Alignment (I)

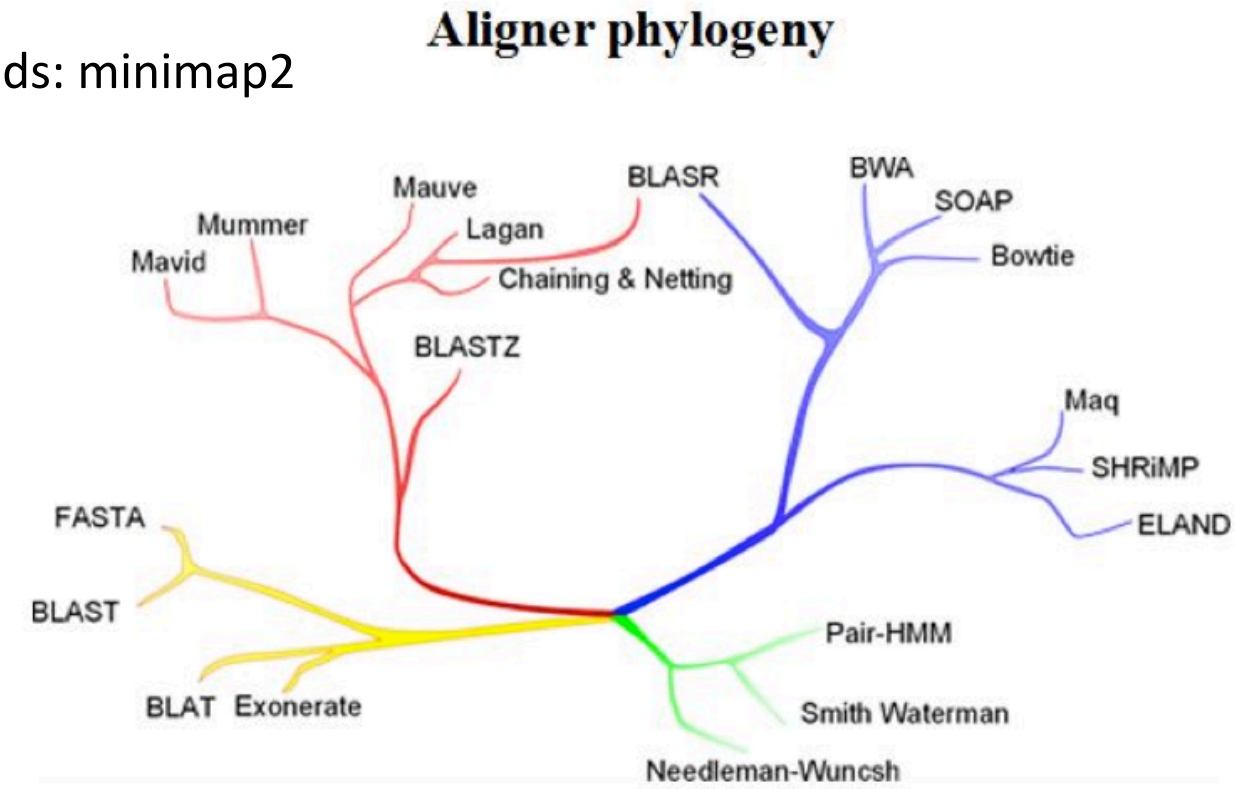
Bioinformatics Applications (PLPTH813)

Sanzhen Liu

2/14/2019

Alignment algorithms

long noisy reads: minimap2



Long sequences	Short reads
Pairwise heuristic	Sensitive aligners

http://evomicsorg.wpengine.netdna-cdn.com/wp-content/uploads/2014/01/alignCompare2.jpg

Outline

- Alignment overview
- Dot plot
- Dynamic alignment

(example: local alignment)

