Lesson 4 - Genomics

Sequence mapping part I

Working with Tabular files

- TSV = tab separated values, CSV=comma separated values
- A simple way to store tabular data

```
# View a file without line wrapping
$ less -S table.tsv
# Extract specific column from TSV
$ cut -f 2 table.tsv
# Extract multiple columns from TSV
$ cut -f 2,6,7 table.tsv
# Sort a table by column
$ sort -k 2 table.tsv | less -S
# Get unique values of a column
$ cut -f 2 table.tsv | sort | uniq
# Count occurrences of unique values
$ cut -f 2 table.tsv | sort | uniq -c
```

By the end of this lesson you will...

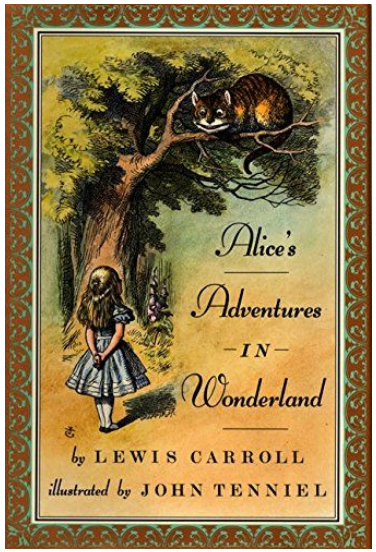
- Understand the concept of sequence mapping and alignment

  https://www.annualreviews.org/doi/full/10.1146/annurev-genom-090413-025358

- Be familiar with the basic Blast algorithm
  - Parameters
  - Outputs

- Know how to use Blast from the command line

Imagine we have a big book...



... and we want to search it for a specific sentence

It would be
“ so nice if
something
made sense
for a
change.

Lewis Carroll
Alice in Wonderland

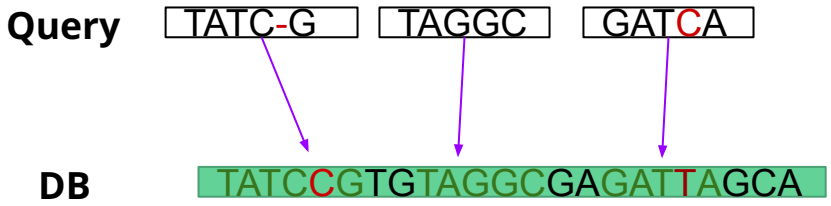
- How can we do it in a timely manner?
  - Brute force
  - Indexing
- Do we allow slight changes?

  e.g. : “*it **c**ould be so nice if something made sense*”
- Do we allow insertions and deletions?

  e.g. : “*it would be ~~**so**~~ nice if something made **a little** sense*”
- What if the sentence is repeated in several places in the book?

Sequence mapping

- Detecting the position of a **query** sequence within a **database (DB)** of sequences
- Amino acid or nucleotide alphabet
- Searching for exact matches - easier but less useful
- Allowing **mismatches** and **InDels** - adds complexity



DB and query types

- DB may be:
  - A whole genome sequence (reference)
  - Whole proteome / transcriptome
  - Collection of genes / proteins from multiple organisms (UniProt, RefSeq)

- Query may be:
  - Gene / protein sequence
  - NGS read

RefSeq

Why do we need sequence mapping?

- Determine the origin of an unknown sequence
- Find homologous sequences
- Determine genomic position of a sequence
- Identify genomic variants between samples (variant calling)
- Determine the function of a sequence (annotation)

Sequence mapping - challenges

- Large DBs - millions to billions of nucleotides/AAs
- Repetition - biological sequences tend to repeat
- Noisy - sequencing errors and real biological variants

