Sequence Alignment

CB2-201 – Computational Biology and Bioinformatics

February 18, 2015

Emidio Capriotti

http://biofold.org/emidio

Biomolecules Folding and Disease

Division of Informatics
Department of Pathology

UAB
THE UNIVERSITY OF
ALABAMA AT BIRMINGHAM

Sequence Analysis

Sequence Analysis is one the first research field in Bioinformatics and Computational biology which consists in processing DNA, RNA or peptide sequence through with wide range of analytical methods to understand characterize the function, structure, or evolution.

The main methodology for sequence analysis consist in the comparison of sequences performed using

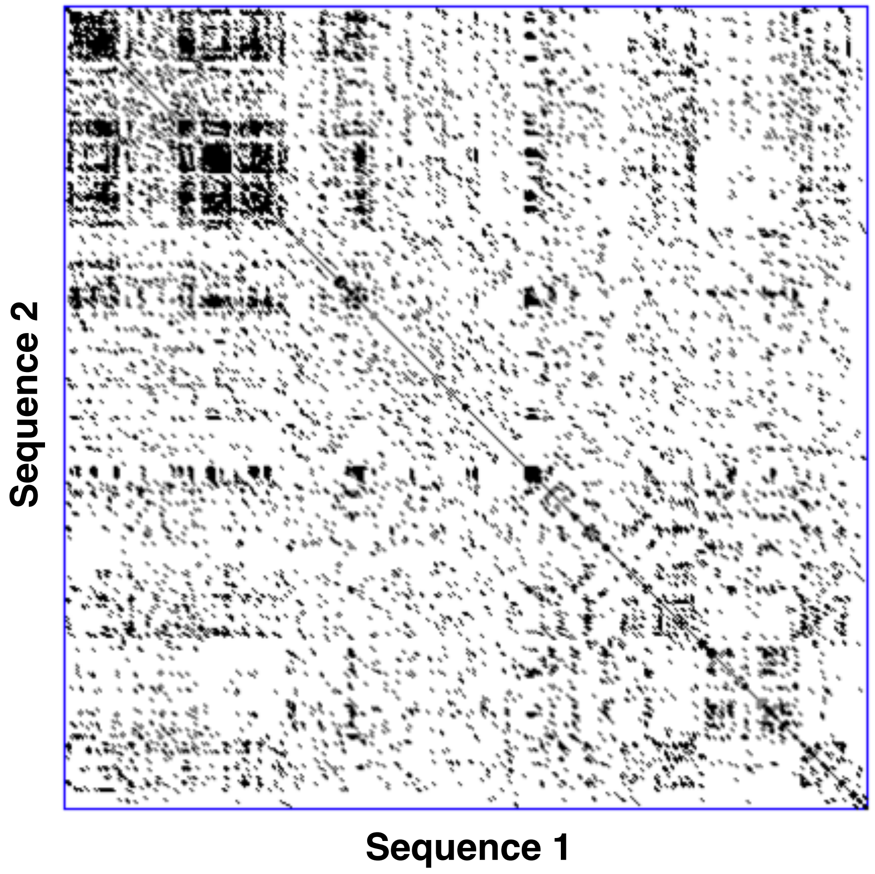
1. Pattern matching
2. Dot Plot
3. Sequence alignment

from wikipedia

Sequence Comparison

Pattern Matching: Search a fraction of the sequence using a regular expression which are used for searching in a text file.

Dot Plot: graphical method that allows the comparison of two biological sequences and identify regions of close similarity between them.



from wikipedia

Sequence Alignment

The sequence alignment is a method of sequence comparison based detection of similarity between 2 or multiple biomolecules.

When sequences are small and similar enough the alignment can be executed manually, in more complex cases an optimization procedure is needed to find the best alignment.

Alignment can be either Pairwise or Multiple and Global or Local

The main issues

How to find the optimal alignment that maximizes the matching between sequences?

How the score is defined. Are gaps allowed?

What is the best algorithm?

How the result is evaluated?

A simple score

A basic alignment score consists in assigning 1 to matching residues or nucleotides otherwise -1

```
AGATCAGAAATG
--AT-AG-AAT-
--::-::-:::-
```

```
AGATCAGAAATG
--AT-AGAA-T-
--::-::::-:-
```

The optimization

Given two sequences of length *m* and *n* a brute force algorithm that calculates all the possible pairwise alignment will have the exponential complexity