- BLAST

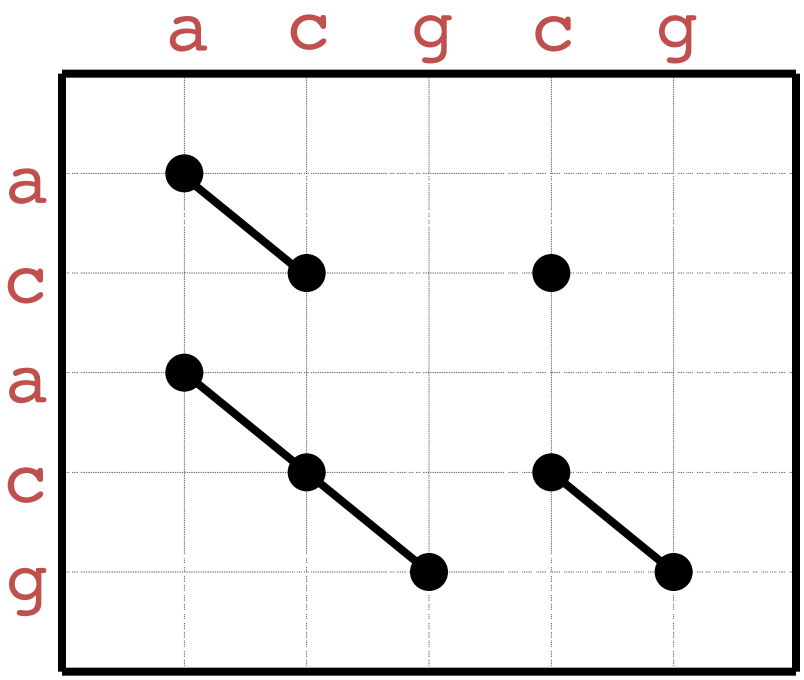
Sequence alignment

Sequence alignment is the approach of comparing the sequences of nucleotides or amino acids to identify regions of similarity.

Applications:

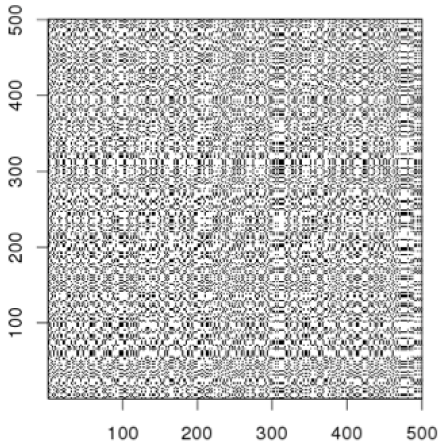
1. Measure relatedness between sequences
2. Identify homologous genes or duplication regions
3. Identify source of a sequence in a database
4. Locate the position of a sequence in the genome
5. etc.

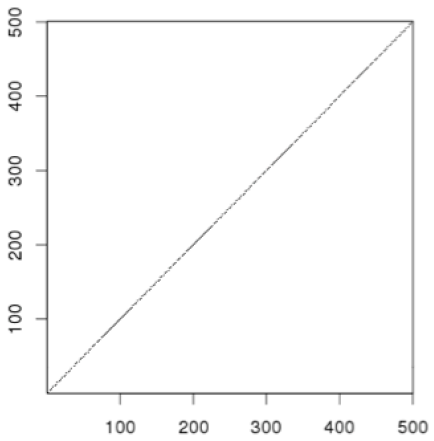
Dot matrices in a single-base resolution



Dot plot comparison using windows

- Dot matrices for long sequences can be noisy due to insignificant matches





e.g., Put a dot/line only if at least 9 out of 10 nucleotides are identical.

window size = 10

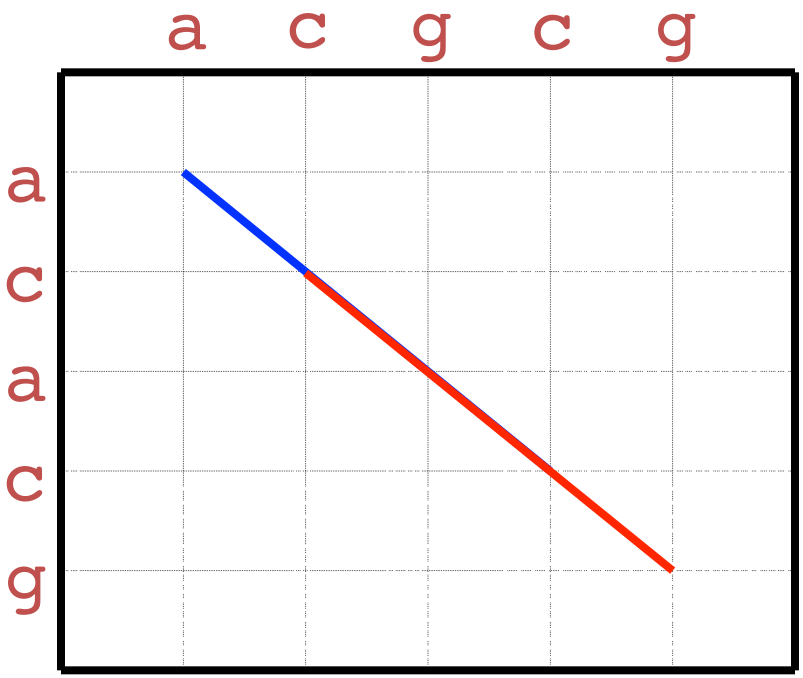
min matches = 9

- Solution: use a window and a threshold
  - compare letter by letter within a window (have to choose window size)
  - require certain fraction of matches within window in order to display it with a dot