Two stages of sequence mapping

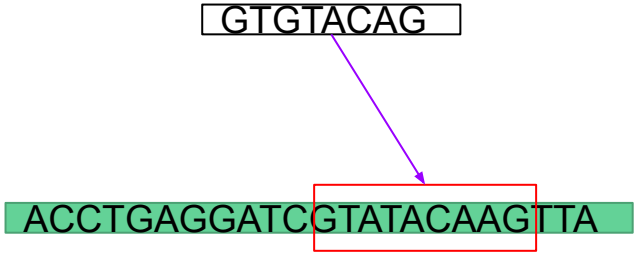
1. **SEARCH** -

   Roughly find the position of the query in the DB

2. **ALIGN** -

   Find the exact pairwise alignment of

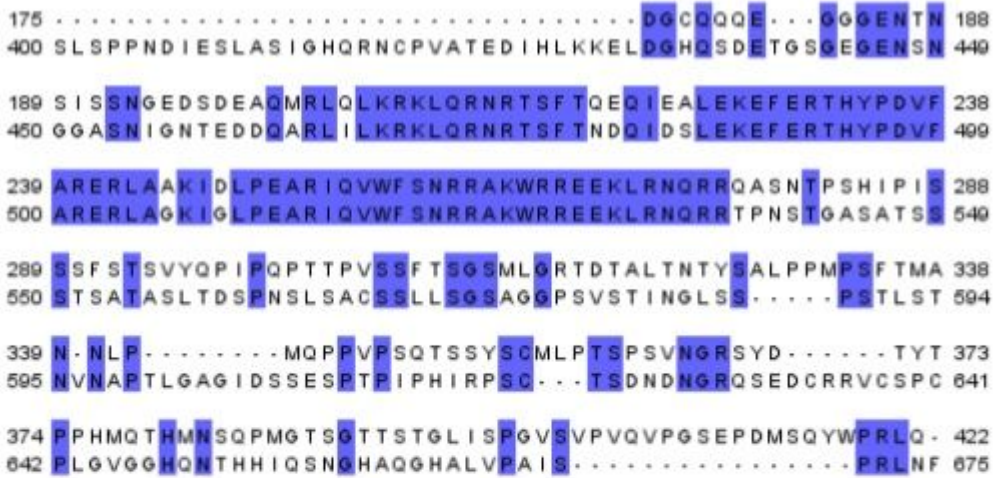
   the query and the DB sequences



G	T	G	T	A	C	A	-	G
G	T	A	T	A	C	A	A	G

Searching for imperfect matches - intuition

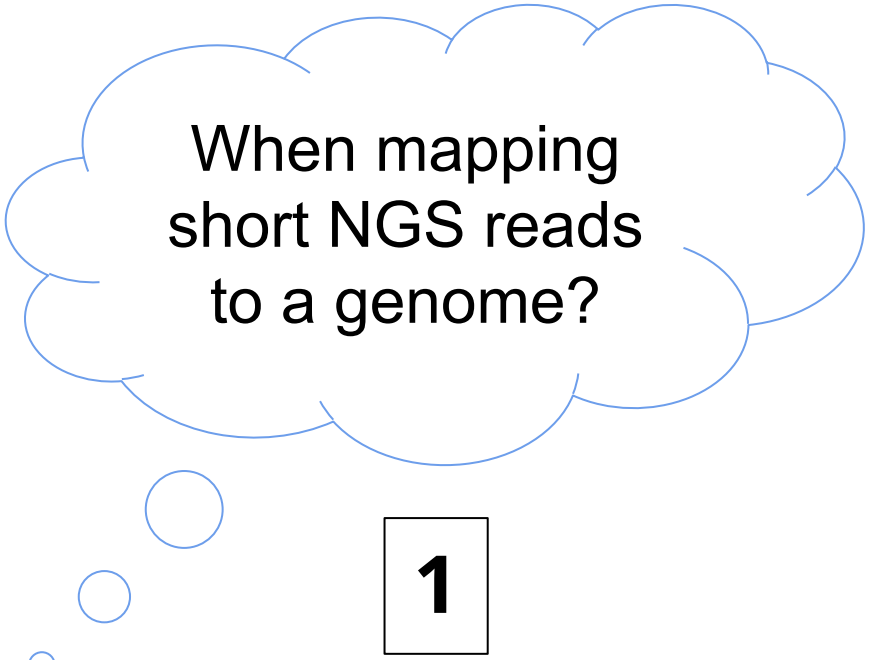
A good match should have lots of short **exact** matches, called **seeds**

