**Homework Assignment 3**

*Due date and time: 11 AM, Wednesday, January 27th, 2016*

*You may work with other members of your group on homework assignments but what you write-up and turn in must be solely your individual intellectual contribution.*

*Please submit homework assignments in hard copy and electronic formats. Python scripts must be uploaded as \*.py files.*

**TASK 1:** *Comparisons between Sequence and Structural Alignments* (**5 points**)

Download the PDB files for bacterial and human thioredoxin just like we did in class. Perform a structural alignment using [DeepView v3.7](https://canvas.northwestern.edu/courses/30325/files/folder/Freeware) (for PCs) or [v4.1](http://spdbv.vital-it.ch/download/binaries/SPDBV_4.1.0_OSX.zip) (apparently, unlike PCs, this version works well and is not ‘buggy’ for Macs). Download the corresponding FASTA sequence files and perform sequence alignments using stretcher (that uses the N-W algorithm) in the **EMBOSS** suite (set output format: EMBOSS format). Use two different PAM and BLOSUM matrices each for scoring alignments and use the following gap opening and extension penalties: PAM70: 10/1; PAM250: 14/2; BLOSUM80: 10/1; BLOSUM62: 11/1. Comment on the accuracy of the alignments; do you see any correlation between segments that align well structurally that also appear to be aligned correctly by stretcher? Note that the *Iterative Magic Fit* algorithm in DeepView does not consider the underlying sequence information in order to generate structural alignments; similarly, stretcher has no access to structural information while generating optimal alignments. The former is deemed the ‘correct’ alignment while the latter generates a best guess based on the scoring matrix and gap penalties used.

**TASK 2:** *Searching Databases with BLAST*(**10 points**)

A yeast two-hybrid screen of a cDNA library leads to the discovery of a nucleotide sequence whose conceptual translation yields the following protein sequence:

MAPKQDPKPKFQEGERVLCFHGPLLYEAKCVKVAIKDKQVKYFIHYSGWNKNWDEWVPESRVLKYVDTNL

QKQRELQKANQEQYAEGKMRGAAPGKKTSGLQQKNVEVKTKKNKQKTPGNGDGGSTSETPQPPRKKRARV

DPTVENEETFMNRVEVKVKIPEELKPWLVDDWDLITRQKQLFYLPAKKNVDSILEDYANYKKSRGNTDNK

EYAVNEVVAGIKEYFNVMLGTQLLYKFERPQYAEILADHPDAPMSQVYGAPHLLRLFVRIGAMLAYTPLD

EKSLALLLNYLHDFLKYLAKNSATLFSASDYEVAPPEYHRKAV

1. Conduct a BLAST search to identify the protein and the species from which it is derived; use default parameters for the search except increase **Max target sequences** to 500. Can you identify the species with certainty in this case; can you do so with certainty in general? Why or why not? Comment on the species distribution of this protein. Say you want to find a potential ortholog in budding yeast (*Saccharomyces cerevisiae*), your favorite organism. What parameters of BLAST would you consider changing? Can you confirm the presence of an ortholog following adjustment of these parameters? If so, what is the name of the protein and why do you suspect it to be an ortholog as opposed to being just a homolog?
2. Examine the data contained in the ‘Search Summary’ at the top of the BLAST Results page to determine the number of ‘letters’ in the database. Assuming your computer is equipped with a [Intel Core i7 4770k](http://en.wikipedia.org/wiki/Million_instructions_per_second) processor, how long would it take to do a rigorous search using the Smith-Waterman algorithm? How long (wall clock time) did the BLAST algorithm take to do the search? Note that the latter is an underestimate while the former is an overestimate because the time it takes for parsing input/output, internet bandwidth, and server traffic considerations are not taken into account. If it takes 1 byte to store the optimal score in each cell of the dynamic programming matrix, how much space would it take to store the results of the full matrix (assume also 1 KB = 1000 bytes; 1 MB = 1000 KB; 1 GB = 1000 MB).

**TASK 3:** *Python I/O* (**5 points**)

Read Chapter 3 of the Python textbook before attempting the task below. Store the nucleotide sequence in **HA2 T5 ii** in a file called dna.txt. Write a python script that reads the contents of this file and outputs the sequences of exon 1, the sole intron, and exon 2 in separate files. The files should be in FASTA format and the header of each file should reflect the contents of the file.