Read alignment

Introduction to alignments, scoring matrices, alignment practice

Required software

Install EMBOSS bioinformatics suite:

```
$ conda install -c bioconda emboss
```

Install BLAST

```
$ conda install -c bioconda blast
```

Install entrez-direct

```
$ conda install -c bioconda entrez-direct
```

Install seqtk

```
$ conda install -c bioconda seqtk
```

Introduction to alignments

- Sequence alignment is fundamental bioinformatics task
- Practically, any bioinformatics works involves DNA, RNA, or protein sequence alignment
- Alignment means arranging two or more sequences in a way that the regions of similarity line up

Example of an alignment

```
GATTACA
||| |
GATCA--
```

This is not the only possible arrangement

```
GATTACA
|||  ||
GAT--CA
```

Examples from Bioinformatics Handbook, 2022

Introduction to alignments

- Humans mind is notoriously bad in evaluating alignments!
- We tend to get fixated on certain pattern while ignoring many other possible alternatives
- The problem of sequence alignment is best suited for computational solution

Let's examine this 3 alternative alignments:

1	2	3
GATTACA ||| | GATCA- -	GATTACA ||| || GAT- - CA	GATTACA ||| | | GAT -C -A

Which of them is better? Which of them we should rely on when answering biological questions.

Introduction to alignments

Why do we use alignments?

- Finding similar regions between sequences (similarity may indicate evolutionary relatedness)
- Finding which sequences from many alternatives in the most similar query (input) sequence

What determines a sequence alignment

- Type of the alignment algorithm: 1) global; 2. local; 3. semi-global
- Scoring matrix: numerical values assigned to matches, mismatches, and gaps
- Different algorithms may produce different alignments for the same sequences
- Different scoring matrices may generate different alignments with the same sequences even with the same algorithm

Algorithm * Scoring matrix → Alignment

Introduction to alignments

Scoring alignments

1	2	3
GATTACA ||| | GATCA- -	GATTACA ||| || GAT- - CA	GATTACA ||| | | GAT -C -A

We have 3 alternative alignments; how do we determine the best one?

We can devise a way to assign numeric scores that will reward an alignments for match and punish it for mismatches or gaps

Mismatch → substitution of one nucleotide or amino-acid by another, for example A/C

Gap → a nucleotide or an amino-acid is missing from a query sequence caused by insertion or deletion of a part of a sequence *(indels)*

Introduction to alignments

Scoring alignments

```
1            2            3
GATTACA      GATTACA      GATTACA
|||  |        |||   ||     |||  | |
GATCA- -     GAT- - CA    GAT -C -A
```

Example scoring matrix

Value	**Alignment**
1	Match
-1	Mismatch
-2	Opening a gap
-1	Extending a gap

Let’s score GATTACA alignments in a table above:

M → Match; X → Mismatch; D → gap open; E → gap extension

1. M + M + M +X + M + D + E = 3 – 1 + 1 - 2 – 1 = **0**
2. Score **2** – This is the best alignment according to our scheme
3. Score **1**

Introduction to alignments

What is the best alignment?

- There is no universally best alignment
- The aligner generates a set of alternative alignments and calculates scores based on a scoring matrix
- The alignment with a maximum score is selected as a "best", therefore the objective of the aligner is a score maximization

Value	Alignment
1	Match
-1	Mismatch
-2	Opening a gap
-1	Extending a gap
0	**Opening or extending a gap at the end of either sequence**

- ✓ The choice of the scoring matrix is critical to answering biologically meaningful questions
- ✓ For example, we would like to detect the longest marching run in the shorter sequence
- ✓ To achieve this, we will remove penalty for opening or extending a gap at either end of the sequence
- ✓ Alignment 1 scores **4**; 2 → **2**; 3 → **1**

Introduction to alignments

What are scoring matrices

- Scoring matrices can apply to nucleotides, amino-acids, and codons
- Scoring matrices reflect the likelihoods of substitution rates along the evolutionary history
- Scoring matrices are calculated based on the alignment of many homologous sequences
- Two large families of scoring matrixes: BLOSUM and PAM

BLOSUM	PAM
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC50453/	https://doi.org/10.1093/molbev/msi005
BLOcks **SU**bstitution **M**atrix	Point Accepted Mutation
Based on very conserved regions in protein families with no gaps in alignments	Replacement of a single AA accepted by natural selection. Silent and lethal mutations are ignored
Based on relative frequencies of amino-acids (AA) and their substitution probabilities	Entry in a PAM matrix indicate the likelihood of the AA of that row being replaced with the AA of that column through a series PAM during a specified evolutionary
log-odds scores for each of the 210 possible substitution pairs of the 20 standard amino acids.	

Introduction to alignments

EDNAFULL nucleotide scoring matrix

Where to find scoring matrices:
ftp://ftp.ncbi.nlm.nih.gov/blast/matrices

Download and view EDNAFULL matrix:
```
$ curl -O ftp://ftp.ncbi.nlm.nih.gov/blast/matrices/NUC.4.4
$ cat NUC.4.4
```

	A	T	G	C
A	5	-4	-4	-4
T	-4	5	-4	-4
G	-4	-4	5	-4
C	-4	-4	-4	5

The actual matrix contains also contains ambiguous bases

Introduction to alignments

EDNAFULL nucleotide scoring matrix

M : A – A → 5

X : A – T → -4

	A	T	G	C
A	5	-4	-4	-4
T	-4	5	-4	-4
G	-4	-4	5	-4
C	-4	-4	-4	5

How to select a scoring matrix:

https://pubmed.ncbi.nlm.nih.gov/24509512/

Selecting the Right Similarity-Scoring Matrix

William R Pearson

This matrix tends to produce mismatches, for example:

1. Alignment of ACT to AGT will produce:

```
A C T
|   |
A G T
```

If we change penalties of a match to 4 and mismatch to -5 this alignment with look as follows

```
A
|
A
```

The alignment will not extend to the left, since the penalty of mismatch outweighs the reward of a match

Introduction to alignments

Other properties of the scoring matrixes

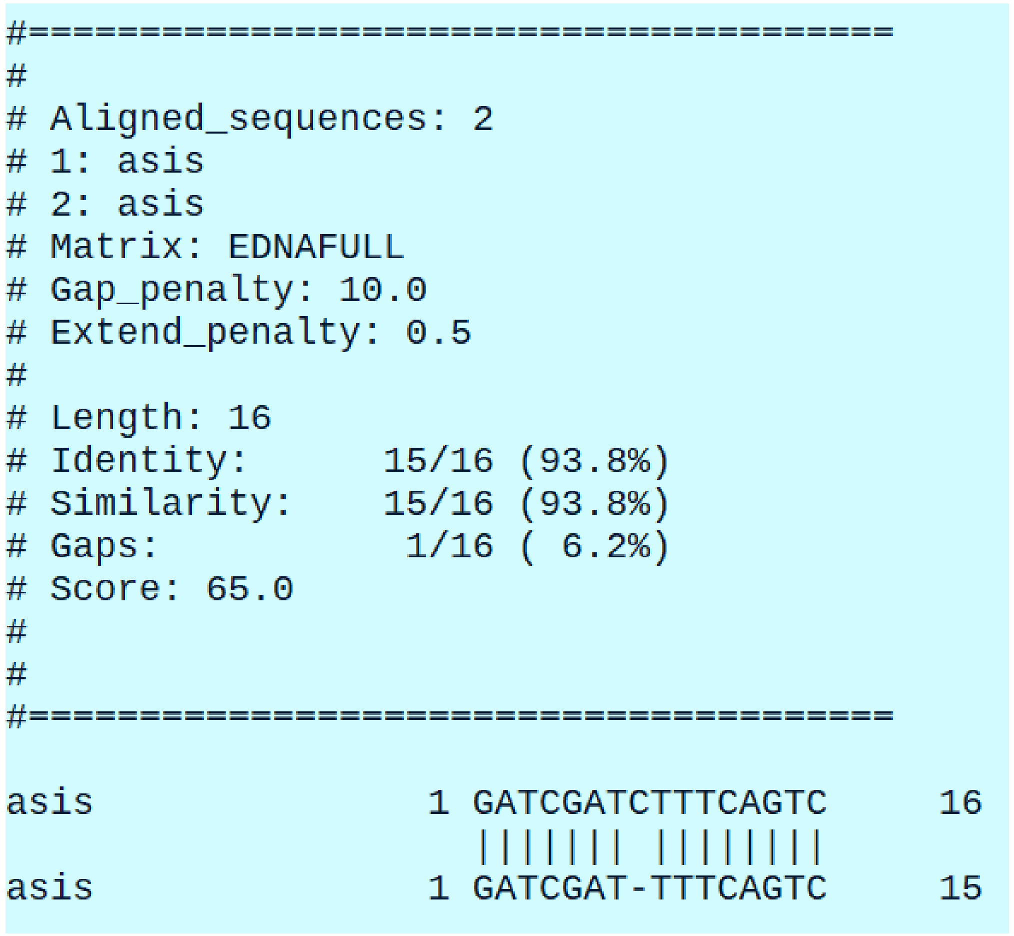
- Scoring matrixes do not contain gap opening and gap extension penalties
- Gap opening and extension penalties are typically different; gap extension penalty is typically smaller than gap opening. Additional reading: https://en.wikipedia.org/wiki/Gap_penalty
- Choosing right scoring is critical as gap penalties decide the “willingness” of the aligner to open gaps
- The scores are typically on the logarithmic scale
- The choice of scoring matters less when the sequences are very similar
- The more different are sequences, the more sensitive the alignment is to scoring choices
- The longer the sequence

Practice alignments

We will use **needle** from EMBOSS suite to practice alignments.

needle uses Needleman-Wunsch global alignment algorithm

Let’s try an alignment
$ needle -asequence asis:GATCGATCTTTCAGTC -bsequence asis:GATCGATTTTCAGTC -auto -stdout # **-asequence** and **-bsequence** specify query and reference sequences; **asis** allows to enter a sequence on the command line; **-auto** – use default gap-open [10] and gap-extend [0.5] penalty; **-stdout** prints output to screen



```
#=======================================
#
# Aligned_sequences: 2
# 1: asis
# 2: asis
# Matrix: EDNAFULL
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 16
# Identity:      15/16 (93.8%)
# Similarity:    15/16 (93.8%)
# Gaps:           1/16 ( 6.2%)
# Score: 65.0
#
#
#=======================================

asis                  1 GATCGATCTTTCAGTC     16
                        ||||||| ||||||||
asis                  1 GATCGAT-TTTCAGTC     15
```

Practice alignments

We will use **needle** from EMBOSS suite to practice alignments.

There are different ways to visualize the alignments, but the following symbols are commonly used:

- **| - match**
- **- - gap**
- **. - mismatch**

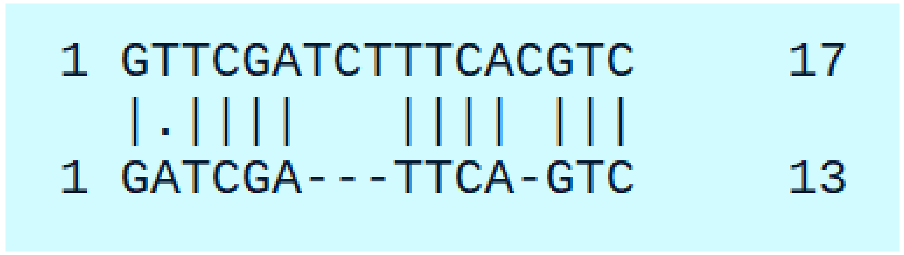
```
1 GTTCGATCTTTCACGTC     17
  |.||||   |||| |||
1 GATCGA---TTCA-GTC     13
```

- ✓ In the alignment above we have 12 matches, 2 gaps, and 1 mismatch
- ✓ Usually, a **query** sequence is on the bottom and **subject** sequence is on the top
- ✓ **Query –** sequence we need to compare; **Subject -** sequence we are comparing the query to

Practice alignments

We will use **needle** from EMBOSS suite to practice alignments.