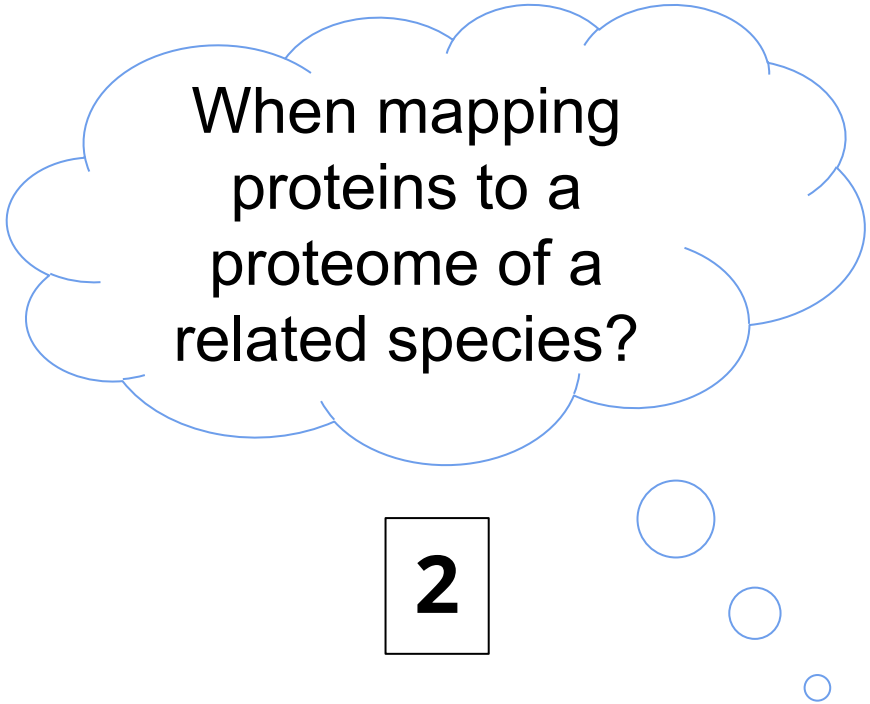


Local vs. global alignment

- **Global alignment** - try to match entire sequences
  Useful for closely-related sequences of similar size
- **Local alignment** - allow partial matching
  Useful for sequences expected to contain some similarity regions

Global or local?





Alignment scoring

- Scoring matrix

EEELTKPRLLWALYFNMRDALSSG-

---VEKPRILYALYFNMRD--SSDE

	A	G	C	T
A	10	-1	-3	-4
G	-1	7	-5	-3
C	-3	-5	9	0
T	-4	-3	0	8

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	−1	5																		
Asn	−2	0	6																	
Asp	−2	−2	1	6																
Cys	0	−3	−3	−3	9															
Gln	−1	1	0	0	−3	5														
Glu	−1	0	0	2	−4	2	5													
Gly	0	−2	0	−1	−3	−2	−2	6												
His	−2	0	1	−1	−3	0	0	−2	8											
Ile	−1	−3	−3	−3	−1	−3	−3	−4	−3	4										
Leu	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	4									
Lys	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	5								
Met	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	5							
Phe	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	6						
Pro	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	7					
Ser	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	4				
Thr	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	5			
Trp	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	11		
Tyr	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	7	
Val	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	4

- Gap (InDel) penalty - gap open / gap extension

BLAST - Basic Local Alignment Search Tool

- One of the most popular bioinformatic tools
- BLAST finds regions of similarity between biological sequences.
- Compares nucleotide or protein sequences to sequence databases
- Calculates the statistical significance of DB hits
- Allows searching for **imperfect** sequence matches
- Uses a **heuristic** algorithm to improve efficiency

BLAST algorithm steps

1. Index the DB
2. Generate query words
3. compute neighbourhood words
4. Search the DB for exact word matches - seeds
5. Elongate and combine seeds to get final alignment
6. Score alignment