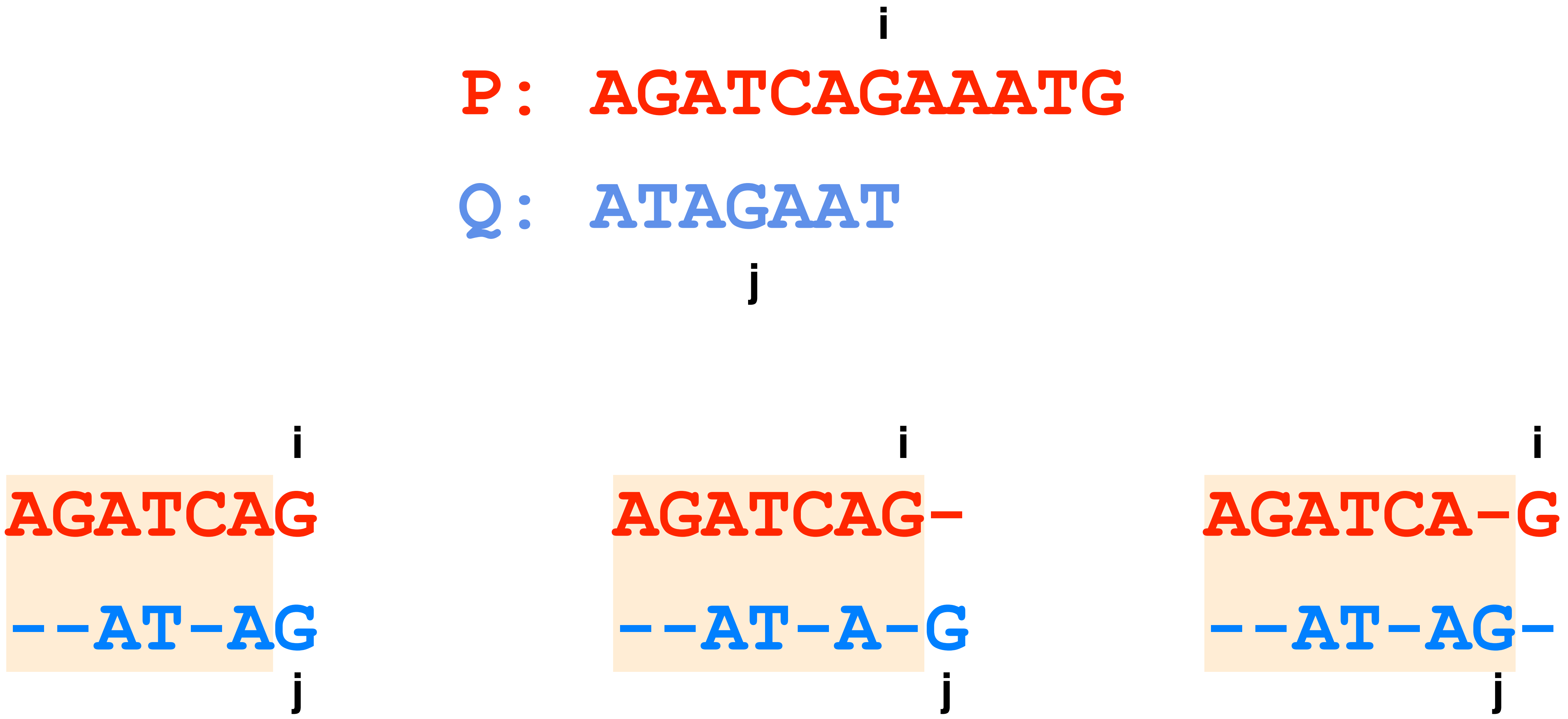
$$\binom{n+m}{m} = \frac{(m+n)!}{n!m!}$$

“*Divide and conquer*” approach allow to reduce the computational complexity of the problem.

The basic assumption: the optimal alignment between two strings should include the optimal alignment of the smaller prefixes.

Divide and Conquer

Given two sequence *P* and *Q*, the knowing the optimal alignment between the prefix $P_{1,i-1}$ and $Q_{1,j-1}$ the best alignment between the positions *i* in *P* and *j* in *Q* is the optimal alignment between positions prefixes *i-1,j-1* or *i-1,j* or *i,j-1*.



```
           i
P:  AGATCAGAAATG

Q:  ATAGAAT
       j
```

```
      i               i              i
AGATCAG        AGATCAG-        AGATCA-G

--AT-AG        --AT-A-G        --AT-AG-
      j               j              j
```

Examples from CAP5937: Introduction to Bioinformatics: http://people.cs.missouri.edu/~chengji/