Dot plot with a window method

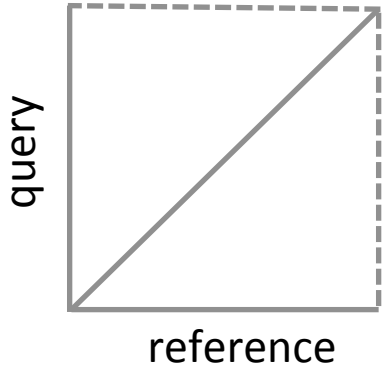
Window size = 4

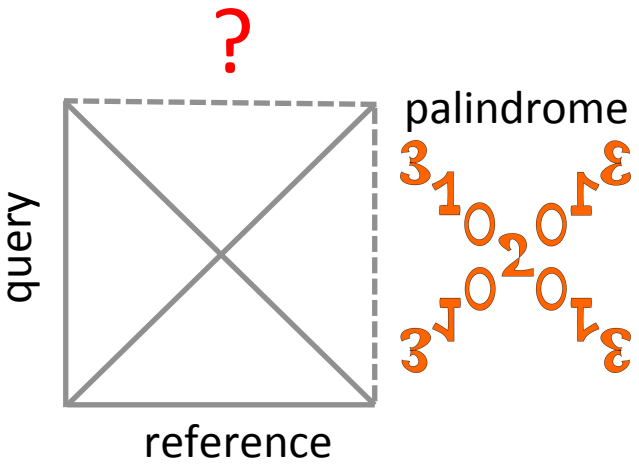
Stringency = 3 (min matches)

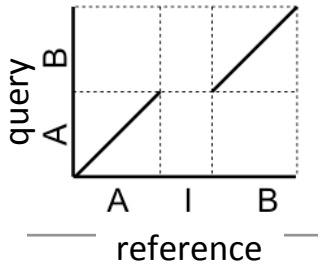


Dot-plots (examples)

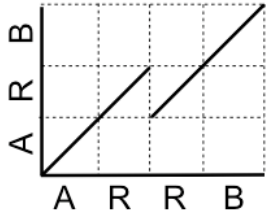
query is identical to reference
and contains no repeats



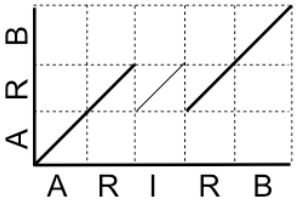




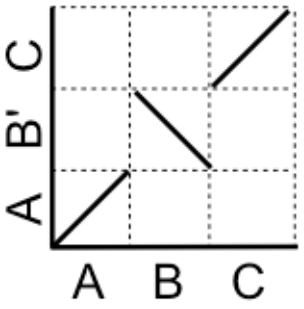
deletion of "I" in query



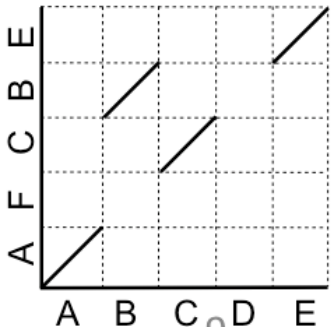
deletion of a "R" in query



deletion of "IR" in query



inverstion



Rearrangement
with disagreement

Outline

- Alignment overview
- Dot plot
- Dynamic alignment

(example: local alignment)

- BLAST

Local and global alignments

- Local alignment: to find similar sequence regions between sequences

```
C  T  G  T  T  G  C  T  G  C
         T  G  C  T  G
```

- Global alignment: to attempt to optimally align the entire length of two sequences.

```
C  T  G  T  T  G  C  T  G  C
-  T  G  -  -  -  C  T  G  -
```

Best (local) alignment

Question: How to determine which alignment is better?

```
Alignment 1: C  T  G  T  T  G  C  T  G  C
                T  G  C  T  G

Alignment 2: C  T  G  T  T  G  C  T  G  C
                T  G  -  -  -  C  T  G
```

