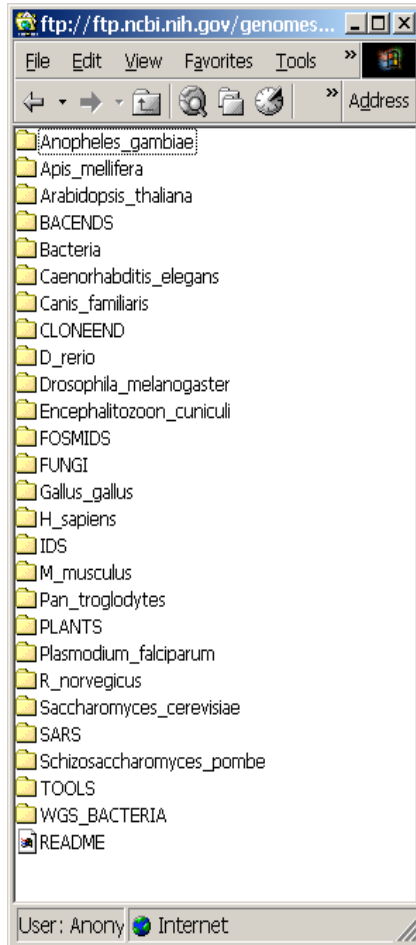
**Problem:** If you do 1000 blast searches, you expect one match due to chance with a P-value of 0.0001

“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/`

# More databases



# And more databases

## BLAST the Environmental Samples data

The data sources:

**Sargasso Sea Environmental Samples:** The Institute for Biological Energy Alternatives.

Venter, J.C., et al., *Environmental Genome Shotgun Sequencing of the Sargasso Sea*, [Science](#), 2004 Apr 2,304(5667):66-74.

**Mine Drainage Environmental Samples:** DOE Joint Genome Institute.

Tyson, G.W., et al., *Community structure and metabolism through reconstruction of microbial genomes from the environment*, [Nature](#), 2004 Mar 4;428(6978):37-43.

**Database:**

<b>Nucleotide:</b>
Sargasso Sea
Mine Drainage
Both Sample Sets
<b>Proteins:</b>
Sargasso Sea

**Program:**   MegaBlast

*Enter an accession, GI, or a sequence in FASTA format:*

*Optional parameters:*

[Expect](#)  [Filter](#)  [Descriptions](#)  [Alignments](#)

[Advanced options:](#)

