ECS 124A Theory and Practice of Bioinformatics

Lab Assignment 3

Instructor: Ilias Tagkopoulos (iliast@ucdavis.edu)
TA: Linh Huynh (huynh@ucdavis.edu)

Key dates:
Due by: Tuesday 11/18/2014 before class.

Scope:
The goal of this lab is for you to (a) learn the principles behind BLAST and its variations, (b) learn how to use BLAST, (c) learn more about programming techniques and implement them in PERL, (d) make your own, simplified version of BLAST, that builds on the same principles, (e) perform a sequence alignment for RNA-Seq samples by using various tools (BONUS EXERCISE!)

Deliverables:

Hand in answers to all the exercises/questions. For exercise 3 hand in your code for each step and printouts/files of the results wherever it is applicable.

Grading:

Exercise 1: 20
Exercise 2: 30
Exercise 3: 100
Exercise 4: 40 (BONUS!)
TOTAL : 150 pts (190 possible)

Note: Bonus points will be awarded towards lost points in ANY exercise and in any homework set. You do not have to complete exercise 4 if you do not want to.

PART I: PERL and BLAST

Exercise 1. Regular expressions

Continue reading the second part of the Perl notes:

http://cs124.cs.ucdavis.edu/lectures/secondnotes.pdf

Do exercises 2.12, 2.13, 2.14 of those notes. In your lab report please reference then with the index one in front, i.e. exercise 2.12 will be 1.2.12, exercise 2.13 will be 1.2.13 and so forth.

Exercise 2. BLAST

In this part of the lab you will familiarize with BLAST. BLAST's main page is:

http://blast.ncbi.nlm.nih.gov/Blast.cgi

From there you can you to the BLAST handbook through help->NCBI Handbook:BLAST or just following this link:

http://www.ncbi.nlm.nih.gov/bookshelf/br.fcgi?book=handbook&part=ch16

Read some details about BLAST and answer the following questions:

Question 2.1. What scores and statistics BLAST use? What does each one means and how is it calculated?

Question 2.2. What is the difference between BLAST, BLAST 2, PSI-BLAST, FASTA? When should each of these been used (trade-offs)?

Question 2.3. Read:

http://www.ncbi.nlm.nih.gov/BLAST/tutorial/Altschul-1.html

And explain what is the p-value for the top match of the following query (that actually comes from the movie Jurassic Park) :

```
>DinoDNA from JURASSIC PARK  p. 103 nt 1-1200
GCGTTGCTGGCGTTTTTCCATAGGCTCCGCCCCCCTGACGAGCATCACAAAAATCGACGC
GGTGGCGAAACCCGACAGGACTATAAAGATACCAGGCGTTTCCCCCTGGAAGCTCCCTCG
TGTTCCGACCCTGCCGCTTACCGGATACCTGTCCGCCTTTCTCCCTTCGGGAAGCGTGGC
TGCTCACGCTGTACCTATCTCAGTTCGGTGTAGGTCGTTCGCTCCAAGCTGGGCTGTGTG
```

```
CCGTTCAGCCCGACCGCTGCGCCTTATCCGGTAACTATCGTCTTGAGTCCAACCCGGTAA
AGTAGGACAGGTGCCGGCAGCGCTCTGGGTCATTTTCGGCGAGGACCGCTTTCGCTGGAG
ATCGGCCTGTCGCTTGCGGTATTCGGAATCTTGCACGCCCTCGCTCAAGCCTTCGTCACT
CCAAACGTTTCGGCGAGAAGCAGGCCATTATCGCCGGCATGGCGGCCGACGCGCTGGGCT
GGCGTTCGCGACGCGAGGCTGGATGGCCTTCCCCATTATGATTCTTCTCGCTTCCGGCGG
CCCGCGTTGCAGGCCATGCTGTCCAGGCAGGTAGATGACGACCATCAGGGACAGCTTCAA
CGGCTCTTACCAGCCTAACTTCGATCACTGGACCGCTGATCGTCACGGCGATTTATGCCG
CACATGGACGCGTTGCTGGCGTTTTTCCATAGGCTCCGCCCCCCTGACGAGCATCACAAA
CAAGTCAGAGGTGGCGAAACCCGACAGGACTATAAAGATACCAGGCGTTTCCCCCTGGAA
GCGCTCTCCTGTTCCGACCCTGCCGCTTACCGGATACCTGTCCGCCTTTCTCCCTTCGGG
CTTTCTCAATGCTCACGCTGTAGGTATCTCAGTTCGGTGTAGGTCGTTCGCTCCAAGCTG
ACGAACCCCCCGTTCAGCCCGACCGCTGCGCCTTATCCGGTAACTATCGTCTTGAGTCCA
ACACGACTTAACGGGTTGGCATGGATTGTAGGCGCCGCCCTATACCTTGTCTGCCTCCCC
GCGGTGCATGGAGCCGGGCCACCTCGACCTGAATGGAAGCCGGCGGCACCTCGCTAACGG
CCAAGAATTGGAGCCAATCAATTCTTGCGGAGAACTGTGAATGCGCAAACCAACCCTTGG
CCATCGCGTCCGCCATCTCCAGCAGCCGCACGCGGCGCATCTCGGGCAGCGTTGGGTCCT
```