BLAST - indexing the DB

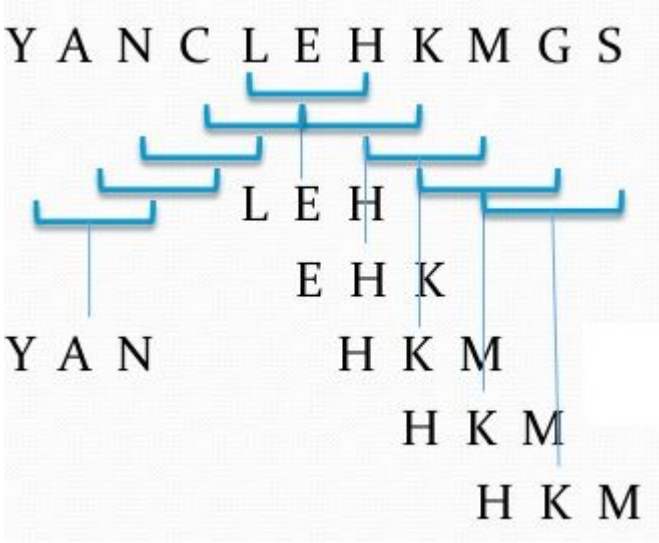
- Only needed the first time a DB is used
- Mask repetitive and low-complexity regions - `ATATATTTATT → atatatttatt`
- Break DB sequences into overlapping words of length *W*
  - W=3 for amino acids
  - W=11 for nucleotides
- Create a lookup table of words with their positions

BLAST - breaking query to words

A query of length *L* produces *L-W+1* **overlapping** words of length *W*

L = 11

W = 3



BLAST - finding neighbourhood words

1. For each word, find all neighbourhood words

   = words with one change

2. Use a scoring matrix to assign each neighbourhood word a score
3. Discard neighbourhood words with score < *T*

Y A N C L E H K M G S

L E H

L K H
C E H
Q E H
L M H
L F H
L E R
D E H
...

BLOSUM 62 scoring matrix

(positive values are shaded)

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	4																			
R	-1	5																		
N	-2	0	6																	
D	-2	-2	1	6																
C	0	-3	-3	-3	9															
Q	-1	1	0	0	-3	5														
E	-1	0	0	2	-4	2	5													
G	0	-2	0	-1	-3	-2	-2	6												
H	-2	0	1	-1	-3	0	0	-2	8											
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

The values for amino acid substitutions were obtained from Henikoff S & Henikoff JG (1992) Amino acid substitutions matrices from protein blocks. *Proc. Natl. Acad. Sci.* **89**: 10915-10919.

```
L  E  H
4+1+8 = 13
L  K  H
```

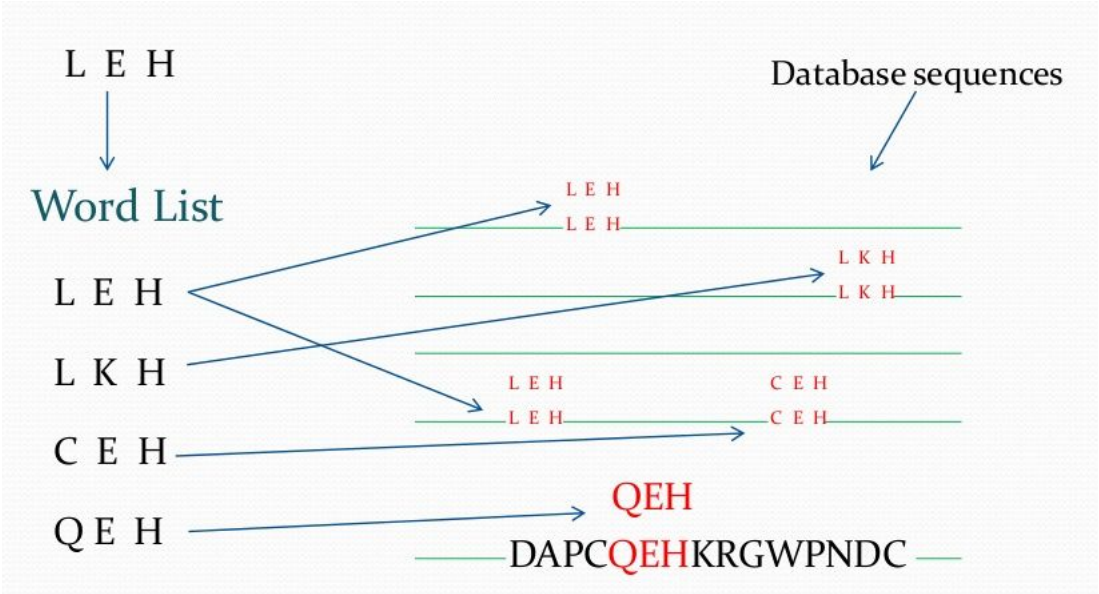
L K H	13
C E H	12
Q E H	11
L M H	10
L F H	9
L E R	9
D E H	9
...	