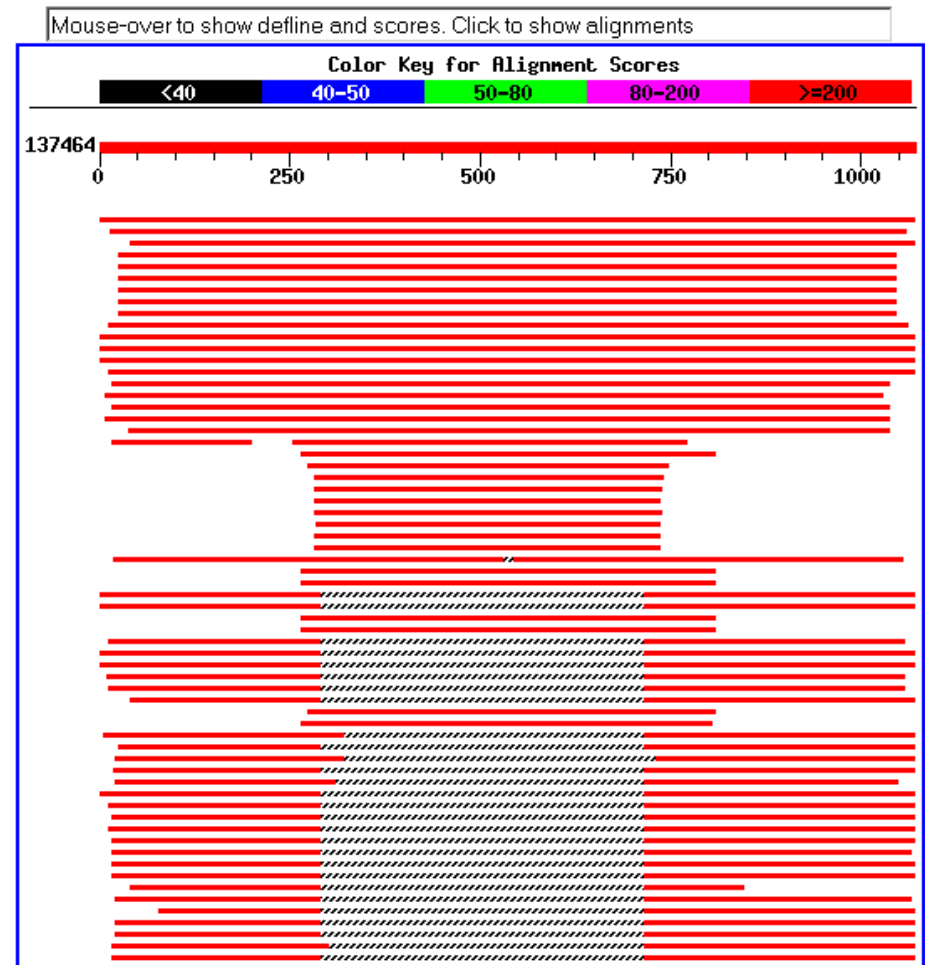
# Example of web based BLAST

program: **BLASTP**

sequence: vma1 gi:  
137464

**BLink** provides similar  
information

## Distribution of 823 Blast Hits on the Query Sequence





# Effect of low complexity filter

```
Query: 506 RATEFVDSRDTSLMERVTEYAERKLNLCAEYKDRKEPQVAKTIVNLYSKVVRGNGIRNNLNT 565
      RATEFVDSRDTSLMERVTEYAERKLNLCAEYKDRKEPQVAKTIVNLYSKVVRGNGIRNNLNT
Sbjct: 481 RATEFVDSRDTSLMERVTEYAERKLNLCAEYKDRKEPQVAKTIVNLYSKVVRGNGIRNNLNT 540

Query: 566 ENPLWDAIVGLGFLKDGVKNIPSFLSTDNIGTRETFLAGLIDSDGYVTDEHGKATIKTI 625
      ENPLWDAIVGLGFLKDGVKNIPSFLSTDNIGTRETFLAGLIDSDGYVTDEHGKATIKTI
Sbjct: 541 ENPLWDAIVGLGFLKDGVKNIPSFLSTDNIGTRETFLAGLIDSDGYVTDEHGKATIKTI 600

Query: 626 HTSVRDGLVSLARSLGLVSVNAEPAKVDMMNGTKHKISYAIYMSGGDVLLNVLSKCAGSX 685
      HTSVRDGLVSLARSLGLVSVNAEPAKVDMMNGTKHKISYAIYMSGGDVLLNVLSKCAGS
Sbjct: 601 HTSVRDGLVSLARSLGLVSVNAEPAKVDMMNGTKHKISYAIYMSGGDVLLNVLSKCAGSK 660

Query: 686 XXXXXXXXXXXXXECRGFYFELQELKEDDYGITLSDDSDHQFLLANQVVVHNCGERGNEM 745
      ECRGFYFELQELKEDDYGITLSDDSDHQFLLANQVVVHNCGERGNEM
Sbjct: 661 KFRPAPAAAFARECRGFYFELQELKEDDYGITLSDDSDHQFLLANQVVVHNCGERGNEM 720

Query: 746 AEVLMEFPPELYTEMSGTKPEIMKRTTLVANTSNNMPVAAREASIYTGITLAEYFRDQGKNV 805
      AEVLMEFPPELYTEMSGTKPEIMKRTTLVANTSNNMPVAAREASIYTGITLAEYFRDQGKNV
Sbjct: 721 AEVLMEFPPELYTEMSGTKPEIMKRTTLVANTSNNMPVAAREASIYTGITLAEYFRDQGKNV 780
```

**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

# Formatting a custom database

Format sequence data into Fasta format

Example of Fasta format:

```
>sequence 1
```

```
AAATGCTTAAAAA
```

```
>sequence 2
```