Global Alignment

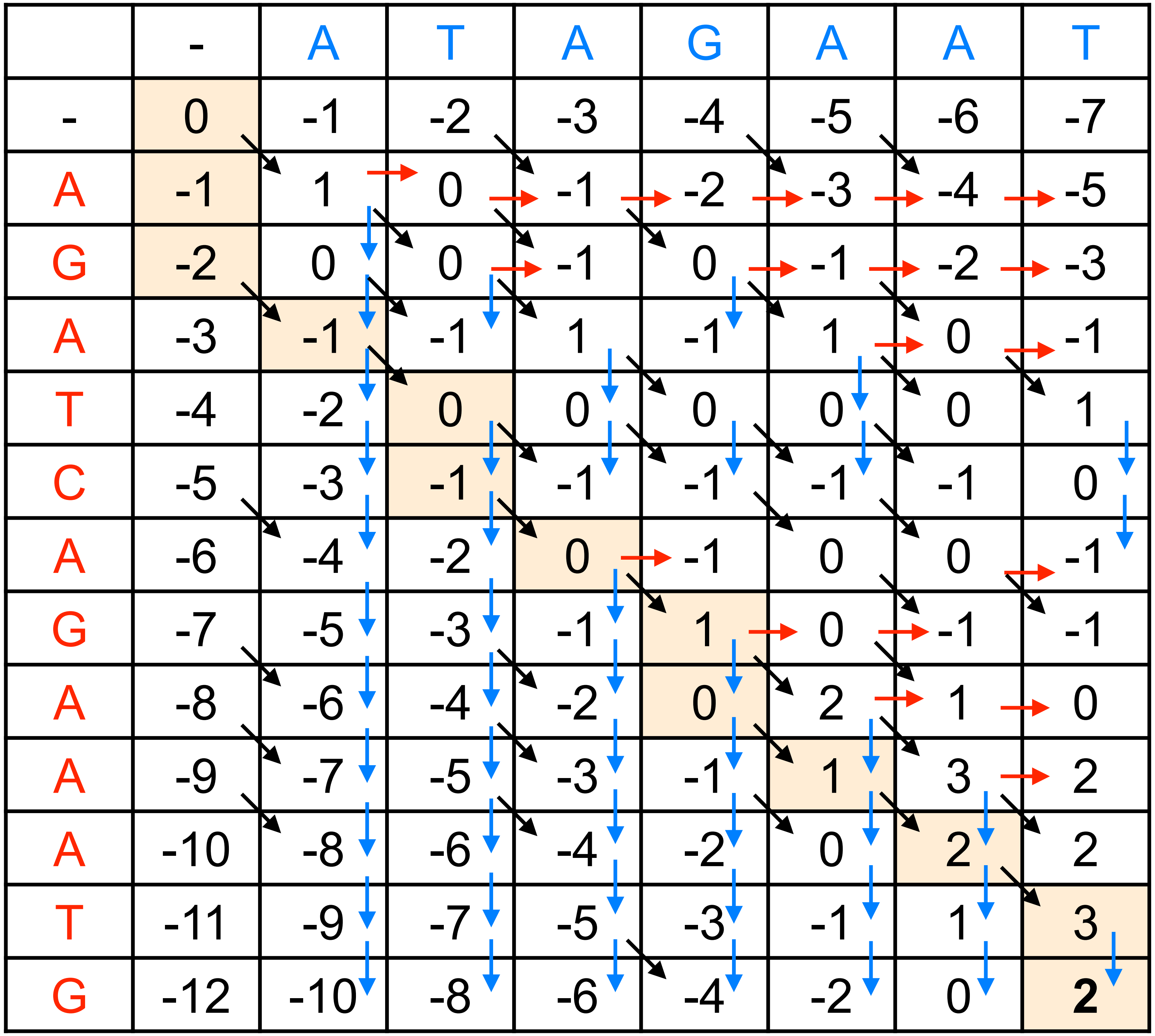
Global alignment performed using the Needleman and Wunsch algorithm and Dynamic Programming approach.

The algorithm is based on the calculation of a matrix M[i,j] and a Trace Back step that defines the alignment.

$$\begin{aligned} M[i,j] = \max [\; & M[i-1,j-1]+S(P[i],Q[j]), \\ & M[i,j-1]+ S(-,Q[j]), \\ & M[i-1,j]+S(P[i],-) \;] \end{aligned}$$

where S and P are the two sequences.

An example



	-	A	T	A	G	A	A	T
-	0	-1	-2	-3	-4	-5	-6	-7
A	-1	1	0	-1	-2	-3	-4	-5
G	-2	0	0	-1	0	-1	-2	-3
A	-3	-1	-1	1	-1	1	0	-1
T	-4	-2	0	0	0	0	0	1
C	-5	-3	-1	-1	-1	-1	-1	0
A	-6	-4	-2	0	-1	0	0	-1
G	-7	-5	-3	-1	1	0	-1	-1
A	-8	-6	-4	-2	0	2	1	0
A	-9	-7	-5	-3	-1	1	3	2
A	-10	-8	-6	-4	-2	0	2	2
T	-11	-9	-7	-5	-3	-1	1	3
G	-12	-10	-8	-6	-4	-2	0	**2**

Examples from CAP5937: Introduction to Bioinformatics: http://people.cs.missouri.edu/~chengji/

Local Alignment

Global alignment performed using the Smith and Waterman algorithm and Dynamic Programming approach.