T=11

L K H	13
C E H	12
Q E H	11
L M H	10
L F H	9
L E R	9
D E H	9
...	

BLAST - finding alignment seeds in DB

- Look for **exact** matches of query words with the DB words
- Masked regions are ignored

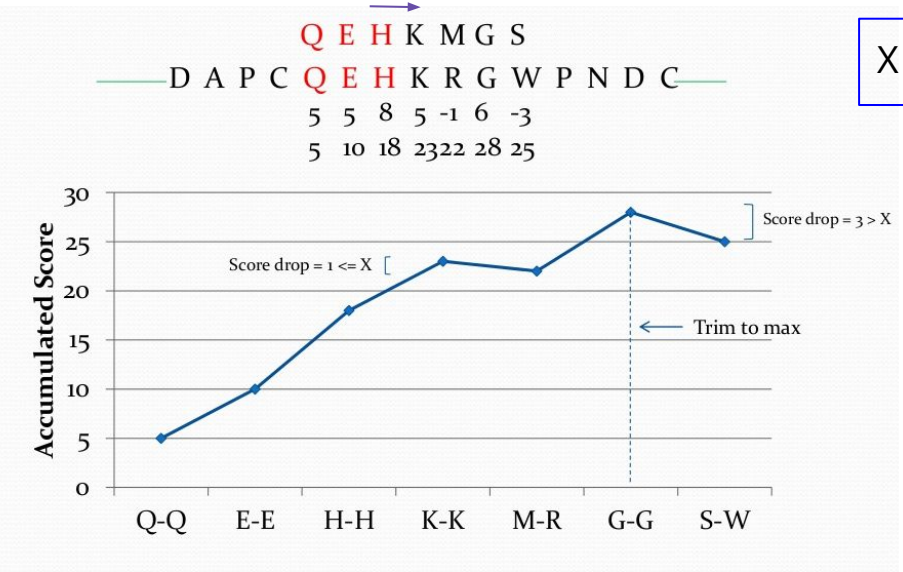


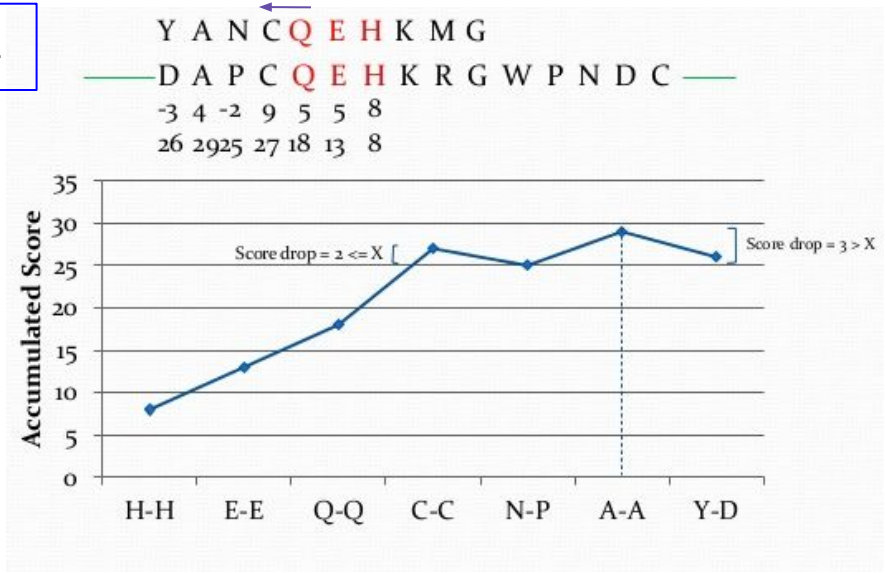
BLAST - seed elongation

Elongate each seed to both directions until a score drop > *X* is encountered

Query:
YANC**LEH**KMGS

X = 2





BLAST - scoring the alignment

- Calculate total alignment score

- Discard alignments with score < *S*
- Remaining alignments are called High scoring Sequence Pairs - **HSP**s