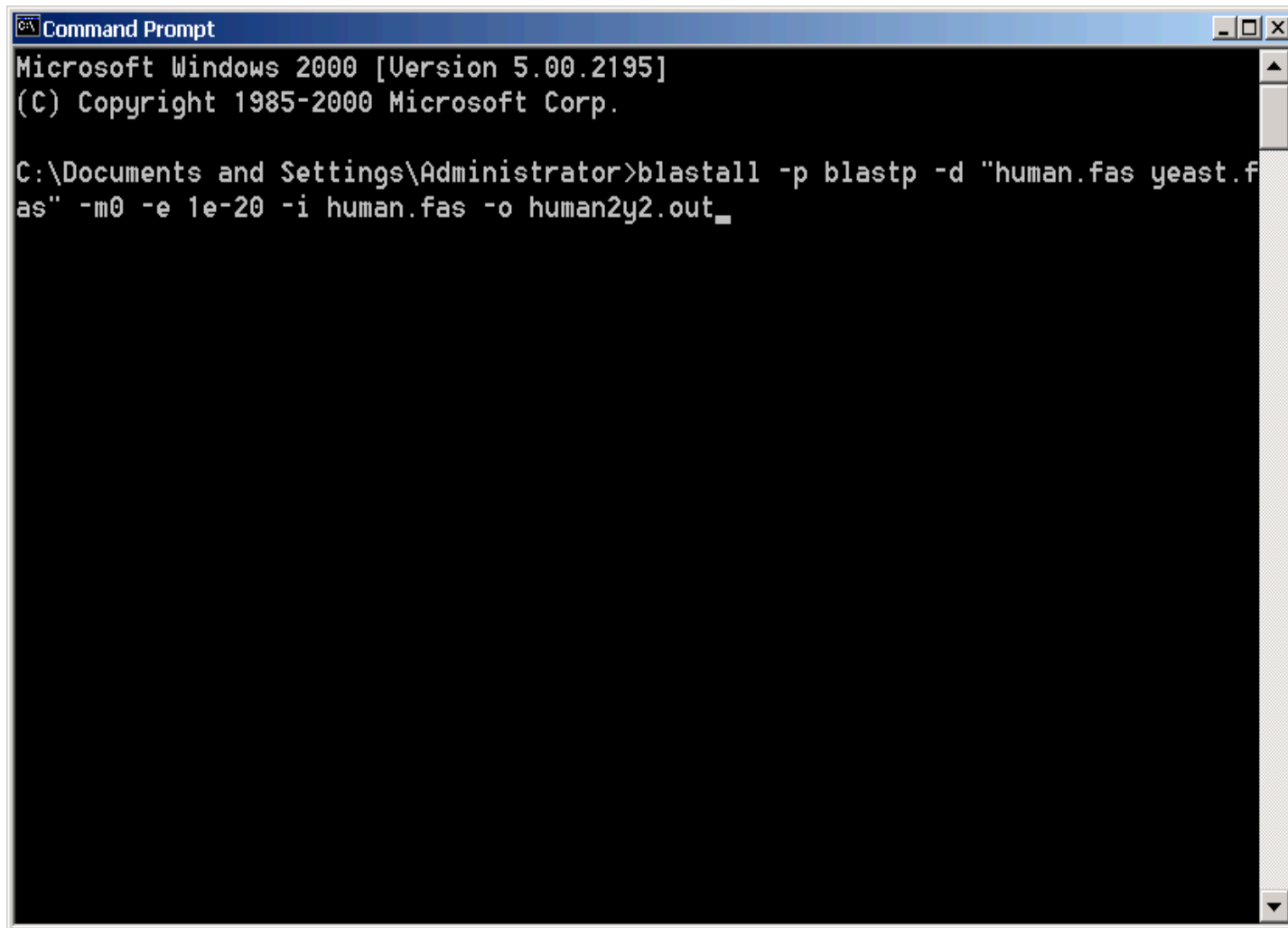
```
AAATTGCTAAAAGA
```

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)

## BlastP search of custom database



```
Command Prompt
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\Documents and Settings\Administrator>blastall -p blastp -d "human.fas yeast.f
as" -m0 -e 1e-20 -i human.fas -o human2y2.out_
```

# Example demo SSH to bbcxsrvc.biotech.uconn.edu

- A) Move files to bbcxsrvc p\_abyssi.faa and t\_maritima.faa (using ssh or enter afp://bcxsrvc.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)

vi blast.out, extract\_lines.pl, check spreadsheet

do histogram example, discuss % id, #id residues, alignment lengths.