How to describe alignments?

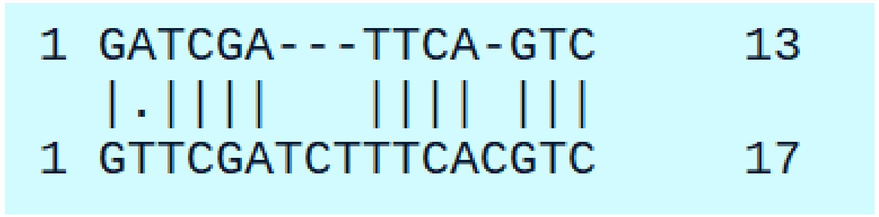


```
1 GTTCGATCTTTCACGTC     17
  |.||||   |||| |||
1 GATCGA---TTCA-GTC     13
```

In this case we will say that the alignment has 2 deletions (part of the sequence is missing) and 1 mismatch

Let’s flip the sequences and make query sequence a subject

```
$ needle -asequence asis:GATCGATTCAGTC -bsequence asis:GTTCGATCTTTCACGTC -auto -stdout
```



```
1 GATCGA---TTCA-GTC     13
  |.||||   |||| |||
1 GTTCGATCTTTCACGTC     17
```

Now we will describe the alignment as having 2 insertions and 1 mismatch

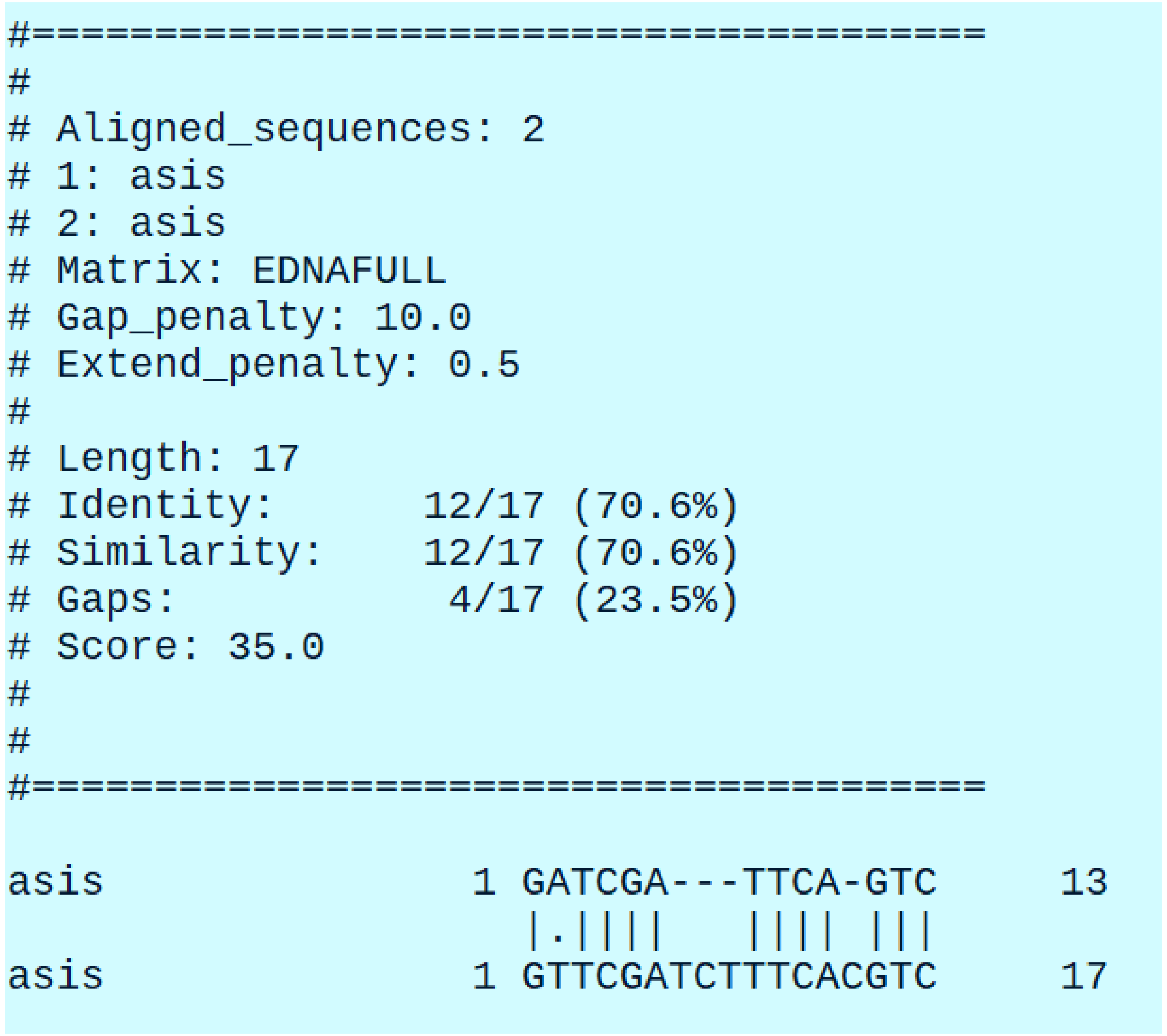
Practice alignments

Other characteristics of the alignment (apart of the score)

Let’s run this command again

$ needle -asequence asis:GATCGATTCAGTC -bsequence asis:GTTCGATCTTTCACGTC -auto -stdout

- ✓ Score – 35
- ✓ Percent identity – 70.6% # what percent of the sequence is the same
- ✓ Percent similarity – 70.6% # what percent of the sequence with similar bases/amino-acids



```
#=======================================
#
# Aligned_sequences: 2
# 1: asis
# 2: asis
# Matrix: EDNAFULL
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 17
# Identity:      12/17 (70.6%)
# Similarity:    12/17 (70.6%)
# Gaps:           4/17 (23.5%)
# Score: 35.0
#
#
#=======================================

asis               1 GATCGA---TTCA-GTC     13
                     |.||||   |||| |||
asis               1 GTTCGATCTTTCACGTC     17
```

Practice alignments

Confusion of terminology

Terms in bioinformatics are frequently poorly defined or calculated differently depending on the tool

[On the definition of sequence identity (lh3.github.io)]

Gap excluded identity: pident = matches / (matches + mismatches)

BLAST identity: pident = matches / (matches + mismatches + deletions)

Gap compressed identity: pident = matches / (matches + mismatches + gapopen)

Practice alignments

Compact record of the alignment: **CIGAR** string

```
1 GTTCGATCTTTCACGTC    17
  |.||||   |||| |||
1 GATCGA---TTCA-GTC    13
```

How can we record this alignment?

- 1 match – 1M
- 1 mismatch – 1X
- 4 matches – 4M
- 3 deletions – 3D
- 4 matches – 4M
- 1 deletion – 1D
- 3 matches – 3M

Put it all together:

1M1X4M3D4M1D3M This format type is called **Extended CIGAR** format

Sequence Alignment Map (SAM) files use different CIGAR format, where both matches in mismatches are shown as **M**

6M3D4M1D3M

Yet another CIGAR format goes one step further and skips a preceding digit for single-base changes

6M3D4MD3M

Global and local alignments

Three types of alignment algorithms:

- Global – Needleman-Wunsch
- Local – Smith-Waterman
- Semi-global

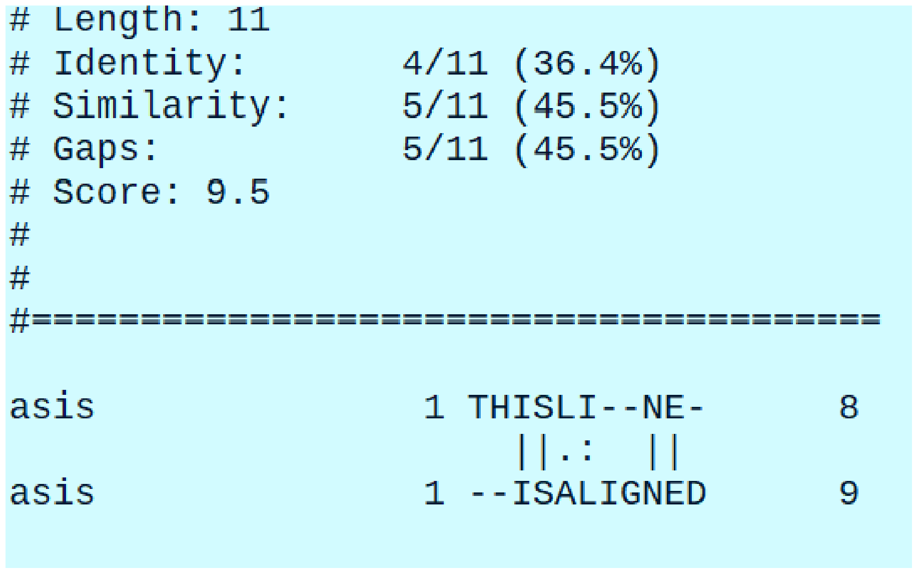
Global and local alignments

Global alignment

Global alignment is designed to find and alignment of sequences over their full length while allowing gaps

Let’s try the following example from the book Understanding Bioinformatics by Marketa Zvelebil and Jeremy Baum:

```
$ needle -asequence asis:THISLINE -bsequence asis:ISALIGNED -auto -stdout
```



```
# Length: 11
# Identity:       4/11 (36.4%)
# Similarity:     5/11 (45.5%)
# Gaps:           5/11 (45.5%)
# Score: 9.5
#
#
#========================================

asis               1 THISLI--NE-       8
                       ||.:  ||
asis               1 --ISALIGNED       9
```

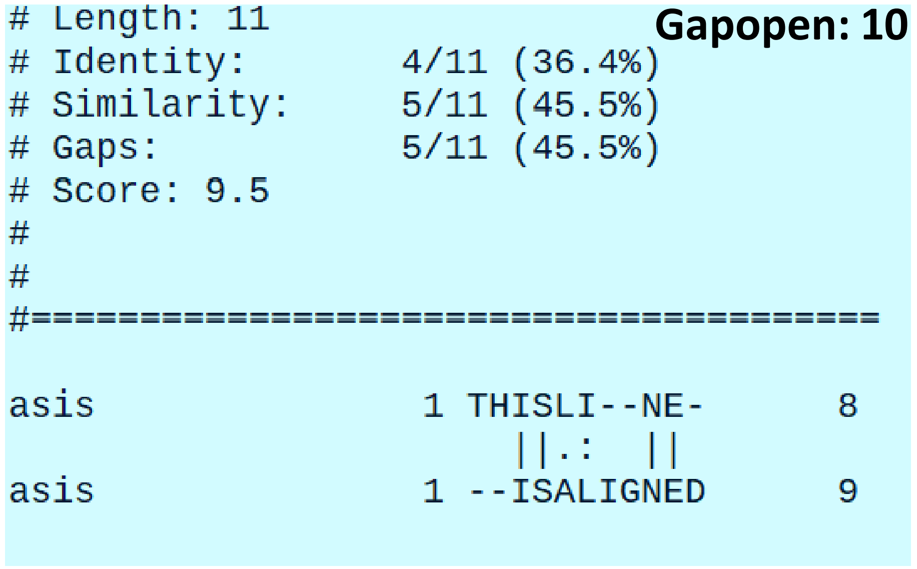
Global and local alignments

Global alignment

By default, ***needle*** has a gapopen penalty of 10.