Need a scoring scheme:

e.g., match +1; mismatch -1; gap -2

then, a score can be assigned to each alignment

Best (local) alignment

match +1; mismatch -1; gap -2

```
Alignment 1: C  T  G  T  T  G  C  T  G  C
             T  G  C  T  G
             1  2  1  2  3                 score = 3
Alignment 2: C  T  G  T  T  G  C  T  G  C
             T  G  -  -  -  C  T  G
             1  2  0 -2 -4 -3 -2 -1        score = -1
```

match +1; mismatch **-2**; gap **0**

```
Alignment 1: C  T  G  T  T  G  C  T  G  C
             T  G  C  T  G
             1  2  0  1  1                 score = 2
Alignment 2: C  T  G  T  T  G  C  T  G  C
             T  G  -  -  -  C  T  G
             1  2  0  0  0  3  4  5        score = 5
```

A classic algorithm for local alignment – Smith-Waterman

Local alignment

```
C  T  G  T  T  G  C  T  G  C
         T  G  C  T  G
```

List all possible alignments and to find the winner with the highest score?

Smith–Waterman (SW)

Using dynamic programming to find the best local alignment(s) between two sequences with respect to a scoring scheme

SW example

Question: to find the optimal local alignments between s and t.

```
s: CTGTTGCT
t: ATGCTGCA
```

Scoring rule:
match +1; mismatch -1; gap -2 (γ)

1. Initialize top row and leftmost column to zero.

2. Fill in the table using the following formula

$C[i-1, j-1]$	$C[i-1, j]$
$C[i, j-1]$	$C[i, j]$

$$C[i,j] = \max \begin{cases} C[i-1,j-1] + score(s[i], t[j]) \\ C[i-1,j] - \gamma \\ C[i,j-1] - \gamma \\ 0 \end{cases}$$

s

	j	1	2	3	4	5	6	7	8	9
i			C	T	G	T	T	G	C	T
1		0	0	0	0	0	0	0	0	0
2	A	0								
3	T	0								
4	G	0								
5	C	0								
6	T	0								
7	G	0								
8	C	0								
9	A	0								

t

SW example

Scoring rule: $score(s,t)$
match +1; mismatch -1; gap -2 (γ)

1. Initialize top row and leftmost column to zero.

2. Fill in the table using the following formula

$C[i\text{-}1, j\text{-}1]$	$C[i\text{-}1, j]$
$C[i, j\text{-}1]$	$C[i, j]$

$$C[i,j] = \max \begin{cases} C[i-1,j-1] + score(s[i], t[j]) & \text{0 - 1} \\ C[i-1,j] - \gamma & \text{0 - 2} \\ C[i,j-1] - \gamma & \text{0 - 2} \\ 0 & \text{0} \end{cases}$$

s

	j	1	2	3	4	5	6	7	8	9
i			C	T	G	T	T	G	C	T
1		0	0	0	0	0	0	0	0	0
2	A	0	0							
3	T	0								
4	G	0								
5	C	0								
6	T	0								
7	G	0								
8	C	0								
9	A	0								

t

SW example

Scoring rule: $score(s,t)$
match +1; mismatch -1; gap -2 (γ)

1. Initialize top row and leftmost column to zero.

2. Fill in the table using the following formula

$C[i-1, j-1]$	$C[i-1, j]$
$C[i, j-1]$	$C[i, j]$

$$C[i,j] = \max \begin{cases} C[i-1,j-1] + score(s[i], t[j]) \\ C[i-1,j] - \gamma \\ C[i,j-1] - \gamma \\ 0 \end{cases}$$

			s							
	j	1	2	3	4	5	6	7	8	9
	i		C	T	G	T	T	G	C	T
1		0	0	0	0	0	0	0	0	0
2	A	0	0	0						
3	T	0	0	1						
4	G	0								
t 5	C	0								
6	T	0								
7	G	0								
8	C	0								
9	A	0								

http://rna.informatik.uni-freiburg.de/Teaching/index.jsp?toolName=Smith-Waterman

SW example (cont.)