## Assignment for Wednesday

- 1) Read through the Perl scripts [extract\\_lines.pl](#) and [extract\\_lines\\_mod.pl](#)
- 2) Why does the first of these get along without `chomp ($line);` (chomp is a built-in command in Perl to remove a trailing newline, if any, from a string)
- 3) Write a short Perl script that calculates the circumference of a circle given a radius provided by the user. (see exercises 1-4 chapter 2 in Learning Perl). (One set of answers is given in Appendix A of the book)
- 4) Do “Hello world” example (class 1) using a variable!

From Learning Perl:

“Each time you use `<STDIN>` in a place where a scalar value is expected, Perl reads the next complete text line from standard input (up to the first newline) and uses that string as the value of `<STDIN>`. Standard input can mean many things; unless you do something uncommon, it means the keyboard of the user who invoked your program (probably you). If there's nothing waiting for `<STDIN>` to read (typically the case unless you type ahead a complete line), the Perl program will stop and wait for you to enter some characters followed by a newline (return).\*

Example (~perl/class2/demo.pl)

```
#!/usr/bin/perl -w -s  
  
print "Enter a number:\n" ;  
chomp(my $input = <STDIN>);  
my $squared=$input**2;  
print "the input squared is $squared\n";
```

Go through [class2.pl](http://gogarten.uconn.edu/mcb5472_2010/class2.pl) [http://gogarten.uconn.edu/mcb5472\\_2010/class2.pl](http://gogarten.uconn.edu/mcb5472_2010/class2.pl)

# For next Monday:

- 1) What is the difference between a compiler and an interpreter?
- 2) When is it useful to make a script executable, when not?
- 3) What is the value of `$i` after each of the following operations?

```
$i=1;  
$i++;  
$i *= $i;  
$i .= $i;  
$i = $i/11;  
$i = $i . "score and" . $i+3;
```

First make a guess, then test your prediction using a script.

- 4) If `$a = 2` and `$b=3`, what is the type and values of the scalar stored in `$c` after each of the following statements:

```
$c = $a + $b;  
$c = $a / $b;  
$c = $a . $b;  
$c = "$a + $b";  
$c = '$a + $b';
```

First make a guess, then test your prediction using a script.



# Psi-Blast: Detecting structural homologs

Psi-Blast was designed to detect homology for highly divergent amino acid sequences

Psi = position-specific iterated

Psi-Blast is a good technique to find “potential candidate” genes

Example: Search for Olfactory Receptor genes in Mosquito genome

Hill CA, Fox AN, Pitts RJ, Kent LB, Tan PL, Chrystal MA, Cravchik A, Collins FH, Robertson HM, Zwiebel LJ (2002) G protein-coupled receptors in *Anopheles gambiae*. *Science* 298:176-8

# Psi-Blast Model

Model of Psi-Blast:

