Perl Programming Fundamentals for the Computational Biologist

Lab 2

Marine Biological Laboratory, Woods Hole
“Advances in Genome Technology and Bioinformatics”
Fall 2004

Andrew Tolonen
Chisholm lab, MIT

Goal of the lab: In the previous lab, you read a FASTA file of genes, built a hash, and searched each gene sequence for a pattern. Now you can ask "If I BLAST each of those genes against the swissprot database, what will the top hit be?" Here is what you will do:

copy makeHash.plx to create a new script called blastGenes.plx that will accomplish the following:

- BLAST the sequence of each of the genes in your hash from last time against a local copy of the swissprot database.
- read the output files to grab the top hit for each gene.
- print the top hit for each the to an output file

Prelab: download the exercises, input files, and their solutions from cyano.mit.edu. I really hope it will work this time!

1. make a directory for lab 2
   ```
   % mkdir Lab2
   ```

2. cd into that directory
   ```
   % cd Lab2
   ```

3. Open your browser

4. Type "ftp://PerlClass@cyano.mit.edu" into the browser window
   password: perlmbl

5. Retrieve the archive by "save target as" for Lab2.tar.gz into your Lab2 directory

6. Unzip the archive by running the following command at the shell prompt. (Note: the % sign signifies the shell prompt. It may look different on your computer).
   ```
   % gunzip Lab2.tar.gz
   ```

7. Untar the archive
   ```
   % tar -xvf Lab2.tar
   ```

now if you 'ls', you should see a list of files in your current working directory:

```
% ls
genes.faa  makeHash.plx perlLab2.pdf  Solutions
```

Step 1. Your goal is to modify makeHash.plx to BLAST the genes in your hash. The first step is to print the gene name and its sequence to an output file. Replace the print statements with these instructions. Run the script. Open the file “blast.faa”. Does it contain a gene entry? If you are having problems, see /Solutions/exercise1.plx

```
foreach $x (keys(%genes))
{
 # first print the gene out to a file for blasting
 open (BLAST, ">blast.faa") or die "cant open blast file\n";
 print BLAST "$x\n";  # print out the gene name line
 print BLAST "$genes{$x}\n";  # print out the sequence
 close BLAST;
}
```

Step 2. Now you are ready to BLAST the gene sequences. To do this, use backticks to make a system call to the blastall executable. blastall can take many different arguments. For your purposes this will work:

```
`blastall -m 8 -p blastp -d swissprot -i ./blast.faa -o ./
blast.out`;
```

Here is what the blastall arguments mean:

Argument	Meaning
-m 8	make tabular output
-p blastp	BLAST protein seqs
-d swissprot	BLAST against swissprot database
-i ./blast.faa	specify input file

-o ./blast.out specify output file

When you insert the blastall statement in the foreach loop after the open BLAST statements, you will be BLASTing each gene. After running the script, examine the contents of ./blast.out. Does it contain tabular blast output? See /Solutions/exercise2.plx for the solution.

Note: You can use backticks to execute any linux command line utility from within your Perl script. If you want, try it out!

Step 3. Grab the top hit from each BLAST search and print it to a separate file. This is easy because the information about the top hit is the first line of the output file. See Solutions/exercise3.plx for the solution.

```
# now grab the top hit from the blast output file
open (IN, "<./blast.out") or die "cant open blast output
file\n";

$line = <IN>;  # grab the first line of the file (contains
               # top hit)
push(@tophits, $line); # push the tophit info into an array
close IN;
```

Now your foreach loop is done! It should contain code to execute the following commands:

```
foreach $x (keys(%genes))
{
  # 1. print a gene entry to a file for blasting
  # 2. BLAST the gene against swissprot
  # 3. grab the top hit and print it to a file
}

# now print each element of the @tophits array to
# an output file
```

You have an array @tophits that contains the information about the top hit of each blast search. Print the @tophits array to an output file. From outside the foreach loop.

```
# print each element in @tophits to ./tophits.txt
open (HITS, ">./tophits.txt") or die "cant open hits
file\n";
foreach $x (@tophits)
{
 print HITS "$x\n";
}
close HITS;
```

Open the file ./tophits.txt. Does it contain tabular BLAST hit information for the top hit of each gene?

Extra Credit:

1. What if you only wanted to BLAST the genes that contained a given motif?

```
$motif = “STG”;

foreach $x (keys(%genes))
{
  if ($genes{$x} =~ /$motif/)
  {
    # blast the genes
  }
}

# print the top hits to an output file
```

Could you use regular expressions to make the motif more flexible.? ie. “A serine or a threonine followed by a Glycine or an Alanine, etc.”

2. You could use fastacmd to retrieve the sequences of the top hits.

Step 1. First of all, try running fastacmd from the command line. Open the file “tophits.txt” and write down the gene ID for the first top hit. In the line below, the gene ID is 38605646.

gene gi|38605646|sp||P02562_1 32.20 59 33 2 81 138 27 79 1.4 30.42

```
% fastacmd -d swissprot -s 38605646
```

Step 2. In order to run fastacmd on each top hit, you need to grab the gene ID number. You can do this using regular expressions:

```
$line =~ /gi\|(\d+)\|sp/;
```

Once you have grabbed the gene ID's try running fastacmd from within your Perl script.