Question: to find the optimal local alignments between s and t.

```
s: CTGTTGCT
t: ATGCTGCA
```

	j	1	2	3	4	5	6	7	8	9
i			C	T	G	T	T	G	C	T
1		0	0	0	0	0	0	0	0	0
2	A	0	0	0	0	0	0	0	0	0
3	T	0	0	1	0	1	1	0	0	1
4	G	0	0	0	2	0	0	2	0	0
5	C	0	1	0	0	1	0	0	3	1
6	T	0	0	2	0	1	2	0	1	4
7	G	0	0	0	3	1	0	3	1	2
8	C	0	1	0	1	2	0	1	4	2
9	A	0	0	0	0	0	1	0	2	3

SW example (cont.)

Question: to find the optimal local alignments between s and t.

```
s: CTGTTGCT
t: ATGCTGCA
```

To obtain the optimum local alignment,

3. Identify the highest scores in the matrix.

4. Then, go backwards to the cell with the highest score of the positions of (i – 1, j), (i, j – 1), and (i – 1, j – 1)

5. This procedure is repeated until a cell with zero value is reached.

	j	1	2	3	4	5	6	7	8	9
i			C	T	G	T	T	G	C	T
1		0	0	0	0	0	0	0	0	0
2	A	0	0	0	0	0	0	0	0	0
3	T	0	0	1	0	1	1	0	0	1
4	G	0	0	0	2	0	0	2	0	0
5	C	0	1	0	0	1	0	0	3	1
6	T	0	0	2	0	1	2	0	1	4
7	G	0	0	0	3	1	0	3	1	2
8	C	0	1	0	1	2	0	1	4	2
9	A	0	0	0	0	0	1	0	2	3

```
s: CTGTTGCT      s: CTGTTGCT
    ||||||           ||||
t: ATGCTGCA      t: ATGCTGCA
```

Global alignment – Needleman-Wunsch

Global alignments attempt to optimally align the entire length of two sequences.

+10 for match, -2 for mismatch, -5 for gap

```
s: CTGTTGCT
t: ATGCTGCA
```

$$C[i,j] = \max \begin{cases} C[i-1,j-1] + s(s[i],t[j]) \\ C[i-1,j] - \gamma \\ C[i,j-1] - \gamma \end{cases}$$

t

s

	j	1	2	3	4	5	6	7	8	9
i			C	T	G	T	T	G	C	T
1		0	-5	-10	-15	-20	-25	-30	-35	-40
2	A	-5	-2	-7	-12	-17	-22	-27	-32	-37
3	T	-10	-7	8	3	-2	-7	-12	-17	-12
4	G	-15	-12	3	18	13	8	3	-2	-7
5	C	-20	-5	-2	13	16	11	6	13	8
6	T	-25	-10	5	0	23	26	21	16	23
7	G	-30	-15	0	15	18	21	36	31	26
8	C	-35	-20	-5	10	13	16	31	46	41
9	A	-40	-25	-10	5	8	11	26	41	44

```
s: CTG-TTGCT
     ||   |||
t: ATGC-TGCA
```

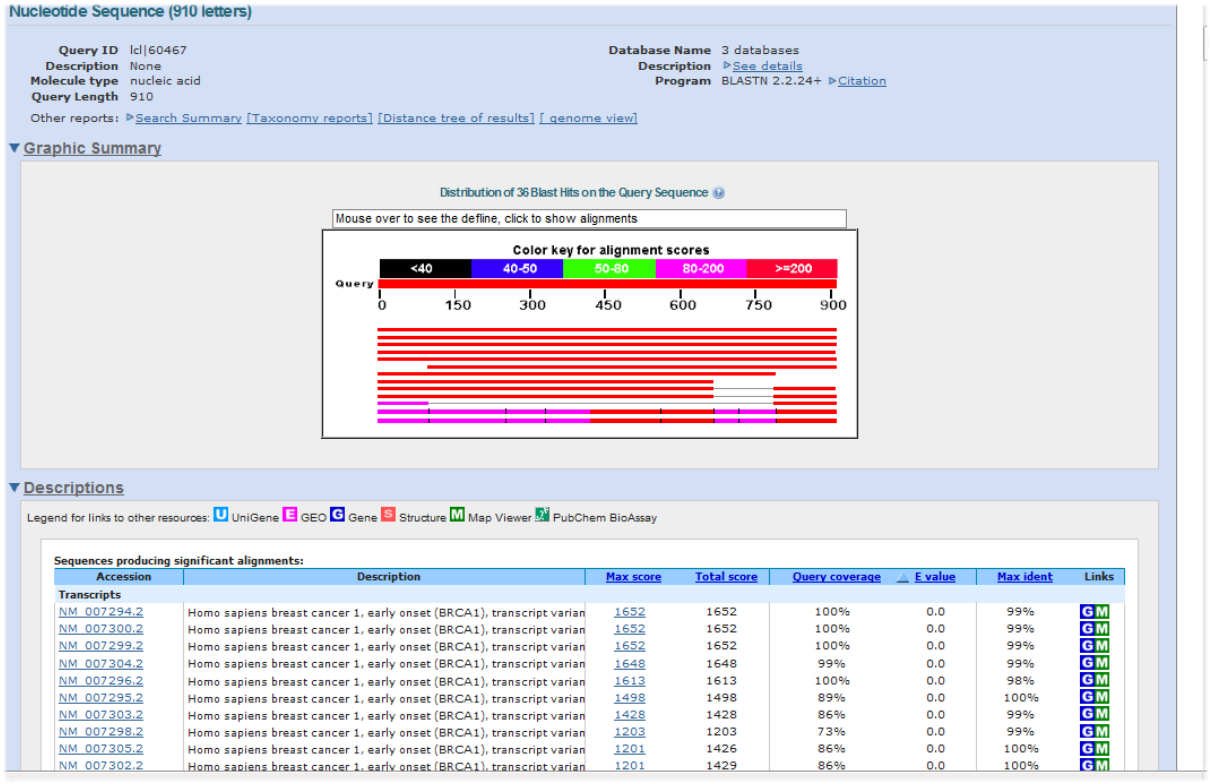
BLAST (Basic Local Alignment Search Tool)