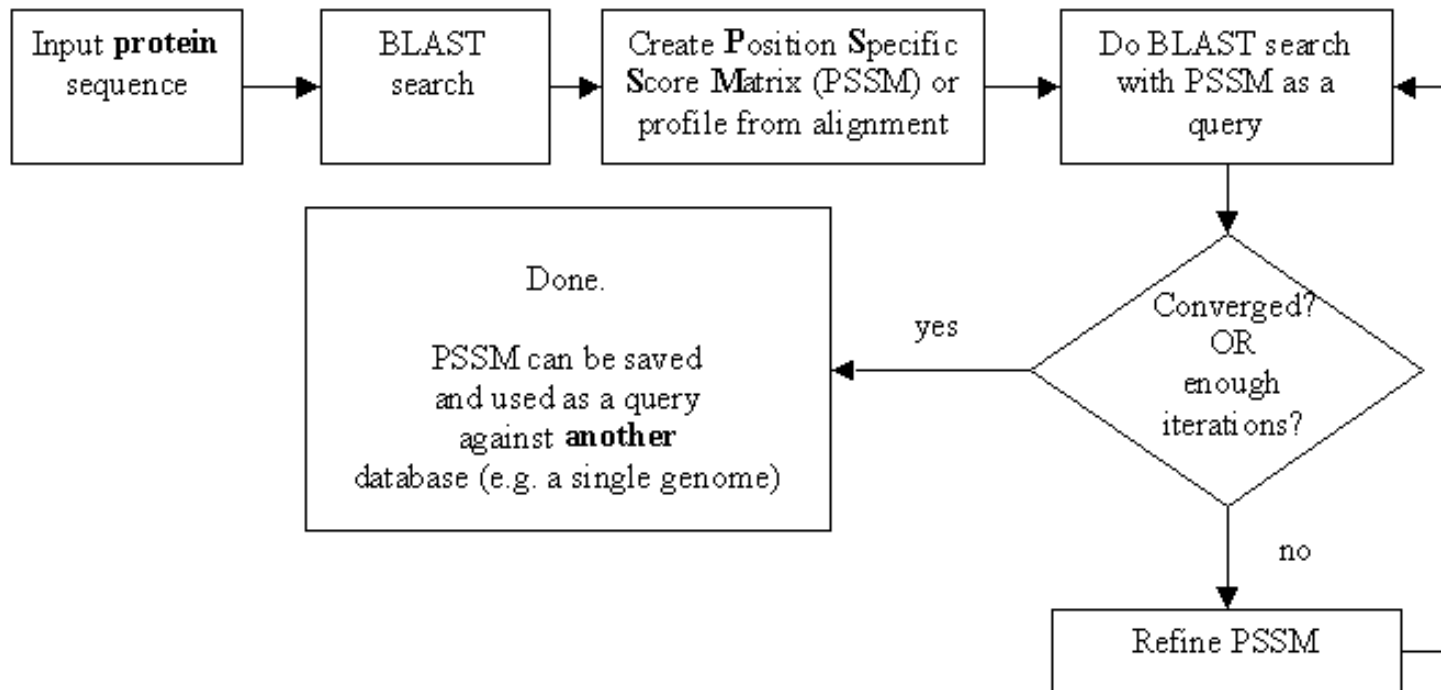
1. Use results of gapped BlastP query to construct a multiple sequence alignment
2. Construct a position-specific scoring matrix from the alignment
3. Search database with alignment instead of query sequence
4. Add matches to alignment and repeat

Similar to Blast, the E-value in Psi-Blast is important in establishing matches

E-value defaults to 0.001 & Blosom62

Psi-Blast can use existing multiple alignment - particularly powerful when the gene functions are known (prior knowledge) or use RPS-Blast database