BLAST - scoring the alignment

- Calculate alignment **bit score**
  - Independent of query length
  - Independent of DB size

$$S' = \frac{\lambda S - \ln(K)}{\ln(2)}$$

- Calculate **E-value** - the number of hits with score >= s that one can expect to find in DB by chance

L - query length , *N* - DB length , *S'* - bit score

$$E = \frac{L \cdot N}{2^{S'}}$$

Smaller E ➔ better hit

Why do we use different *W* for nucleotides and AAs?

$W_{prot} = 3$
$W_{nuc} = 11$

Different alphabet size!
What is the probability to find a match of length 3 by chance?

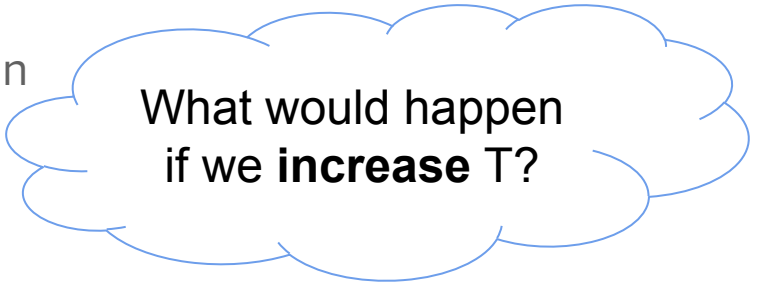
Protein: $(1/20)^3 = 0.000125$

Nucleotides: $(1/4)^3 = 0.015625$

Lots of “noise” seeds. If we increase to W=11 → $(1/4)^{11} = 0.000000238$
Also – DNA sequences are usually longer than protein sequences
Smaller W → higher sensitivity, but slower runs

Summary – BLAST parameters

- *W* – word size (query and DB)
- *T* – neighborhood words score cutoff
- *X* – allowed score drop during seed elongation
- *S* – HSP score cutoff



BLAST programs

Program	Database	Query
BLASTN	Nucleotide	Nucleotide
BLASTP	Protein	Protein
BLASTX	Protein	Nucleotide translated into protein
TBLASTN	Nucleotide translated into protein	Protein
TBLASTX	Nucleotide translated into protein	Nucleotide translated into protein

The Blast command line software

- Commands:

  `blastn , blastp , blastx , tblastn , tblastx , makeblastdb`

- Use the -help flag to get the full usage instructions

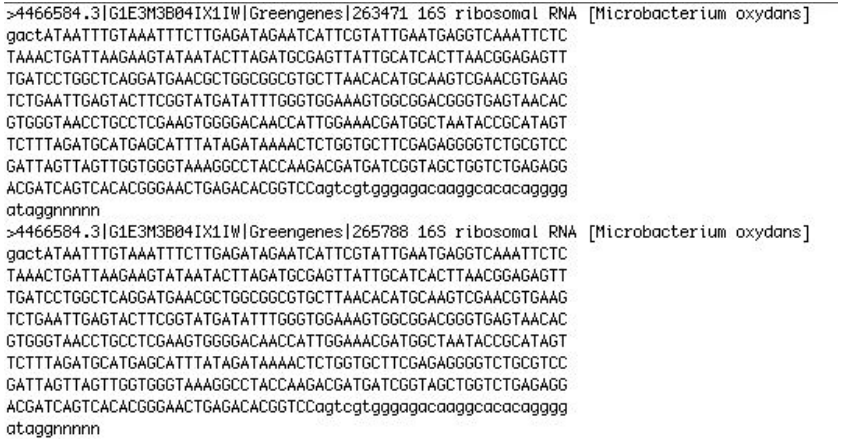
```
$ blastn -help | less
```

Creating a blast DB

- Mandatory arguments:
  - DB type - nucleotides or proteins
  - Input file - path to fasta