- The classic algorithm, Smith–Waterman algorithm, optimizes the similar measure. It ensured the *best performance on accuracy* and the most precise results with respect to *its scoring scheme*.

- However, Smith–Waterman algorithm is *time-consuming* and computational burdensome. It is not practical to apply it to align a query sequence to a large database.

- *BLAST* emphasizes on speed to make the algorithm practical on huge genome databases.

Question



Could you recall the procedure of a BLAST job to achieve the BLAST alignment results? And what does NCBI actually provide?

BLAST+

- BLAST was first introduced by NCBI in 1989.
- NCBI introduced BLAST+ in 2009, which is faster and allows more flexibility in output formats and in the search input.
- It provides a variety of BLAST functions for both DNA and protein sequences.

For example:

blastp

blastp	Traditional BLASTP to compare a protein query to a protein database
blastp-short	BLASTP optimized for queries shorter than 30 residues

blastn

blastn	Traditional BLASTN requiring an exact match of 11
blastn-short	BLASTN program optimized for sequences shorter than 50 bases

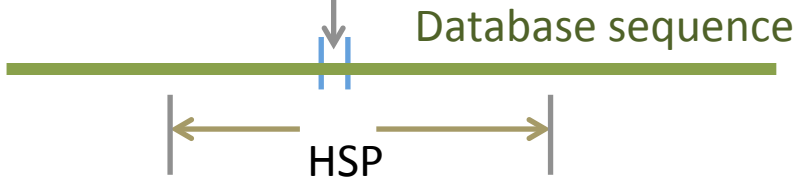
BLAST algorithm

1. Make k-tuple words (seeds) of the query sequence.

```
CTGTTGCTCGTCTCGGGACTGT
CTG
 TGT
  GTT
   …
```

2. List possible matching words for **each k-tuple word** & remove low-scoring words

```
k   mismatch
CTG 0
ATG 1
TTG 1   high-scoring words
GTG 1
CAG 1
…
AAG 2
…
```



3. Compare the high-scoring words to the database sequences to identify exact matches

4. Extend the exact matches to both directions on the database sequences to obtain high-scoring segment pairs (HSPs)

Command line based BLAST

Step 1:

- Computer/server
- Install the “BLAST+” software package
- Make databases of collected sequences

Step 2:

Run BLAST searching with your query sequences on the server

Step 1: Create a database

- **makeblastdb**

A program to create a BLAST database

```
makeblastdb –in MG1655.fasta -out MG1655 -dbtype nucl
```

Database files were generated:

```
---output---
MG1655.nhr
MG1655.nin
MG1655.nsq
...
```