# PSI BLAST scheme



# Position-specific Matrix

POS	PROBE	CONSENSUS	PROFILE																				
			A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y	+/-
1	EGVLL	V	3	-2	3	4	0	4	-1	3	-1	4	4	1	1	1	-2	1	2	6	-6	-2	9
2	LLSPL	L	2	-2	-2	-1	3	0	-1	3	-1	6	5	-1	3	0	-1	3	1	4	1	-1	9
3	VVVVV	V	2	2	-2	-2	2	2	-3	11	-2	8	6	-2	1	-2	-2	0	2	15	-9	-1	9
4	KEAT	A	6	-2	5	6	-5	4	1	0	5	-2	0	3	3	3	1	3	6	0	-6	-4	9
5	APLP	P	6	-1	0	1	-2	2	0	1	0	2	2	0	8	2	0	2	2	3	-5	-4	9
6	GGGGG	G	7	1	7	5	-6	15	-1	-3	0	-4	-3	4	3	2	-3	6	4	2	-11	-7	9
7	SSSQE	D	4	-1	7	7	-6	7	2	-2	2	-3	-2	4	3	6	1	6	2	-1	-6	-5	9
8	SSTP	S	4	4	2	2	-4	4	-1	0	2	-3	-2	2	7	0	1	10	6	0	-2	-4	9
9	VLVA	V	5	0	-1	-1	3	1	-2	7	-2	7	6	-1	1	-1	-3	0	2	10	-5	-1	9
10	KRRS	R	0	-1	1	1	-5	0	2	-2	8	-3	1	3	3	3	10	5	1	-2	7	-5	9
11	MLII	I	0	-2	-3	-2	7	-3	-3	11	-1	11	10	-2	-2	-1	-2	-2	1	9	-3	1	9
12	SSTS	S	4	6	2	2	-3	5	-1	0	2	-3	-2	3	4	-1	1	12	6	0	0	-4	9
13	CCCC	C	3	15	-5	-5	-1	2	-1	3	-5	-8	-6	-3	1	-6	-3	7	3	3	-13	10	9
14	KSQR	K	1	-2	3	3	-6	1	3	-2	7	-3	0	3	3	5	7	4	1	-2	2	-5	9
15	AAGS	A	10	3	4	3	-5	8	-1	1	-2	-1	3	4	1	-2	7	4	2	-6	-4	9	
16	TSDS	S	4	3	5	4	-5	6	0	0	2	-3	-2	4	3	1	1	9	6	0	-3	-4	9
17	GGSQ	G	5	1	6	5	-6	9	1	-2	1	-3	-2	4	3	4	0	6	3	0	-6	-6	9
18	YFLS	F	-1	2	-4	-3	9	-3	0	4	-3	6	3	-1	-3	-3	-3	1	-1	2	7	7	9
19	TTRL	T	1	-2	0	1	0	0	0	2	2	2	3	1	1	1	3	1	7	2	1	-2	9
20	FF.L	F	-2	-3	-6	-4	10	-4	-1	6	-4	9	6	-3	-4	-4	-3	-2	-1	3	7	8	4
21	SS.D	S	3	2	5	4	-4	5	0	-1	2	-3	-2	4	3	1	1	8	2	-1	-2	-3	4
22	S.S	S	2	3	1	1	-2	3	-1	0	1	-2	-1	2	2	0	1	8	2	0	1	-2	4
23	. . . G	G	2	0	2	1	-2	4	0	0	0	-1	-1	1	1	1	-1	2	1	1	-3	-2	4
24	. . . D	D	1	-1	4	3	-2	2	1	0	1	-1	2	1	2	0	1	1	0	-3	-1	4	
25	. . . G	G	2	0	2	1	-2	4	0	0	0	-1	-1	1	1	1	-1	2	1	1	-3	-2	4
26	. AGN	A	6	0	4	3	-4	6	1	-1	1	-2	-1	5	2	2	-1	3	3	1	-5	-3	4
27	YNYT	Y	0	5	0	-1	5	-1	2	1	-1	0	-1	4	-3	-2	-2	0	3	0	3	6	4
28	EDDY	D	2	-2	9	8	-3	3	4	-1	1	-3	-2	5	-1	4	-1	1	1	-1	-6	0	9
29	LMAL	L	3	-5	-3	-1	6	-1	-2	6	-1	10	10	-2	0	0	-2	-1	0	6	-1	0	9
30	YNAW	N	4	1	3	2	0	2	3	-1	1	-1	8	0	1	-1	2	1	-1	-1	2	9	
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
48	SGNS	S	4	3	5	3	-4	7	0	-2	2	-4	-3	6	3	1	0	10	3	0	-2	-4	9
49	SSNY	S	2	5	2	1	1	2	1	0	1	-2	-2	5	1	-1	0	8	1	-1	3	1	9

FIG. 1. The concept of a profile. (a) A flow diagram of profile analysis. (b) A 49-residue sample profile for the immunoglobulin variable-region domain, generated from the four-probe sequences shown at the left (see Fig. 2b for details). The profile is shown in the box. The rightmost column of the profile gives the penalty for insertion/deletion (+/-). Positions 31-47 of the profile are omitted from the figure for clarity. Notice that where gaps appear in some of the probe sequences, the insertion/deletion penalty is lower than elsewhere.

M Gribskov, A D McLachlan, and D Eisenberg (1987) Profile analysis: detection of distantly related proteins. PNAS 84:4355-8.