Go through the calculations and see if the p-value matches the one reported by BLAST. If you have to make any assumptions please state them explicitly.

Exercise 3. Create PERL-BLAST

In this exercise we will create a toy blast program that we will call PERL-BLAST. It will use the same principles of finding words (k-meres) and extending them to find alignments. The first step is to get four k-mere programs from Smartsite, under resources->Labs>Lab3, file kmer_v123_and_kmerfirst.txt

Start from kmer1.pl and work your way to kmerfirst.pl, executing each code and understanding how it works. The new elements of Perl include the **substr** function, the **length** function, two-dimensional **hashes**, the **defined** function, building up a **list as a hash value**, among others. You should read Johnson on these elements of Perl. The program **kmerfirst.pl** finds the first position of each of the different kmers of length k. This program will be the starting point for your PERL-BLAST program. Your program should do the following things:

1. Read in from file a query string Q.

2. For k = 4, use program **kmerfirst.pl** to find the first location of each different k-mer in Q.

3. Successively read in one string at a time from a file called **perlblastdata.txt** that is located again under the resources>Labs>Lab 3.

When you hand in your lab 3 report, be sure to show the program working at least on this toy data file. When a string S is read in, scan through its 4-mers, using the same hash as before. For this, extract and adapt what you need from kmerfirst.pl

4. Whenever (if ever) a 4-mer in S is determined to be in Q, extract the location of the first occurrence of that 4-mer in Q. Then put the characters of Q and S in arrays (as we did in needleman.pl) so that you can examine individual characters. Then scan left from the k-mer in Q and in S, as long as you find matching characters. Repeat to the right. Let L denote the length of the whole match obtained in this way. If L is greater than 10, then print a message that a good HSP has been found between Q and S, and print S. Notice that the same HSP gets reported multiple times. Explain why that happens.

5. Now we will fix that by not reporting an HSP multiple times. We can do it using a hash called **stringhash**, like this: Whenever PERL-BLAST finds a reportable substring in a database, starting at position $i, say, in the database string, it looks to see if $stringhash{$i} is defined. If so, it does not report the string again. Otherwise it assigns the string to $stringhash{$i} and does report the string. Implement this change in the program.

6. We would like to process strings that are more than a single line long. So in the file each string will be held in consecutive lines, with strings separated by blank lines. That is like saying that each string is a paragraph instead of just a single line. To read in a whole paragraph, put the line

```
$/ = "";
```

somewhere in the program before the reading begins. Read about this on page 102 of Johnson or some other Perl reference.