Step 2: BLAST a query to a DNA database

- **blastn**

blastn -query MG1655dnaseq.fa –db MG1655

---output---

```
Query= MG1655_partial
Length=280

                                                                      Score     E
Sequences producing significant alignments:                          (Bits)  Value
gi|556503834|ref|NC_000913.3| Escherichia coli str. K-12 substr...    518   1e-147

> gi|556503834|ref|NC_000913.3| Escherichia coli str. K-12 substr.
MG1655, complete genome
Length=4641652

 Score =   518 bits (280),  Expect = 1e-147
 Identities = 280/280 (100%), Gaps = 0/280 (0%)
 Strand=Plus/Plus

Query  1      TAGAAAATGCCCATGGCAAGAATAATACCGTCCAGAGCGAAATAACCCACGTTGTGCAGG  60
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  10361  TAGAAAATGCCCATGGCAAGAATAATACCGTCCAGAGCGAAATAACCCACGTTGTGCAGG  10420

Query  61     TTAAGCAGAATGGTGGTCATGCCGAAGCCCATCAGGCCCAGCGGTGCCGGATTAGCCAAC  120
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  10421  TTAAGCAGAATGGTGGTCATGCCGAAGCCCATCAGGCCCAGCGGTGCCGGATTAGCCAAC  10480

Query  121    TTAGTGTTGCCCATAATTCCTCAAAAATCATCATCGAATGAATGGTGAAATAATTTCCCT  180
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  10481  TTAGTGTTGCCCATAATTCCTCAAAAATCATCATCGAATGAATGGTGAAATAATTTCCCT  10540

Query  181    GAATAACTGTAGTGTTTTCAGGGCGCGGCATAATAATCAGCCAGTGGGGCAGTGTCTACG  240
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  10541  GAATAACTGTAGTGTTTTCAGGGCGCGGCATAATAATCAGCCAGTGGGGCAGTGTCTACG  10600

Query  241    ATCTTTTGAGGGGAAAATGAAAATTTTCCCCGGTTTCCGG  280
              |||||||||||||||||||||||||||||||||||||||||
Sbjct  10601  ATCTTTTGAGGGGAAAATGAAAATTTTCCCCGGTTTCCGG  10640
```

Select output format

```
blastn -query MG1655dnaseq.fa -db MG1655 -outfmt 6
```

query id	subject id	% identity	alignment length	mismatches	gap opens	q. start	q. end	s. start	s. end	evalue	bit score
MG1655_partial	gi|556503834|ref|NC_000913.3|	100	280	0	0	1	280	10361	10640	1.00E-147	518

E-value

```
Score =   518 bits (280),  Expect = 1e-147
Identities = 280/280 (100%), Gaps = 0/280 (0%)
Strand=Plus/Plus

Query  1      TAGAAAATGCCCATGGCAAGAATAATACCGTCCAGAGCGAAATAACCCACGTTGTGCAGG  60
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  10361  TAGAAAATGCCCATGGCAAGAATAATACCGTCCAGAGCGAAATAACCCACGTTGTGCAGG  10420

…

Query  241    ATCTTTTGAGGGGAAAATGAAAATTTTCCCCGGTTTCCGG  280
              ||||||||||||||||||||||||||||||||||||||||
Sbjct  10601  ATCTTTTGAGGGGAAAATGAAAATTTTCCCCGGTTTCCGG  10640
```

- **E-value** is a parameter that describes the number of hits that one can "expect" to see by chance when searching a database of a particular size. It is used to describe the significance (instead of a p-value) of each sequence alignment hit.

For example, E-value = 1 means that in a database of the similar size 1 match with a similar score would be obtained simply by chance.

- The lower the E-value, the more "significant" the match is.

Score and Bit scores

```
Score =   518 bits (280),  Expect = 1e-147
Identities = 280/280 (100%), Gaps = 0/280 (0%)
Strand=Plus/Plus

Query  1      TAGAAAATGCCCATGGCAAGAATAATACCGTCCAGAGCGAAATAACCCACGTTGTGCAGG  60
              ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
Sbjct  10361  TAGAAAATGCCCATGGCAAGAATAATACCGTCCAGAGCGAAATAACCCACGTTGTGCAGG  10420

…

Query  241    ATCTTTTGAGGGGAAAATGAAAATTTTCCCCGGTTTCCGG  280
              ||||||||||||||||||||||||||||||||||||||||
Sbjct  10601  ATCTTTTGAGGGGAAAATGAAAATTTTCCCCGGTTTCCGG  10640
```

- In the context of sequence alignments, a score is a numerical value that describes the overall quality of an alignment.
- The **bit-score** is a rescaled alignment score to indicate the alignment quality, which is independent of the size of the search database.
- The higher the score/bit-score, the better alignment is.