Let’s decrease gapopen penalty and observe the results:

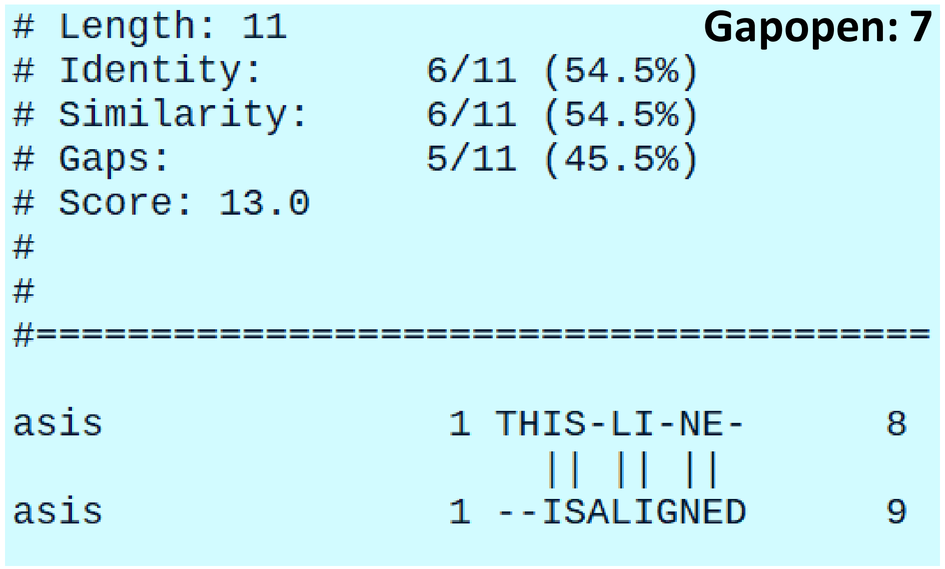
```
$ needle -asequence asis:THISLINE -bsequence asis:ISALIGNED -auto -stdout -gapopen 7
```



Gapopen: 10

```
# Length: 11
# Identity:       4/11 (36.4%)
# Similarity:     5/11 (45.5%)
# Gaps:           5/11 (45.5%)
# Score: 9.5
#
#
#=======================================

asis               1 THISLI--NE-      8
                       ||.:  ||
asis               1 --ISALIGNED      9
```



Gapopen: 7

```
# Length: 11
# Identity:       6/11 (54.5%)
# Similarity:     6/11 (54.5%)
# Gaps:           5/11 (45.5%)
# Score: 13.0
#
#
#=======================================

asis               1 THIS-LI-NE-      8
                       || || ||
asis               1 --ISALIGNED      9
```

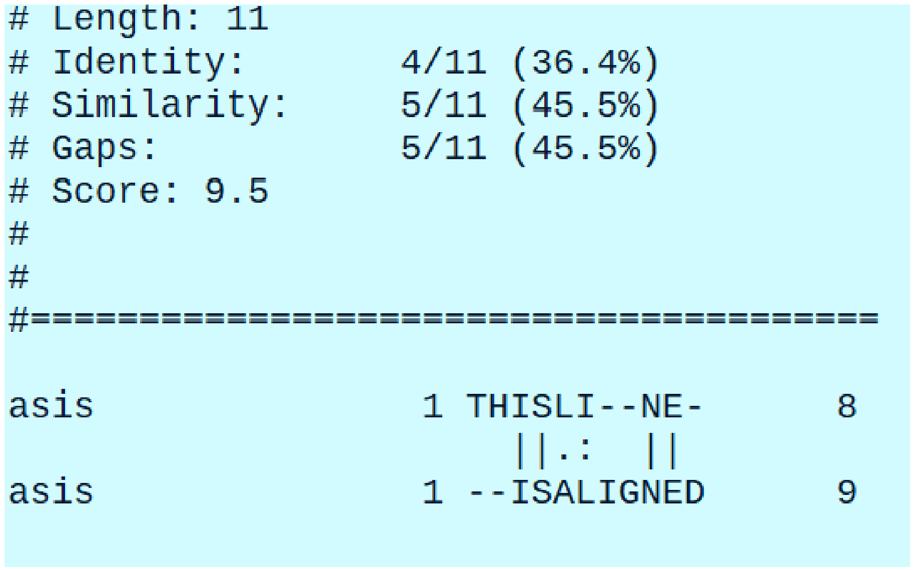
We created an alignment that favors gaps over the mismatches!

Global and local alignments

Global alignment

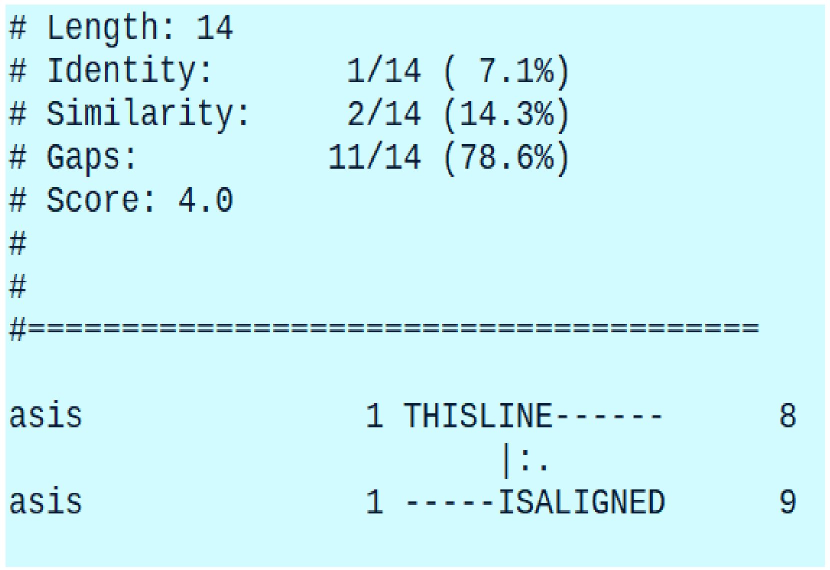
By default, ***needle*** has a gapopen penalty of 10.

Let’s increase gapopen penalty to 20 and observe the results:
$ needle -asequence asis:THISLINE -bsequence asis:ISALIGNED -auto -stdout -gapopen 20



```
# Length: 11
# Identity:       4/11 (36.4%)
# Similarity:     5/11 (45.5%)
# Gaps:           5/11 (45.5%)
# Score: 9.5
#
#
#=======================================

asis               1 THISLI--NE-      8
                       ||.:  ||
asis               1 --ISALIGNED      9
```



```
# Length: 14
# Identity:       1/14 ( 7.1%)
# Similarity:     2/14 (14.3%)
# Gaps:          11/14 (78.6%)
# Score: 4.0
#
#
#=======================================

asis               1 THISLINE------      8
                          |:.
asis               1 -----ISALIGNED      9
```

High penalty for opening the gaps prevented any gaps in the alignment, the highest score could be achieved with mismatches only.

Global and local alignments

Local alignment

- Local alignment is designed to find the region of highest similarity between the sequences

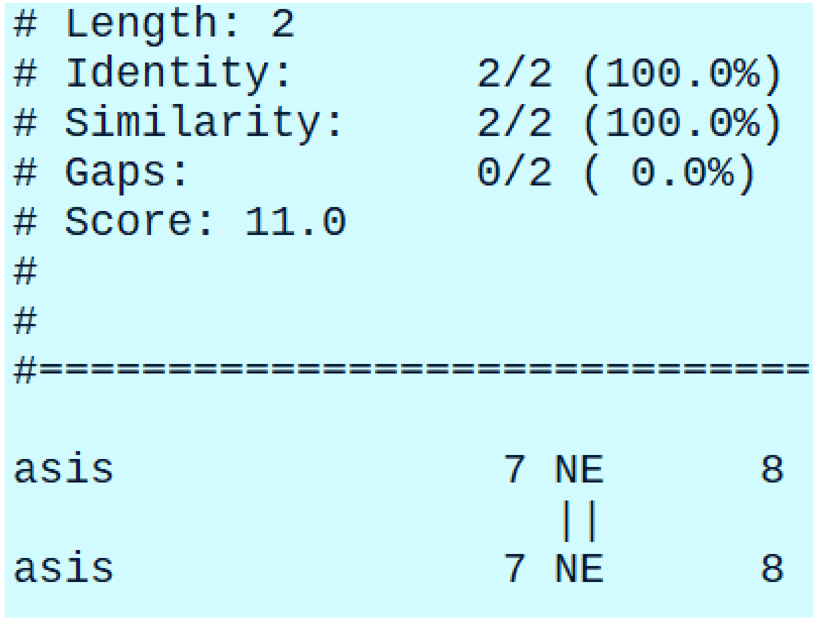
- In other words, we are looking for the partial interval of the query sequence that produces the highest scoring alignment with the subject

- Local alignment is implemented in ***Smith-Waterman*** algorithm

- EMBOSS utility for local alignment is called ***water***

Let's try local alignment with the same sample sequences as in global alignment

$ water -asequence asis:THISLINE -bsequence asis:ISALIGNED -auto -stdout



```
# Length: 2
# Identity:        2/2 (100.0%)
# Similarity:      2/2 (100.0%)
# Gaps:            0/2 ( 0.0%)
# Score: 11.0
#
#
#================================

asis                  7 NE       8
                        ||
asis                  7 NE       8
```

Global and local alignments

Local alignment

- Local alignment generated here is very short
- We tried all possible alternatives and NE=NE matches gave us the maximum score
- By default, water uses BLOSUM62 protein matrix.
- Let’s change the scoring matrix and see the effect on the results

```
# Length: 2
# Identity:       2/2 (100.0%)
# Similarity:     2/2 (100.0%)
# Gaps:           0/2 ( 0.0%)
# Score: 11.0
#
#
#================================

asis                 7 NE      8
                       ||
asis                 7 NE      8
```

Download BLOSUM90 from NCBI site:

```
$ wget -nc ftp://ftp.ncbi.nlm.nih.gov/blast/matrices/BLOSUM90
$ cat BLOSUM90
```

Global and local alignments

Local alignment

Map the same 2 sequences with BLOSUM90 matrix

$ water -asequence asis:THISLINE -bsequence asis:ISALIGNED -auto -stdout -datafile BLOSUM90

```
# Aligned_sequences: 2
# 1: asis
# 2: asis
# Matrix: BLOSUM90
# Gap_penalty: 10.0
# Extend_penalty: 0.5
#
# Length: 6
# Identity:       4/6 (66.7%)
# Similarity:     5/6 (83.3%)
# Gaps:           1/6 (16.7%)
# Score: 14.0
#
#
#====================================

asis                  4 SLI-NE      8
                        :|| ||
asis                  3 ALIGNE      8
```

Global and local alignments

Selecting the right similarity matrix

Paper by FASTA author Pearson: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3848038/
A paper about evolutionary distances: https://pubmed.ncbi.nlm.nih.gov/11752185/

- Scoring (similarity) matrixes reflect evolutionary relationships

- Different similarity matrixes are effective at different evolutionary distances

- “Deep” scoring matrixes (BLOSUM50, BLOSUM62) allow more AA substitutions and gaps

- “Shallow” matrixes (VT10, VT20, VT40) give higher scores to matches, more negative scores to mismatches and higher gap penalties