# Psi-Blast Results      Query: 55670331 (intein)

NEW	<input checked="" type="checkbox"/>	<a href="#">gi 6706000 dbj BAAU6142.2 </a>	DNA-dependent DNA polymerase [Pyrococ...	<a href="#">48</a>	7e-04	
NEW	<input checked="" type="checkbox"/>	<a href="#">gi 2708498 gb AAB92484.1 </a>	ribonucleotide reductase homolog [Baci...	<a href="#">48</a>	7e-04	
NEW	<input checked="" type="checkbox"/>	<a href="#">gi 50812254 ref NP_389888.2 </a>	hypothetical protein BSU20060 [Baci...	<a href="#">48</a>	8e-04	<b>G</b>
NEW	<input checked="" type="checkbox"/>	<a href="#">gi 7475800 pir  A69927</a>	ribonucleoside-diphosphate reductase (alp...	<a href="#">48</a>	8e-04	
NEW	<input checked="" type="checkbox"/>	<a href="#">gi 15211863 emb CAC51100</a>	bun...	<a href="#">46</a>	0.002	
NEW	<input checked="" type="checkbox"/>	<a href="#">gi 57867420 ref YP_18907</a>	hat...	<a href="#">46</a>	0.003	<b>G</b>
NEW	<input checked="" type="checkbox"/>	<a href="#">gi 14590941 ref NP_143015.1 </a>	ATP-dependent helicase LHR [Pyrococ...	<a href="#">46</a>	0.003	<b>G</b>

link to sequence [here](#),  
check BLink 😊

Run PSI-Blast iteration 3

## Sequences with E-value WORSE than threshold

<input type="checkbox"/>	<a href="#">gi 14590539 ref NP_142607.1 </a>	secretory protein kinase [Pyrococcu...	<a href="#">44</a>	0.006	<b>G</b>
<input type="checkbox"/>	<a href="#">gi 45513096 ref ZP_00164662.1 </a>	COG1372: Intein/homing endonuclea...	<a href="#">44</a>	0.009	
<input type="checkbox"/>	<a href="#">gi 156250075 ref YP_18907</a>		<a href="#">44</a>	0.009	<b>G</b>

# PSI BLAST and E-values!

Psi-Blast is for finding matches among divergent sequences (position-specific information)

**WARNING:** For the nth iteration of a PSI BLAST search, the E-value gives the number of matches to the profile NOT to the initial query sequence! The **danger** is that the profile was corrupted in an earlier iteration.

## PSI Blast from the command line

Often you want to run a PSIBLAST search with two different databanks - one to create the PSSM, the other to get sequences:

To create the PSSM:

```
blastpgp -d nr -i subI -j 5 -C subI.ckp -a 2 -o subI.out -h 0.00001 -F f
```

```
blastpgp -d swissprot -i gamma -j 5 -C gamma.ckp -a 2 -o gamma.out -h 0.00001 -F f
```

Runs a 4 iterations of a PSIBlast

the -h option tells the program to use matches with  $E < 10^{-5}$  for the next iteration, (the default is  $10^{-3}$ )

-C creates a checkpoint (called subI.ckp),

-o writes the output to subI.out,

-i option specifies input as using subI as input (a fasta formatted aa sequence).

The nr databank used is stored in /common/data/

-a 2 use two processors

(It might help to use the node with more memory (017)

(command is `ssh node017`)

## To use the PSSM:

```
blastpgp -d /Users/jpgogarten/genomes/msb8.faa -i subI -a 2 -R  
subI.ckp -o subI.out3 -F f
```

```
blastpgp -d /Users/jpgogarten/genomes/msb8.faa -i gamma -a 2 -R  
gamma.ckp -o gamma.out3 -F f
```

**Runs another iteration of the same blast search, but uses the databank** /Users/jpgogarten/genomes/msb8.faa

- R tells the program where to resume
- d specifies a different databank
- i input file - same sequence as before
- o output\_filename
- a 2 use two processors



## More on blastall:



### **BLAST**

by Joseph Bedell; Ian Korf; Mark Yandell

Publisher: **O'Reilly**

Pub Date: **July 2003**

ISBN: **0-596-00299-8**

Pages: **360**

Slots: **1.0**

**available at safari books online**

<http://proquestcombo.safaribooksonline.com/>

Installation instructions and info on parameters at the NCBI:

<http://www.ncbi.nlm.nih.gov/staff/tao/URLAPI/blastall/>

<ftp://ftp.ncbi.nlm.nih.gov/blast/documents/formatdb.html>

<ftp://ftp.ncbi.nlm.nih.gov/blast/documents/blast.html>

<ftp://ftp.ncbi.nlm.nih.gov/blast/documents/blastpgp.html>

<ftp://ftp.ncbi.nlm.nih.gov/blast/documents/fastacmd.html>

<ftp://ftp.ncbi.nlm.nih.gov/blast/documents/>

<http://www.bioinformatics.ubc.ca/resources/tools/blastall>

<http://en.wikipedia.org/wiki/BLAST>