Example:

```
$ makeblastdb -in my_genome.fa -dbtype nucl
```



Running blast queries (e.g. blastn)

- Mandatory arguments
  - Query - fasta file with one or more records
  - DB - blast DB name
  - Output - where to write output
- Other important parameters:
  - E-value threshold - `-evalue`
  - Max number of DB hits per query - `-max_target_seqs`

Example:

```
$ blastn -query seq.fa -db my_genome.fa -out
blast_result -evalue 0.0001
```

Blast output

- Blast can generate output in multiple formats
- Controlled via the `-outfmt` parameter

```
-outfmt <String>
  alignment view options:
    0 = Pairwise,
    1 = Query-anchored showing identities,
    2 = Query-anchored no identities,
    3 = Flat query-anchored showing identities,
    4 = Flat query-anchored no identities,
    5 = BLAST XML,
    6 = Tabular,
    7 = Tabular with comment lines,
    8 = Seqalign (Text ASN.1),
    9 = Seqalign (Binary ASN.1),
   10 = Comma-separated values,
   11 = BLAST archive (ASN.1),
   12 = Seqalign (JSON),
   13 = Multiple-file BLAST JSON,
   14 = Multiple-file BLAST XML2,
   15 = Single-file BLAST JSON,
   16 = Single-file BLAST XML2,
   17 = Sequence Alignment/Map (SAM),
   18 = Organism Report
```

Blast output format 0 - pairwise (default)

```
>sp|P19767|INSA7_ECOLI Insertion element IS1 7 protein InsA OS=Escherichia
coli (strain K12) OX=83333 GN=insA7 PE=3 SV=1
Length=91
          Bitscore   Score          E-value

 Score = 178 bits (452),  Expect = 2e-61, Method: Compositional matrix adjust.
 Identities = 84/91 (92%), Positives = 88/91 (97%), Gaps = 0/91 (0%)

Query  1   MASVSISCPSCSATDGVVRNGKSTAGHQRYLCSHCRKTWQLQFTYTASQPGTHQKIIDMA  60
           MAS+SI CPSCSAT+GVVRNGKSTAGHQRYLCS CRKTWQLQFTYTASQPG HQKIIDMA
Sbjct  1   MASISIRCPSCSATEGVVRNGKSTAGHQRYLCSPCRKTWQLQFTYTASQPGKHQKIIDMA  60

Query  61  MNGVGCRATARIMGVGLNTILRHLKNSGRSR  91
           MNGVGCRA+ARIMGVGLNT+LRHLKNSGRSR
Sbjct  61  MNGVGCRASARIMGVGLNTVLRHLKNSGRSR  91
```

Blast output format 6 - tabular

- Tab-separated values (TSV)
- Easier to read/parse
- Displayed columns can be configured

```
# Fields: query acc.ver, subject acc.ver, % identity, alignment length, mismatches, gap opens, q. start, q. end, s. start
, s. end, evalue, bit score
# 15 hits found
sp|A0A385XJ53|INSA9_ECOLI       sp|P0CF12|INSA6_ECOLI   100.000 91      0       0       1       91      1       91      9
.66e-66         189
sp|A0A385XJ53|INSA9_ECOLI       sp|P0CF11|INSA5_ECOLI   100.000 91      0       0       1       91      1       91      9
.66e-66         189
sp|A0A385XJ53|INSA9_ECOLI       sp|P0CF07|INSA1_ECOLI   100.000 91      0       0       1       91      1       91      9
.66e-66         189
sp|A0A385XJ53|INSA9_ECOLI       sp|A0A385XJ53|INSA9_ECOLI       100.000 91      0       0       1       91      1       9
1       9.66e-66        189
sp|A0A385XJ53|INSA9_ECOLI       sp|P0CF10|INSA4_ECOLI   98.901  91      1       0       1       91      1       91      3
.20e-65         188
sp|A0A385XJ53|INSA9_ECOLI       sp|P0CF09|INSA3_ECOLI   98.901  91      1       0       1       91      1       91      3
.20e-65         188
```