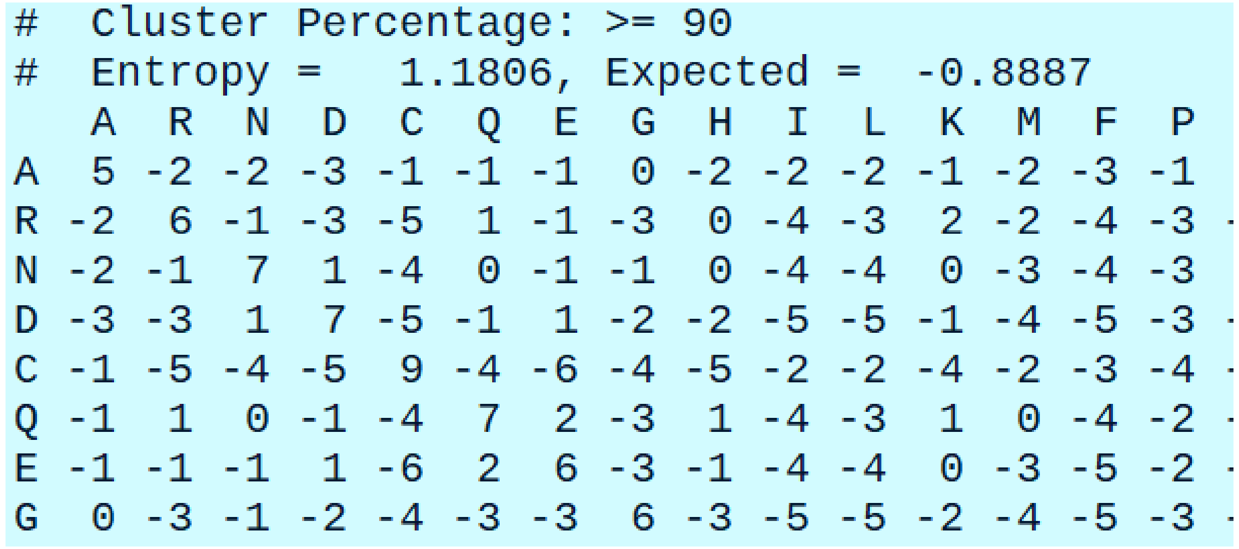
- “Deep” matrixes should be used to target alignments with 20-30% identities

- “Shallow” matrixes are ideal for finding alignments in the sequences with 50-90% similarities, protein domains, exons, DNA reads, closely related orthologs

Global and local alignments

Similarity scores and probabilities

Take a look at a scoring matrix:
$ cat BLOSUM90



```
#  Cluster Percentage: >= 90
#  Entropy =   1.1806, Expected =  -0.8887
   A  R  N  D  C  Q  E  G  H  I  L  K  M  F  P
A  5 -2 -2 -3 -1 -1 -1  0 -2 -2 -2 -1 -2 -3 -1
R -2  6 -1 -3 -5  1 -1 -3  0 -4 -3  2 -2 -4 -3 -
N -2 -1  7  1 -4  0 -1 -1  0 -4 -4  0 -3 -4 -3
D -3 -3  1  7 -5 -1  1 -2 -2 -5 -5 -1 -4 -5 -3 -
C -1 -5 -4 -5  9 -4 -6 -4 -5 -2 -2 -4 -2 -3 -4 -
Q -1  1  0 -1 -4  7  2 -3  1 -4 -3  1  0 -4 -2 -
E -1 -1 -1  1 -6  2  6 -3 -1 -4 -4  0 -3 -5 -2 -
G  0 -3 -1 -2 -4 -3 -3  6 -3 -5 -5 -2 -4 -5 -3 -
```

- Note that all scores are integers
- The scores reflect probabilities represented log 2 odds
- The substitution score of -5 means 2^(-5) = 1/32, and the score of 3 means 2^(-3) = 1/8
- The substitution with the score of -5 is 4 times (32/8 = 4) less likely than that subst. with a score of -3
- AA pairs with lower negative scores have less divergent properties that those with higher negative scores
- For simplicity log 2 odds were rounded to the nearest integers

Global and local alignments

Semi-global alignment

- Semi-global alignments combine the properties of global and local alignments
- The objective of semi-global alignment is to find a maximum scoring full length alignment between a shorter **query** and longer **subject** sequence
- This type of alignment is achieved by setting the **end gap** penalty (gaps at the end and the beginning) to zero
- Most of the alignments in NGS data analysis are semi-global
- Semi-global algorithm – modification of Smith-Waterman
- Question: does a shorter sequence originate from a longer one?

Global and local alignments

Misleading alignment

There are limitations in using mathematical concepts to biological phenomena

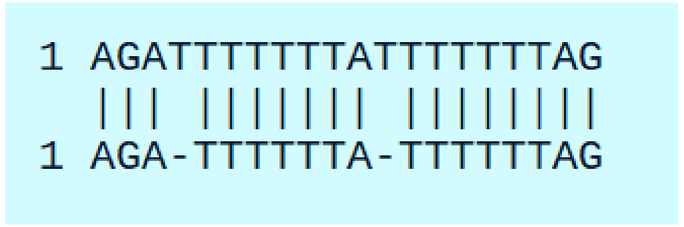
Let's create a sequence with homo-polymer stretches

→ Subject: AGATTTTTTTATTTTTTTAG

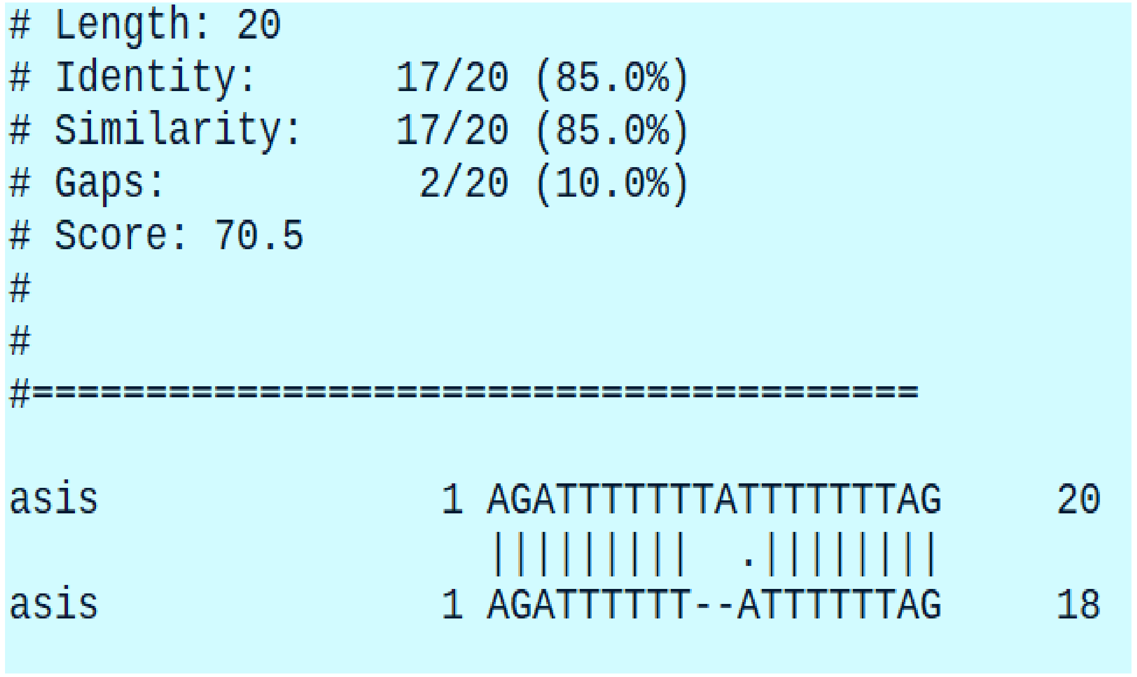
Remove nucleotides marked in red
→ Query: AGATTTTTTATTTTTTAG

Try matching these sequences
$ needle -asequence asis:AGATTTTTTTATTTTTTTAG -bsequence asis:AGATTTTTTATTTTTTAG -auto -stdout

Expected alignment



```
1 AGATTTTTTTATTTTTTTAG
  ||| ||||||| ||||||||
1 AGA-TTTTTTA-TTTTTTAG
```



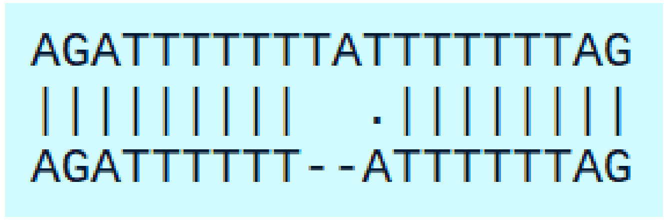
```
# Length: 20
# Identity:      17/20 (85.0%)
# Similarity:    17/20 (85.0%)
# Gaps:           2/20 (10.0%)
# Score: 70.5
#
#
#=======================================


asis                 1 AGATTTTTTTATTTTTTTAG     20
                         |||||||||  .||||||||
asis                 1 AGATTTTTT--ATTTTTTAG     18
```

Global and local alignments

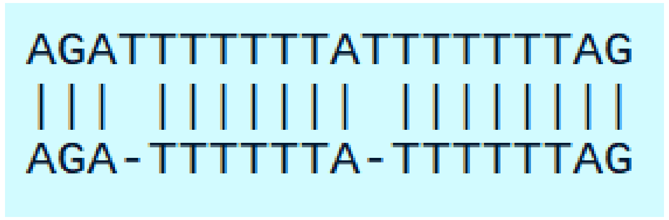
Misleading alignment

1. False, higher score



```
AGATTTTTTTATTTTTTTAG
|||||||||  .||||||||
AGATTTTTT--ATTTTTTAG
```

2. True, lower score



```
AGATTTTTTTATTTTTTTAG
||| ||||||| ||||||||
AGA-TTTTTTA-TTTTTTAG
```

Alignment 1 is mathematically correct, although it does not reflect biological reality

Assuming scores of 1 for match, -1 for mismatch, -10 – gap open and -0.5 gap extend

For alignment 1 (9M2D1X8M) : 17 – 10 – 0.5 – 1 = 5.5

For alignment 2 (3M1D7M1D8M): 18 – 10 - 10 = -2

The algorithm is thrown off by TTTTTTT homopolymer – a region with low information content