In these alignment the M[i,j] is calculate using the following equation

$$M[i,j] = \max \left[0,\ M[i-1,j-1]+S(P[i],Q[j]),\ M[i,j-1]+ S(-,Q[j]),\ M[i-1,j]+S(P[i],-) \right]$$

where S and P are the two sequences.

In this case the Trace Back step can start from each position of the matrix and stops when the score is 0.

The same example

	-	A	T	A	G	A	A	T
-	0	0	0	0	0	0	0	0
A	0	1	0	1	0	1	1	0
G	0	0	0	0	2	1	0	0
A	0	1	0	1	0	3	2	1
T	0	0	2	1	0	2	2	3
C	0	0	1	0	0	1	1	2
A	0	1	0	2	1	1	2	1
G	0	0	0	1	3	2	1	1
A	0	1	0	1	2	4	3	2
A	0	1	0	1	1	3	**5**	4
A	0	1	0	1	0	2	4	4
T	0	0	2	1	0	1	3	**5**
G	0	0	1	1	2	1	2	4

Examples from CAP5937: Introduction to Bioinformatics: http://people.cs.missouri.edu/~chengji/

The results

The procedure returns two alignments with same score

```
ATCAGAA
AT-AGAA
```

```
ATCAGAAAT
AT-AGA-AT
```

The scoring matrix

More sophisticated methods to score the mis-matches in protein sequence alignment based on the observation of mutation rates in curated alignments.

Most famous are the PAM (Point accept mutations) and BLOSUM

There are numbers associated to each matrix.
PAM higher the number high the divergence between the sequence included for the calculation of the mutation rates. (PAM250 most divergent).

For BLOSUM is the opposite higher the number higher the similarity

$$sim(i,j) = \log \frac{P(i,j)}{P(i)P(j)}$$

BLAST Algorithm

The first revolution in the bioinformatics was the development of the Basic Local Alignment Search Tool (BLAST).

The most recent paper *Altschul SF et al.* (1997) NAR PMID: 9254694 more than 41,000 citations in SCOPUS.

BLAST allows to compare large dataset of sequences in short amount time with respect to standard alignment programs.

BLAST implements a heuristic method, to finds similar sequences by locating short matches between the two sequences.

This process of finding initial words is called seeding. The alignment of words by default of 3 letters is used to calculate a local alignment

How to run BLAST

First step consist in creating a database in appropriate format using
formatdb -i database

One example

blastpgp -i fastafile -d database -o output.txt

```
BLASTP 2.2.18 [Mar-02-2008]


Query= sp|P04637|P53_HUMAN Cellular tumor antigen p53 OS=Homo sapiens
GN=TP53 PE=1 SV=4
         (393 letters)

Database: uniprot_sprot.fasta
           547,599 sequences; 195,014,757 total letters

Searching..................................................done
                                                                Score    E
Sequences producing significant alignments:                      (bits) Value

sp|P04637|P53_HUMAN Cellular tumor antigen p53 OS=Homo sapiens G...   813   0.0
sp|P13481|P53_CHLAE Cellular tumor antigen p53 OS=Chlorocebus ae...   746   0.0
```

How to run BLAST

Use biopython to parse blastoutput

```
>>> from Bio.Blast import NCBIXML
>>> result_handle = open(“blast_output.xml")
>>> blast_record = NCBIXML.read(result_handle)
>>> for alignment in blast_record.alignments:
...     for hsp in alignment.hsps:
...         if hsp.expect < E_VALUE_THRESH:
...             print('****Alignment****')
...             print('sequence:', alignment.title)
...             print('length:', alignment.length)
...             print('e value:', hsp.expect)
...             print(hsp.query[0:75] + '...')
...             print(hsp.match[0:75] + '...')
...             print(hsp.sbjct[0:75] + '...')
```

Parse Alignment

Use muscle (http://www.drive5.com/muscle/downloads.htm)
to perform pairwise alignment and parse the output with biopython

```
>>> from Bio import AlignIO
>>> align = AlignIO.read(“output.fasta", “fasta")
>>> seqs= align.get_all_seqs()
>>> s1=seqs[0].seq.tostring()
>>> s2=seqs[1].seq.tostring()
>>> alen=align.get_alignment_length()
```

Calculate the sequence identity between the two sequences.

Problem

1. BLAST P53 against SwissProt. Rank non human proteins according to their e-value.

2. Select sequences with e-value lower than 1.0e-3

3. Calculate the pairwise alignment between the P53_HUMAN and the select sequences and calculate the sequence identity.