7. In this final modification/version of our PERL-BLAST program, we will make it so that if a k-mer found in a database string is in the query string in multiple locations, then a search should be made from each occurrence of the kmer in the query string, spanning outward left and right of each occurrence. To do that use **kmer4.pl** that is found under the resources>Labs>Lab 3 (file kmer4.txt) to build up a list of all occurrence of each distinct kmer in the query string, and use it to implement this change. Use k = 4 and t = 7. Use your modified program on the perlblastdata to test it.

Congratulations, you made your own version of BLAST.

Exercise 4. RNA-Seq mapping

For this exercise, you will use a software suite that is called GALAXY to analyze five RNA-Seq datasets. The datasets are paired-end 50bp reads from adrenal and brain tissues (500Kb region of chromosome 19, chr19:3000000:3500000). You can find the datasets here:

https://usegalaxy.org/u/jeremy/p/galaxy-rna-seq-analysis-exercise

The datasets contain the transcriptional profiles in those tissues and our task is to find genes that are differentially expressed (DEG) in the sample conditions versus the control sample. We currently have 3 programs to perform this task: edgeR, Cuffdiff and DESeq. You are asked to do this analysis with each and finally report the results.

4. 1. (10 points) Using Cuffdiff from Galaxy, complete the tutorial that is found in the previous link and report the top 100 DEG, their fold-change (or score) and their p-values.

4. 2. (10 points) Perform the same analysis by using EdgeR, which can be found here: http://www.bioconductor.org/packages/release/bioc/html/edgeR.html

Again report the top 100 DEG based on EdgeR, their fold-change (or score) and their p-values.

4. 3. (10 points) Finally, perform the same analysis by running DESeq:

http://bioconductor.org/packages/release/bioc/html/DESeq.html

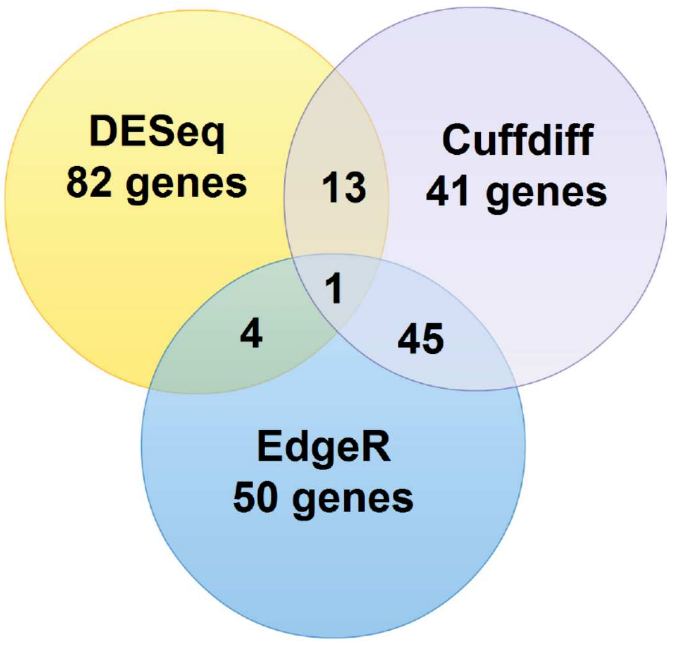
And (as you have guessed) report the top 100 DEG, their fold-change (or score) and their p-values. For EdgeR and DESeq, you can use HTSeq to process the alignment results (.bam files from TopHat on Galaxy):

http://www-huber.embl.de/users/anders/HTSeq/doc/overview.html

before you pass the results into EdgeR and DESeq.

4. 3. (10 points) Compare the results of the 100 DEG as generated by the three tools above. Create a single xls file that contains these genes (with their scores and p-values) and create a Venn Diagram with three sets (i.e. circles that overlap) to report the overlap of DEG in each case (as shown below). What do you observe? Why do they produce different results? What is the basis of each method and where can each be used?

NOTE: This exercise is not for the faint of heart! Remember this is a **bonus exercise** that you should only try after you complete the other 3 exercises and be ready to do a lot of trouble-shooting. Consult with the TA early on if you have any issues!



END OF LAB 3