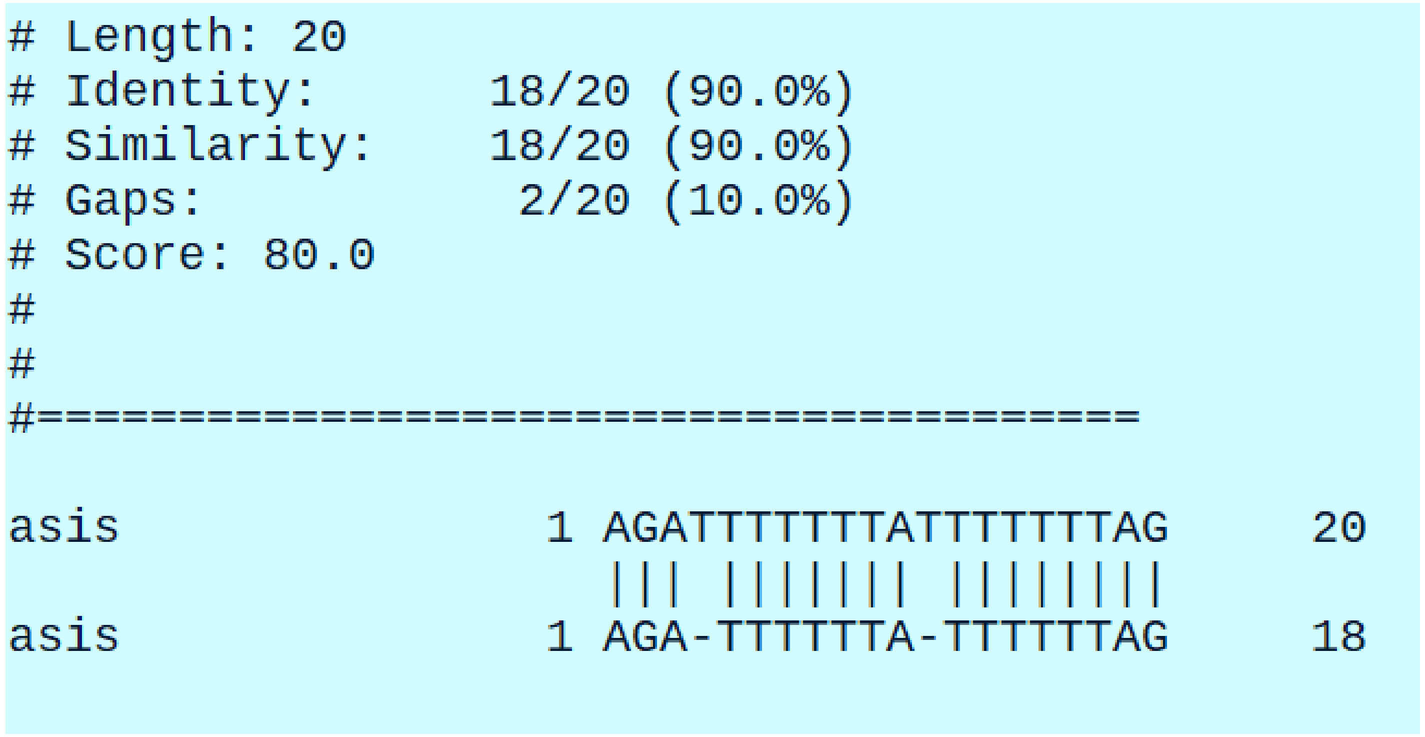
When shifted by one base we are still getting the same base lined up that receives a reward by a matching base

Global and local alignments

Misleading alignment

We can “fix” this alignment by deacreasing gap open penalty

```
$ needle -asequence asis:AGATTTTTTTATTTTTTTAG -bsequence asis:AGATTTTTTTATTTTTTTAG -auto -stdout -gapopen 5
```



```
# Length: 20
# Identity:      18/20 (90.0%)
# Similarity:    18/20 (90.0%)
# Gaps:           2/20 (10.0%)
# Score: 80.0
#
#
#=======================================

asis               1 AGATTTTTTTATTTTTTTAG     20
                       ||| ||||||| ||||||||
asis               1 AGA-TTTTTTA-TTTTTTAG     18
```

Global and local alignments

Misleading alignment

- If decreasing the gap penalty “fixed” the alignment, should we use this setting all the time?
- Lowering the gap penalty will lead to profound effect, the algorithm will be opening gaps at will
- In some cases, lowering the gapopen penalty, will improve the alignments, but normally it will produce erroneous results
- This remains a problem in variant calling, some aligners have procedures that help to recognize and correct “misalignments”
- In generally, alignments in the repetitive (low information) regions produce less reliable variant calls

Global and local alignments

Using BLAST

- **B**asic **L**ocal **A**lignment **S**earch **T**ool – algorithm and a suite of tools
- Primary goal – search a large collection of sequences to find similarities with a query sequence
- BLAST performs local alignments, and the results are mostly partial matches to the query sequence
- BLAST has web interface (https://blast.ncbi.nlm.nih.gov/Blast.cgi) and a command line tool
- Further information: BLAST handbook: https://www.ncbi.nlm.nih.gov/books/NBK279690/

Global and local alignments

Using BLAST

- BLAST is not an optimal aligner, it may not find all the hits, and there are limits how short or how long the sequence is
- A search may occur in nucleotide, protein or translated space
- BLAST is designed to search a huge database of sequences for “hits”
- The more sequences we search the higher is the probability to get a “hit” purely by chance, it is important to know where to draw the line

Let’s try pair-wise BLAST alignment Swine hepatitis gene (AF082843.1) and Orthoherpevirus A (AP003430.1)

```
$ esearch -db nucleotide -query "AF082843.1" | efetch -format fasta > swine_hepE.fasta
$ esearch -db nucleotide -query "AP003430.1" | efetch -format fasta > OrthohepevirusA.fasta
$ blastn -query swine_hepE.fasta -subject OrthohepevirusA.fasta
```

Global and local alignments

Using BLAST

```
Query= AF082843.1 Swine hepatitis E virus genotype 3a strain Meng
nonstructural polyprotein and putative capsid protein genes,
complete cds; and unknown gene

Length=7207
                                                                     Score     E
Sequences producing significant alignments:                         (Bits)  Value

AP003430.1 Orthohepevirus A genomic RNA, complete genome, isolate...  8218    0.0


> AP003430.1 Orthohepevirus A genomic RNA, complete genome, isolate:
JRA1
Length=7230

 Score = 8218 bits (4450),  Expect = 0.0
 Identities = 6291/7207 (87%), Gaps = 17/7207 (0%)
 Strand=Plus/Plus
```

We can learn quite a bit from the report:

- Length of the query sequence – 7207
- Bit-score (8218) – describes overall quality of the alignment, higher score indicates better alignment
- E-value – measures a probability to observe this alignment purely by chance
- Percent identity report a fraction of matching bases between the query and a subject sequences

Global and local alignments

Using BLAST

We can reformat the default output from the command line

```
$ blastn -query swine_hepE.fasta -subject OrthohepevirusA.fasta -outfmt '6 pident’ # this will produce only percent identities
```

Tabular format

```
$ blastn -query swine_hepE.fasta -subject OrthohepevirusA.fasta -outfmt 6
```

Tabular with comment lines

```
$ blastn -query swine_hepE.fasta -subject OrthohepevirusA.fasta -outfmt 7
```

Pairwise

```
$ blastn -query swine_hepE.fasta -subject OrthohepevirusA.fasta -outfmt 0
```

This info can be found out from blast help file

```
$ blastn -help # Check the formatting section
```

Global and local alignments

Using BLAST

Steps to run BLAST on the command line:

- o Download a collection of subject sequences and create a BLAST database using ***makeblastdb*** command
- o Select appropriate BLAST tool (blastp, blastn, blastx, etc.) and tune other parameters if necessary
- o Run the tool and format the output as necessary

Global and local alignments

Using BLAST

BLAST tasks

Nucleotide (blastn) and protein (blastp) BLAST have a **task** option that sets a combination of word size and gap penalties required for a specific search type

Program	Task name	Description
blastn	blastn	Traditional nucleotide BLAST requiring exact match of 11
blastn	blastn-short	BLAST optimized for short sequences less than 50 bp
blastn	megablast	BLAST optimized to find very similar sequences (intraspecies, closely related species)
blastn	dc-megablast	Discontiguous megablast used to find more distant sequences (interspecies)
blastp	blastp	Traditional BLASTP to compare protein query to protein database
blastp	blastp-short	Optimized for sequences less than 30 residues

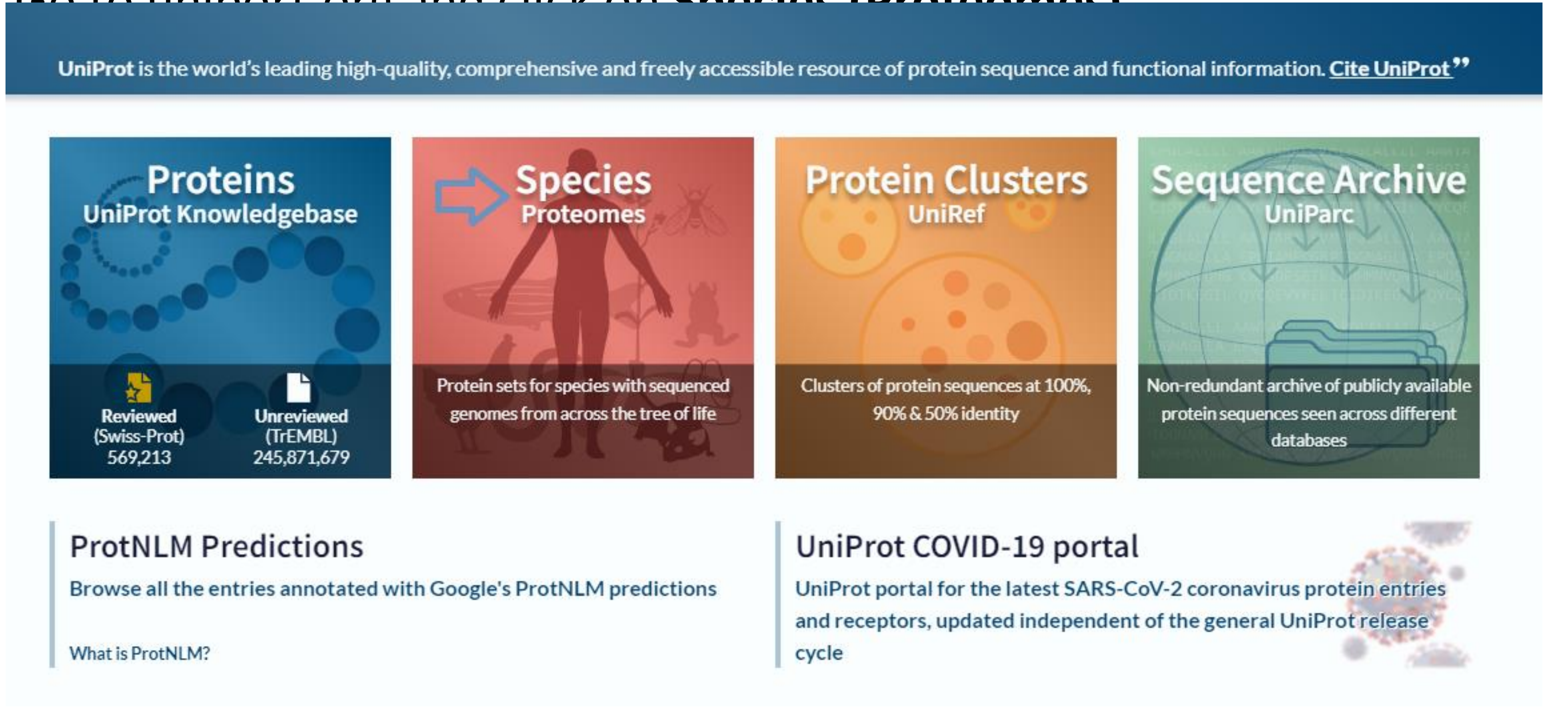
Global and local alignments

Using BLAST

Let’s practice searching for similar sequences with protein BLAST. We will compare M. tuberculosis and M. avium proteomes

First download all protein sequences for [Mycobacterium tuberculosis (strain ATCC 25618 / H37Rv) (ATCC 25618 / H37Rv)]() from Uniprot

Go to uniprot.org and click on **Species (Proteomes)**



UniProt is the world’s leading high-quality, comprehensive and freely accessible resource of protein sequence and functional information. Cite UniProt

Proteins UniProt Knowledgebase
Reviewed (Swiss-Prot) 569,213
Unreviewed (TrEMBL) 245,871,679

Species Proteomes
Protein sets for species with sequenced genomes from across the tree of life

Protein Clusters UniRef
Clusters of protein sequences at 100%, 90% & 50% identity

Sequence Archive UniParc
Non-redundant archive of publicly available protein sequences seen across different databases