Extract sequences or subsequences

- **blastdbcmd**

Extract sequences from the database

```
# Use Gi ID to search*

blastdbcmd –db MG1655 -entry 556503834 -range 150-220


---output---
>gi|556503834|ref|NC_000913.3|:150-220 Escherichia coli str. K-12 substr. MG1655, complete genome
AGCGCACAGACAGATAAAAATTACAGAGTACACAACATCCATGAAACGCATTAGCACCACCATTAC
CACCA
```

* Database formatting needs to be a little different:

```
makeblastdb –in MG1655.fasta -out MG1655 -dbtype nucl –parse_seqids
```

BLAST tools

Table 1. Key features of the BLAST search pages in the "Basic BLAST" category

Search page	Query & database combination	Alignment type	Programs & functions (default program in bold)
nucleotide blast	nucleotide vs nucleotide	nucleotide vs nucleotide	**megablast**: for sequence identification, intra-species comparison discontiguous megablast: for cross-species comparison, searching with coding sequences blastn: for searching with shorter queries, cross-species comparison
protein blast	Protein vs protein	protein vs protein	**blastp**: general sequence identification and similarity searches DELTA-BLAST [2] : protein similarity search with higher sensitivity than blastp PSI-BLAST: iterative search for position-specific score matrix (PSSM) construction or identification of distant relatives for a protein family PHI-BLAST: protein alignment with input pattern as anchor/constraint
blastx	nucleotide (translated) vs protein	protein vs protein	blastx: for identifying potential protein products encoded by a nucleotide query
tblastn	protein vs nucleotide (translated)	protein vs protein	tblastn: for identifying database sequences encoding proteins similar to the query
tblastx	nucleotide (translated) vs nucleotide (translated)	protein vs protein	tblastx: for identifying nucleotide sequences similar to the query based on their coding potential

DNA nucleotide database

Table 2. Contents of the common BLAST sequence databases

Database	Type	Content
nr (nt) default	Nucleotide	All GenBank + EMBL + DDBJ + PDB sequences, excluding sequences from PAT, EST, STS, GSS, WGS, TSA and phase 0, 1 or 2 HTGS sequences, partially non-redundant.
refseq_rna	Nucleotide	Curated (NM_, NR_) plus predicted (XM_, XR_) sequences from NCBI Reference Sequence Project.
refseq_genomic	Nucleotide	Genomic sequences from NCBI Reference Sequence Project.
chromosome	Nucleotide	Complete genomes and complete chromosomes from the NCBI Reference Sequence project.
Human G+T	Nucleotide	The genomic sequences plus curated and predicted RNAs from the current build of the human genome.
Mouse G+T	Nucleotide	The genomic sequences plus curated and predicted RNAs from the current build of the mouse genome.
est	Nucleotide	Database of GenBank + EMBL + DDBJ sequences from EST division
HTGS	Nucleotide	Unfinished High Throughput Genomic Sequences; Sequences: phases 0, 1 and 2
wgs	Nucleotide	Assemblies of Whole Genome Shotgun sequences.
pat	Nucleotide	Nucleotides from the Patent division of GenBank.
pdb	Nucleotide	Nucleotide sequences from the 3-dimensional structure records from Protein Data Bank.
alu_repeats	Nucleotide	Selected Alu repeats from REPBASE, suitable for identifying Alu repeats from query sequences. See "Alu alert" by Claverie and Makalowski, Nature 371: 752 (1994).
TSA	Nucleotide	Transcriptome Shotgun Assemblies, assembled from RNA-seq SRA data
16S microbial	Nucleotide	16S Microbial rRNA sequences from Targeted Loci Project

Protein database

nr **default**	Protein	Non-redundant GenBank CDS translations + RefSeq + PDB + SwissProt + PIR + PRF, excluding those in PAT, TSA, and env_nr.
refseq_protein	Protein	Protein sequences from NCBI Reference Sequence project.
swissprot	Protein	Last major release of the UniProtKB/SWISS-PROT protein sequence database (no incremental updates).
pat	Protein	Proteins from the Patent division of GenBank.
pdb	Protein	Protein sequences from the 3-dimensional structure records from the Protein Data Bank.
env_nr	Protein	Protein sequences translated from the CDS annotation of metagenomic nucleotide sequences.
tsa_nr	Protein	Protein sequences translated from CDSs annotated on transcriptome shotgun assemblies.

ftp://ftp.ncbi.nlm.nih.gov/pub/factsheets/HowTo_BLASTGuide.pdf