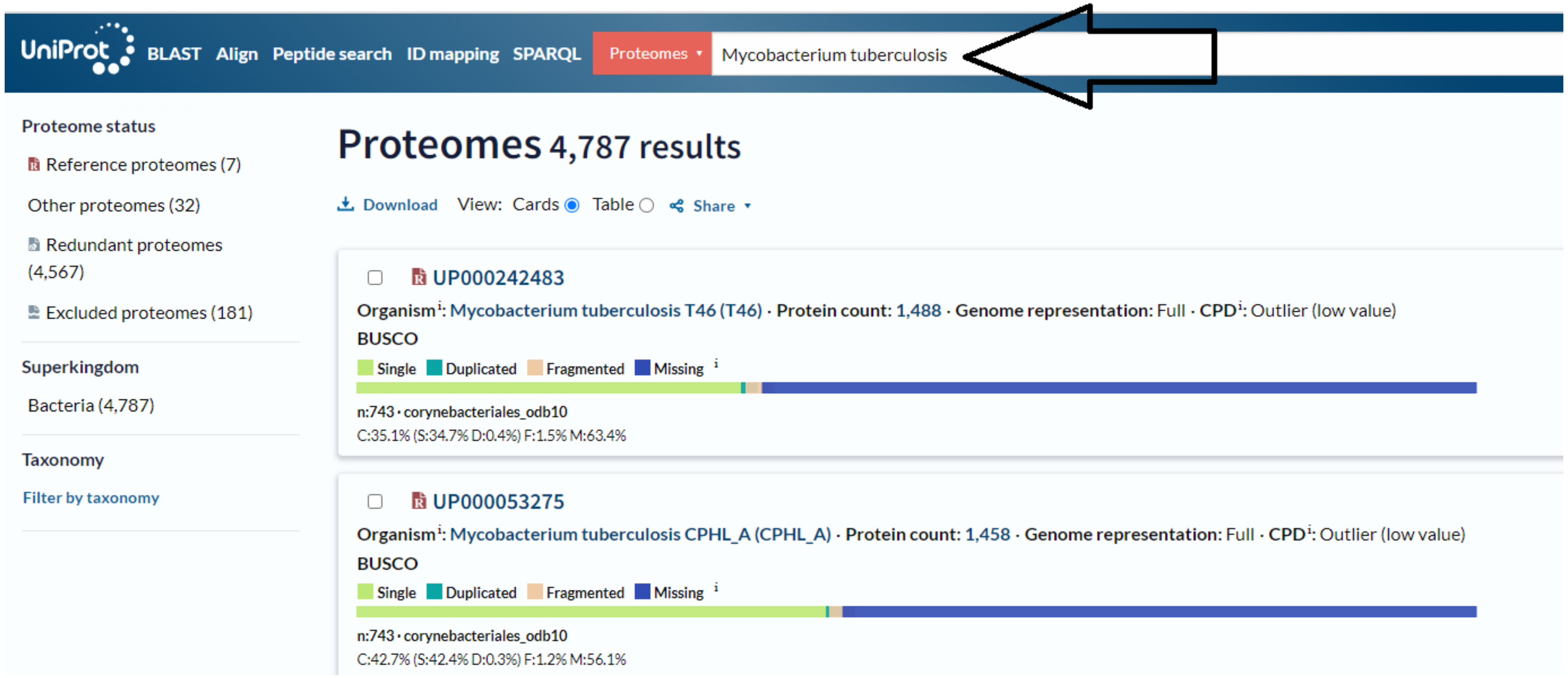
ProtNLM Predictions
Browse all the entries annotated with Google's ProtNLM predictions
What is ProtNLM?

UniProt COVID-19 portal
UniProt portal for the latest SARS-CoV-2 coronavirus protein entries and receptors, updated independent of the general UniProt release cycle

Global and local alignments

Using BLAST

Search for Mycobacterium tuberculosis



Mycobacterium tuberculosis

Proteome status

Reference proteomes (7)

Other proteomes (32)

Redundant proteomes (4,567)

Excluded proteomes (181)

Superkingdom

Bacteria (4,787)

Taxonomy

Filter by taxonomy

Proteomes 4,787 results

View: Cards [✓] Table

UP000242483

Organism: Mycobacterium tuberculosis T46 (T46) · Protein count: 1,488 · Genome representation: Full · CPD: Outlier (low value)

BUSCO

🟩 Single 🟦 Duplicated 🟧 Fragmented 🟦 Missing

n:743 · corynebacteriales_odb10

C:35.1% (S:34.7% D:0.4%) F:1.5% M:63.4%

UP000053275

Organism: Mycobacterium tuberculosis CPHL_A (CPHL_A) · Protein count: 1,458 · Genome representation: Full · CPD: Outlier (low value)

BUSCO

🟩 Single 🟦 Duplicated 🟧 Fragmented 🟦 Missing

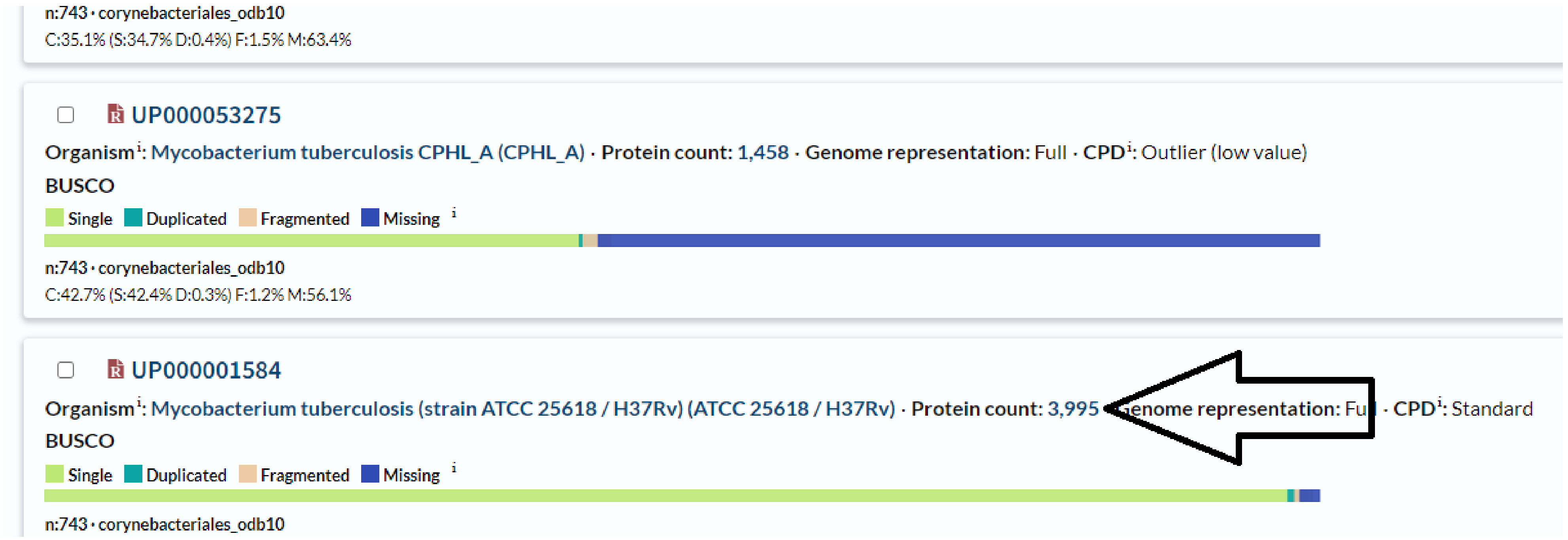
n:743 · corynebacteriales_odb10

C:42.7% (S:42.4% D:0.3%) F:1.2% M:56.1%

Global and local alignments

Using BLAST

Scroll down to Mycobacterium tuberculosis (strain ATCC 25618 / H37Rv) (ATCC 25618 / H37Rv) and press protein counts



n:743 · corynebacteriales_odb10
C:35.1% (S:34.7% D:0.4%) F:1.5% M:63.4%

UP000053275
Organism: Mycobacterium tuberculosis CPHL_A (CPHL_A) · Protein count: 1,458 · Genome representation: Full · CPD: Outlier (low value)
BUSCO
Single Duplicated Fragmented Missing
n:743 · corynebacteriales_odb10
C:42.7% (S:42.4% D:0.3%) F:1.2% M:56.1%

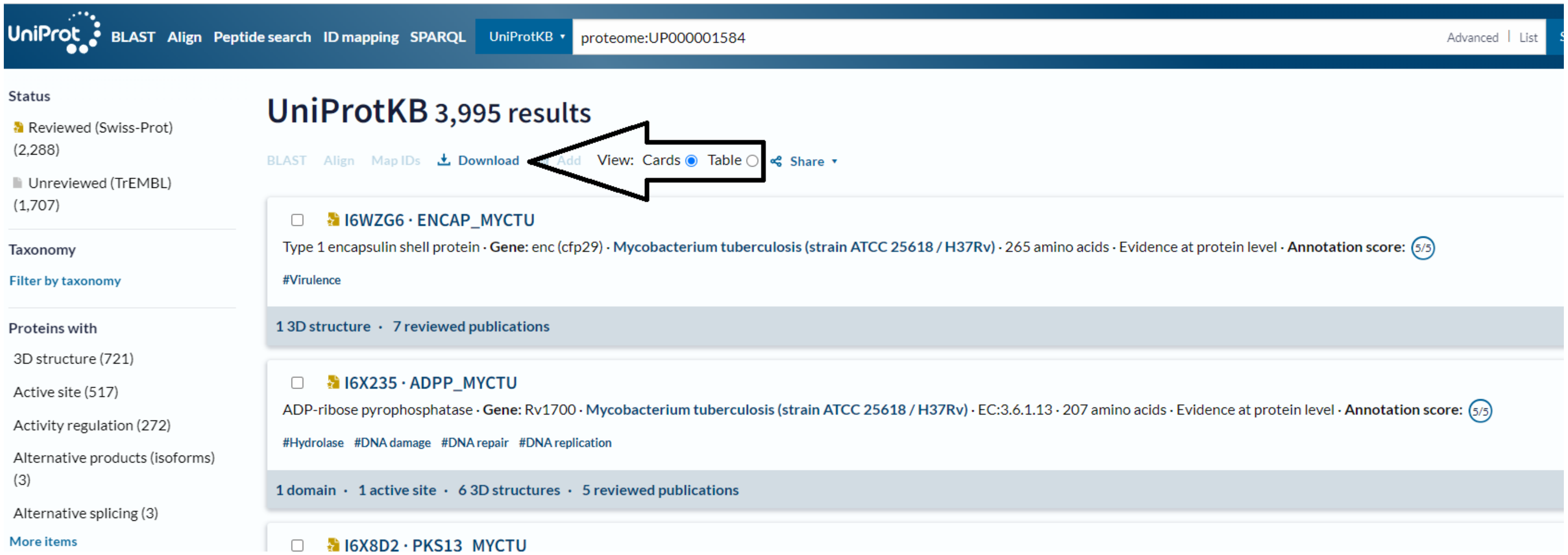
UP000001584
Organism: Mycobacterium tuberculosis (strain ATCC 25618 / H37Rv) (ATCC 25618 / H37Rv) · Protein count: 3,995 · Genome representation: Fu l · CPD: Standard
BUSCO
Single Duplicated Fragmented Missing
n:743 · corynebacteriales_odb10

Global and local alignments

Using BLAST

We will get a list of all 3995 protein sequences from this strain

Now click on download to retrieve all of these proteins in FASTA format



Status

Reviewed (Swiss-Prot) (2,288)

Unreviewed (TrEMBL) (1,707)

Taxonomy

Filter by taxonomy

Proteins with

3D structure (721)

Active site (517)

Activity regulation (272)

Alternative products (isoforms) (3)

Alternative splicing (3)

More items

UniProtKB 3,995 results

BLAST Align Map IDs Download Add View: Cards [✓ ●] Table ○ Share

I6WZG6 · ENCAP_MYCTU

Type 1 encapsulin shell protein · **Gene:** enc (cfp29) · Mycobacterium tuberculosis (strain ATCC 25618 / H37Rv) · 265 amino acids · Evidence at protein level · **Annotation score:** 5/5

#Virulence

1 3D structure · 7 reviewed publications

I6X235 · ADPP_MYCTU

ADP-ribose pyrophosphatase · **Gene:** Rv1700 · Mycobacterium tuberculosis (strain ATCC 25618 / H37Rv) · EC:3.6.1.13 · 207 amino acids · Evidence at protein level · **Annotation score:** 5/5

#Hydrolase #DNA damage #DNA repair #DNA replication

1 domain · 1 active site · 6 3D structures · 5 reviewed publications

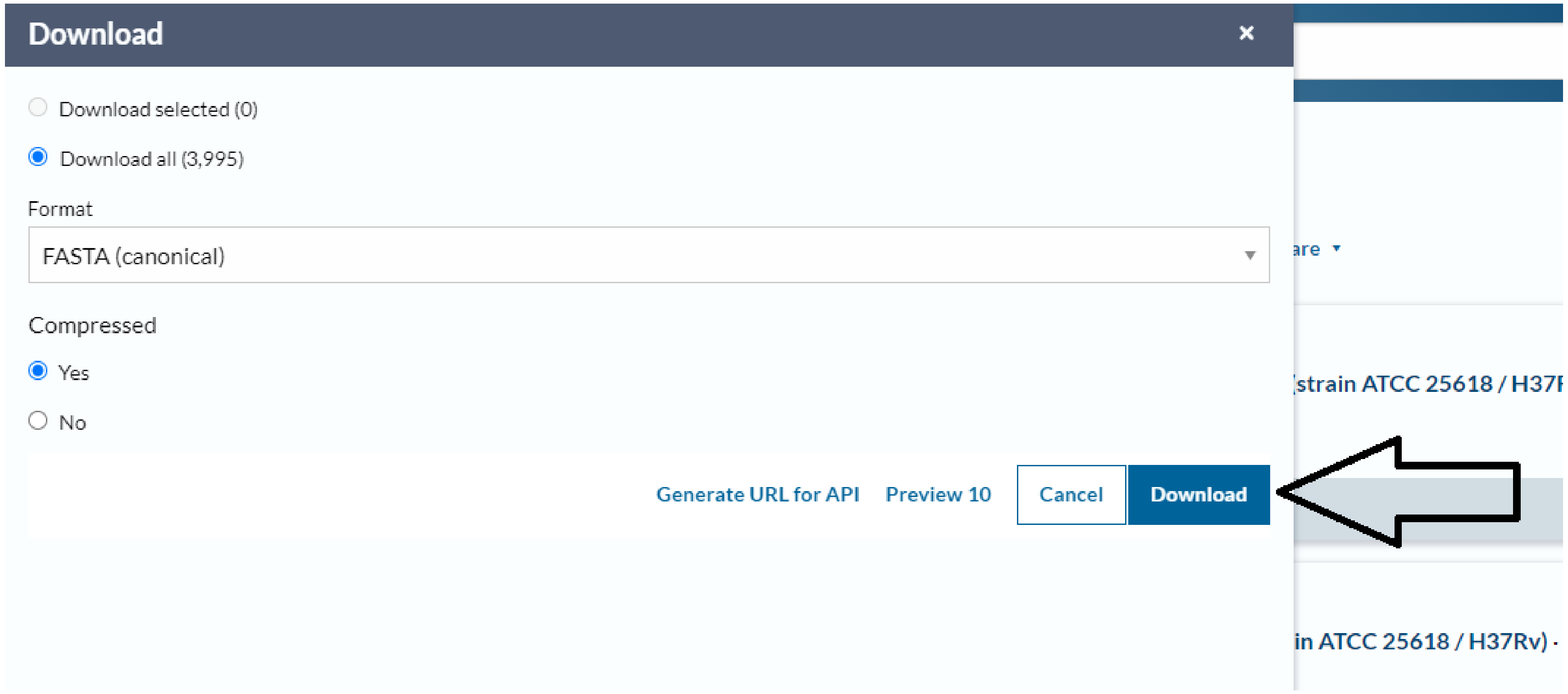
I6X8D2 · PKS13 MYCTU

Global and local alignments

Using BLAST

Download proteins in FASTA format

Repeat this step for Mycobacterium avium subsp. avium (DSM 44156)


Download

Download selected (0)

[✓ Download all (3,995)]

Format

FASTA (canonical)

Compressed

[✓ Yes] No

Global and local alignments

Using BLAST

Unzip downloaded sequences using ***gunzip*** command

```
$ gunzip <uniprot>.fasta.gz # Unzip both M. tuberculosis and M. avium files
```

Give both protein fasta files less bulky name

```
$ mv <uniport>.fasta m_tuber.fasta
$ mv <uniprot>.fasta m_avium.fasta
$ grep '>' m_tuber.fasta | wc -l # verify number of sequences
$ grep '>' m_avium.fasta | wc -l
```

We will use M. tuberculosis as target data base.
Use ***makeblastdb*** to create a database for targets

makeblastdb has a following syntax

```
makeblastdb -in <fasta_file> -out <database_name> -dbtype <type> -title <title> -parse_seqids
```

-dbtype - prot OR nucl
-parse_seqids – include target fasta ids into the output

Global and local alignments

Using BLAST

Create BLAST database from M. tuberculosis protein fasta file

```
$ mkdir ref
$ cd ref
$ makeblastdb -in ../m_tuber.fasta -out m_tuber -dbtype prot -parse_seqids
$ ls -lh # Take a look at database files
$ cd ../
```

Run protein BLAST

```
$ blastp -help # Go through the help file, try to understand options and arguments
```

Common options and arguments

-query – query sequence in fasta format

-task – task to execute

-db – path to database

-out – output file

-outfmt – output format (many variations and options)

Global and local alignments

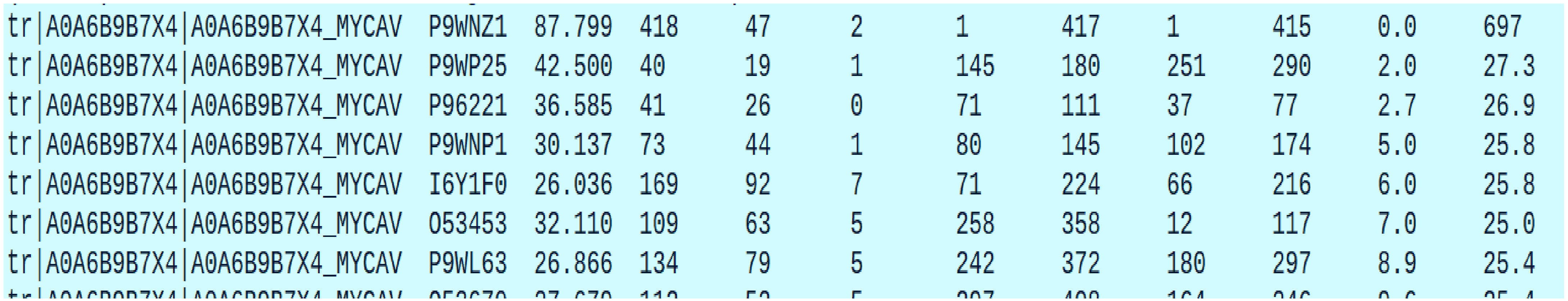
Using BLAST

Align M. avium to M. tuberculosis proteomes

```
$ blastp -task blastp -query m_avium.fasta -db ref/m_tuber -out blast6.txt -outfmt 6 -num_threads 4
```

This alignment task is very large and will take a long, press Ctrl-C to kill the process and examine the output file

```
$ head blast6.txt
```



```
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WNZ1  87.799  418     47      2       1       417     1       415     0.0     697
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WP25  42.500  40      19      1       145     180     251     290     2.0     27.3
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P96221  36.585  41      26      0       71      111     37      77      2.7     26.9
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WNP1  30.137  73      44      1       80      145     102     174     5.0     25.8
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  I6Y1F0  26.036  169     92      7       71      224     66      216     6.0     25.8
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  O53453  32.110  109     63      5       258     358     12      117     7.0     25.0
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WL63  26.866  134     79      5       242     372     180     297     8.9     25.4
```

Global and local alignments

Using BLAST
How to interpret tabular output
There are 12 columns:

Column ID	Meaning
qseqid	query ID
seseqid	subject ID
pident	percent identity
length	alignment length
mismatch	Number of mismatches
gapopen	Number of gap openings
qstart	The start of the alignment relative to query sequence
qend	The end of the alignment relative to query sequence
sstart	The start of the alignment relative to subject sequence
send	The end of the alignment relative to subject sequence
evalue	Probability of the observed alignment by chance (adjusted for the size of the database)
bitscore	Sequence similarity independent of the sequence length and database size

Global and local alignments

Using BLAST

We can instruct BLAST to include specific columns in the output

```
$ blastp -task blastp -query m_avium.fasta -db ref/m_tuber -outfmt "6 qseqid sseqid pident evalue bitscore" | head # We will get only the 5 columns specified
```

Sort the output to get matches with the highest bitscore at the top

```
$ blastp -task blastp -query m_avium.fasta -db ref/m_tuber -outfmt "6 qseqid sseqid pident evalue bitscore" | head | sort -k 5 -n -r
```

Frequently it is useful to set thresholds based on statistical parameters of alignments, for example e-value

```
$ blastp -task blastp -query m_avium.fasta -db ref/m_tuber -outfmt "6 qseqid sseqid pident evalue bitscore" -evalue 0.01 | head # this will output the alignments
```

Limit the number of alignments shown per database sequence (limit 250)

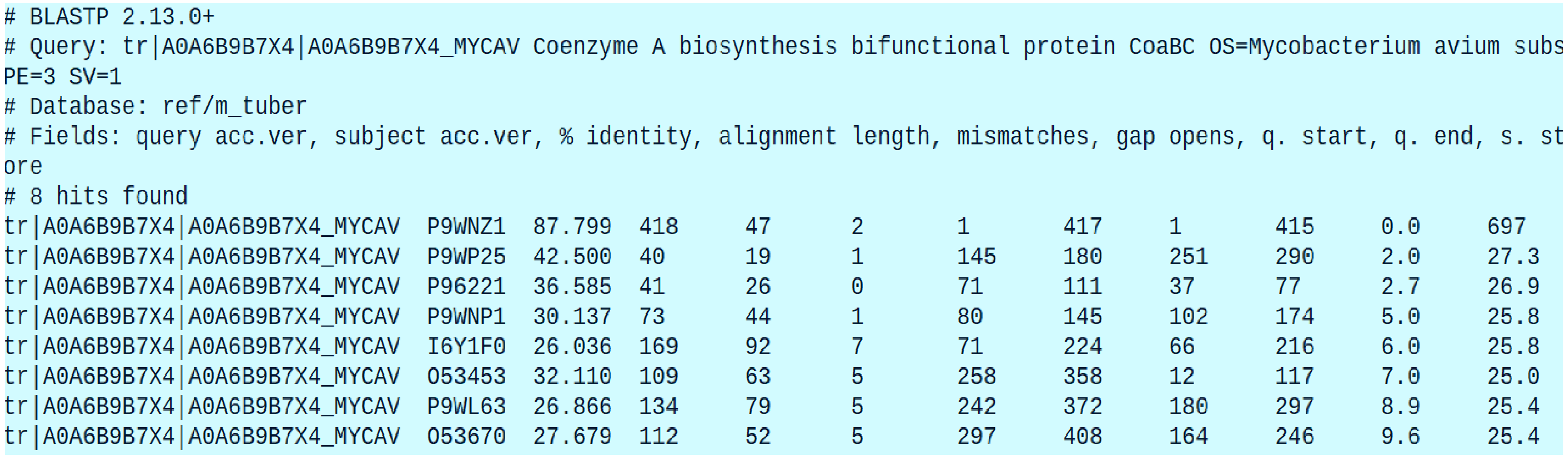
```
$ blastp -task blastp -query m_avium.fasta -db ref/m_tuber -outfmt "6 qseqid sseqid pident evalue bitscore" -num_alignments 10 | head
```

Global and local alignments

Using BLAST

Let’s check other useful formats

```
$ blastp -task blastp -query m_avium.fasta -db ref/m_tuber -outfmt 7 -num_threads 4 | head -n 50
```
Output format 7 adds some useful information



```
# BLASTP 2.13.0+
# Query: tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV Coenzyme A biosynthesis bifunctional protein CoaBC OS=Mycobacterium avium subs
PE=3 SV=1
# Database: ref/m_tuber
# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. st
ore
# 8 hits found
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WNZ1  87.799  418     47      2       1       417     1       415     0.0     697
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WP25  42.500  40      19      1       145     180     251     290     2.0     27.3
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P96221  36.585  41      26      0       71      111     37      77      2.7     26.9
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WNP1  30.137  73      44      1       80      145     102     174     5.0     25.8
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  I6Y1F0  26.036  169     92      7       71      224     66      216     6.0     25.8
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  O53453  32.110  109     63      5       258     358     12      117     7.0     25.0
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  P9WL63  26.866  134     79      5       242     372     180     297     8.9     25.4
tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV  O53670  27.679  112     52      5       297     408     164     246     9.6     25.4
```

Note that the highest scoring gene has score of 697 and evalue of 0.0, which gene is this?

```
$ grep "P9WNZ1" m_tuber.fasta
```

Global and local alignments

Using BLAST

Let’s examine the high scoring alignment between tr|A0A6B9B7X4|A0A6B9B7X4_MYCAV and P9WNZ1 in more detail

$ echo "P9WNZ1" > id_list.txt

$ blastp -task blastp -query m_avium.fasta -db ref/m_tuber -outfmt 0 -seqidlist id_list.txt -evalue 0.0001 -sorthits 1 | head -n 100 # option -**seqidlist** allows us to limit the search to a list of specific ids, we can also sort output using **-sorthits** option

```
>P9WNZ1 Coenzyme A biosynthesis bifunctional protein CoaBC OS=Mycobacterium
tuberculosis (strain ATCC 25618 / H37Rv) OX=83332 GN=coaBC
PE=1 SV=1
Length=418

 Score = 697 bits (1800),  Expect = 0.0, Method: Compositional matrix adjust.
 Identities = 367/418 (88%), Positives = 386/418 (92%), Gaps = 4/418 (1%)

Query  1    MYDRNRAFTRAARIVVGVSGGIAAYKACTVVRQLSEAGHSVRVIPTESALRFVGAATFEA  60
            M D  R      +++VGVSGGIAAYKACTVVRQL+EA H VRVIPTESALRFVGAATFEA
Sbjct  1    MVDHKRI---PKQVIVGVSGGIAAYKACTVVRQLTEASHRVRVIPTESALRFVGAATFEA  57

Query  61   LSGQPVHTGVFDDVPEVPHVQLGKQADLVVVAPATADLLARAVHGRADDLLTATLLTARC  120
            LSG+PV T VF DVP VPHV LG+QADLVVVAPATADLLARA  GRADDLLTATLLTARC
Sbjct  58   LSGEPVCTDVFADVPAVPHVHLGQQADLVVVAPATADLLARAAAGRADDLLTATLLTARC  117

Query  121  PVLFAPAMHTEMWLHPATVDNVATLRRRGAVVLEPAAGRLTGTDSGSGRLPEAEEITTLA  180
            PVLFAPAMHTEMWLHPATVDNVATLRRRGAVVLEPA GRLTG DSG+GRLPEAEEITTLA
```

Global and local alignments

Using BLAST

Compare Ebola virus proteins from Zaire obtained in 1976 to Zaire Ebola virus isolate from 2014 (example inspired by Biostars handbook)

Take a look at Ebola virus genome
https://www.ncbi.nlm.nih.gov/nuccore/AF086833.2/

Take a look at 2014 Ebola isolate
https://www.ncbi.nlm.nih.gov/nuccore/KM233118.1/

Let’s fetch this sequences using Enrez Programming Utilities
https://www.ncbi.nlm.nih.gov/books/NBK25501/

Fetch protein sequences using Entrez E-Utilities, break this command into stages to view intermediate results

```
$ esearch -db nuccore -query AF086833.2 | elink -target protein | efetch -format fasta > ebola1976.fasta
$ esearch -db nuccore -query KM233118.1 | elink -target protein | efetch -format fasta > ebola2014.fasta
```

Global and local alignments

Using BLAST

Compare Ebola virus proteins from Zaire obtained in 1976 to Zaire Ebola virus isolate from 2014 (example inspired by Biostars handbook)

Take a look at Ebola virus genome
https://www.ncbi.nlm.nih.gov/nuccore/AF086833.2/

Take a look at 2014 Ebola isolate
https://www.ncbi.nlm.nih.gov/nuccore/KM233118.1/

Let's fetch this sequences using Enrez Programming Utilities
https://www.ncbi.nlm.nih.gov/books/NBK25501/

Fetch protein sequences using Entrez E-Utilities, break this command into stages to view intermediate results

```
$ esearch -db nuccore -query AF086833.2 | elink -target protein | efetch -format fasta > ebola1976.fasta
$ esearch -db nuccore -query KM233118.1 | elink -target protein | efetch -format fasta > ebola2014.fasta
```

Global and local alignments

Using BLAST

Compare Ebola virus proteins from Zaire obtained in 1976 to Zaire Ebola virus isolate from 2014 (example inspired by Biostars handbook)

Let’s create a BLAST database from Ebola 1976 proteins

```
$ mkdir ebola1976_ref
$ cd ebola1976_ref
$ ls -lh
$ cd ../
```

Compare both proteomes and examine the output

```
$ blastp -task blastp -query ebola2014.fasta -db ebola1976_ref/ebola1976 -outfmt 0
```

This will match every protein of the query to every protein in the target database, try to limit the search to high similarity hits with -evalue option

```
$ blastp -task blastp -query ebola2014.fasta -db ebola1976_ref/ebola1976 -outfmt 0 -evalue 0.01
```

Global and local alignments

Using BLAST

Default behavior of BLAST with repeats and how to overcome it (example from Biostars handbook)

Let’s fetch yeast chromosome 1

```
$ efetch -db nuccore -id NC_001133 -format fasta > NC_001133.fasta
$ head NC_001133.fasta
```

Let’s extract first 60 bases and save than in a separate fasta file

```
$ echo -e "NC_001133.9\t0\t59" > subseq.bed # First we will create a BED file with coordinates
$ echo -e "NC_001133.9\t0\t59" # check this part of the command above
$ echo  "NC_001133.9\t0\t59" # what will happen if you skip -e (evaluate option)
```

Now, extract the first 60 bp using ***seqtk.*** Note, that the coordinates in this case are 0-based, so we will need to specify the interval as **0 – 59** bp

```
$ seqtk subseq NC_001133.fasta subseq.bed > subseq.fasta
$ cat subseq.fasta
$ grep 'CCA' subseq.fasta | wc -c # Let’s verify the length of the sequence
```

Global and local alignments

Using BLAST

Default behavior of BLAST with repeats and how to overcome it (example from Biostars handbook)

Let’s create BLAST database for yeast chromosome I

```
$ mkdir yeast_ref
$ makeblastdb -dbtype nucl -in NC_001133.fasta -out yeast_ref/NC_001133
$ ls -lh yeast_ref
```

Try to map first 60 bases to chromosome I

```
$ blastn -db yeast_ref/NC_001133 -query subseq.fasta # No hits found!
```

But grep finds this match with no problem

```
$ grep --color="auto" -A 1 -B 1
'CCACACCACACCCACACACCCACACACCACACCACACACCACACCACACCCACACACAC' NC_001133.fasta
```

By default, BLAST silently and automatically filters out hits on the repetitive regions

```
$ blastn -db yeast_ref/NC_001133 -query subseq.fasta -dust no
```

# No we get the expected hit! It is important to be avare of this behaviour when mapping sequences with repeats

Global and local alignments

Using BLAST

Extracting data from BLAST databases and using pre-built databases

We can manipulate BLAST databases with ***blastdbcmd*** command

Syntax
blastdbcmd -db <database> -entry <pattern> -out <file>

-db – database name
-entry – pattern to search
-out – file name for output, otherwise print to StdIn

- Before using ***blastdbcmd,*** take a look at BLAST ftp site that contains lots of useful resources including documentation, code, and pre-built databases

- Open https://ftp.ncbi.nlm.nih.gov/blast/

- Examine *blastftp.txt* file, files in *documents* folder, *db* folder and others

Global and local alignments

Using BLAST
Extracting data from BLAST databases and using pre-built databases

Let's go to *db/* folder at https://ftp.ncbi.nlm.nih.gov/blast/ and download a pre-built database for 18S fungal ribosomal sequences

You can right-click on the desired file and press *Copy link address;* use this link to download the file with ***wget***

```
$ wget https://ftp.ncbi.nlm.nih.gov/blast/db/18S_fungal_sequences.tar.gz
$ mkdir ribo_db # create a directory to store the database
$ cd ribo_db
$ tar -xzvf ../18S_fungal_sequences.tar.gz # untar and uncompress the files
$ ls -lh # check the results
```

We can recover all sequences from the database

```
$ cd ../
$ blastdbcmd -db ribo_db/18S_fungal_sequences -entry all -out 18S_fungal_seq.fa
$ head 18S_fungal_seq.fa # check the output
$ grep '>' 18S_fungal_seq.fa | wc -l # how many sequences do we have?
```

Global and local alignments

Using BLAST

Extracting data from BLAST databases and using pre-built databases

We can extract specific sequences

```
$ blastdbcmd -db ribo_db/18S_fungal_sequences -entry 'NG_063391.1’ # extract a single sequence
$ blastdbcmd -db ribo_db/18S_fungal_sequences -entry 'NG_070171.1,NG_070172.1’ # extract multiple sequences with comma separated entries
```

Which taxa are present among the sequences

```
$ blastdbcmd -db ribo_db/18S_fungal_sequences -tax_info # Print taxonomic information
$ blastdbcmd -db ribo_db/18S_fungal_sequences -taxids '3003221,3003220’ # Print 18S sequences for 2 species of Polyporus
```

Extract range of bases

```
$ blastdbcmd -db ribo_db/18S_fungal_sequences -taxids '3003221,3003220' -range 1-50 # Extract first 50 bases from 2 species of Polyporus
```

Examine the wealth of options in blastdbcmd

```
$ blastdbcmd -help | less
```

Global and local alignments

Using BLAST

A guide to BLAST utilities

Tool name	Alignment (Query → Subjct)	Alignment level
blastn	Nucleotide → Nucleotide	Nucleotide
blastp	Protein → Protein	Peptide
blastx	Nucleotide → Peptide	Peptide
tblastn	Peptide → Nucleotide	Peptide
tblastx	Nucleotide → Nucleotide	Peptide