DNA Sequencing and Data Analysis

Prof Noam Shomron
Hadas Volkov

Lecture 6, May 16, 2024

DNA Sequencing and Data Analysis

Sequence Mapping and Alignment

Thursday 18:30 to 21:00

Hangar H2

nshomron@gmail.com

hadas.volkov@post.runi.ac.il

Why Do We Need Sequence Mapping?

Determine the origin of an unknown sequence

Find homologous sequences

Determine genomic position of a sequence

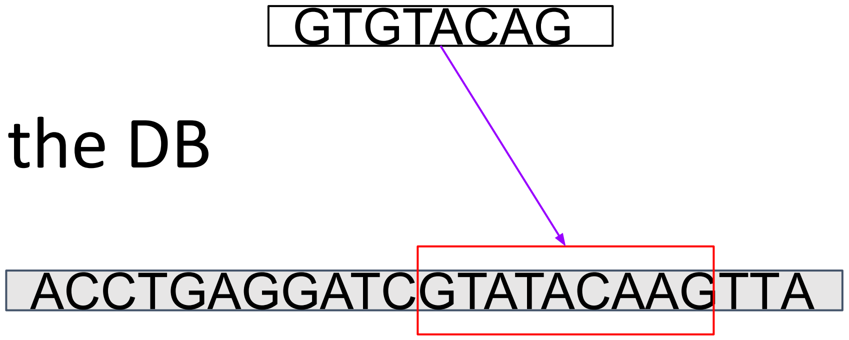
Identify genomic variants between samples (variant calling)

Determine the function of a sequence (annotation)

Two Stages of Sequence Mapping

1. **SEARCH** -

   Roughly find the position of the query in the DB



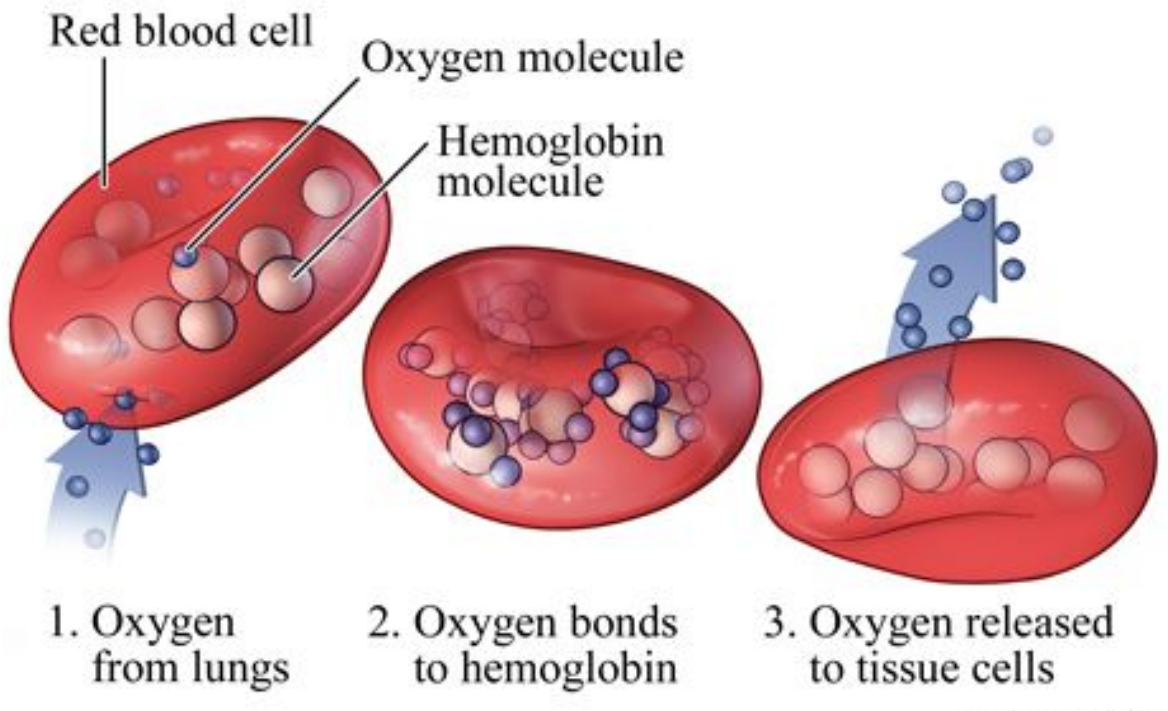
2. **ALIGN** -

   Find the exact pairwise alignment of the query and the DB sequences

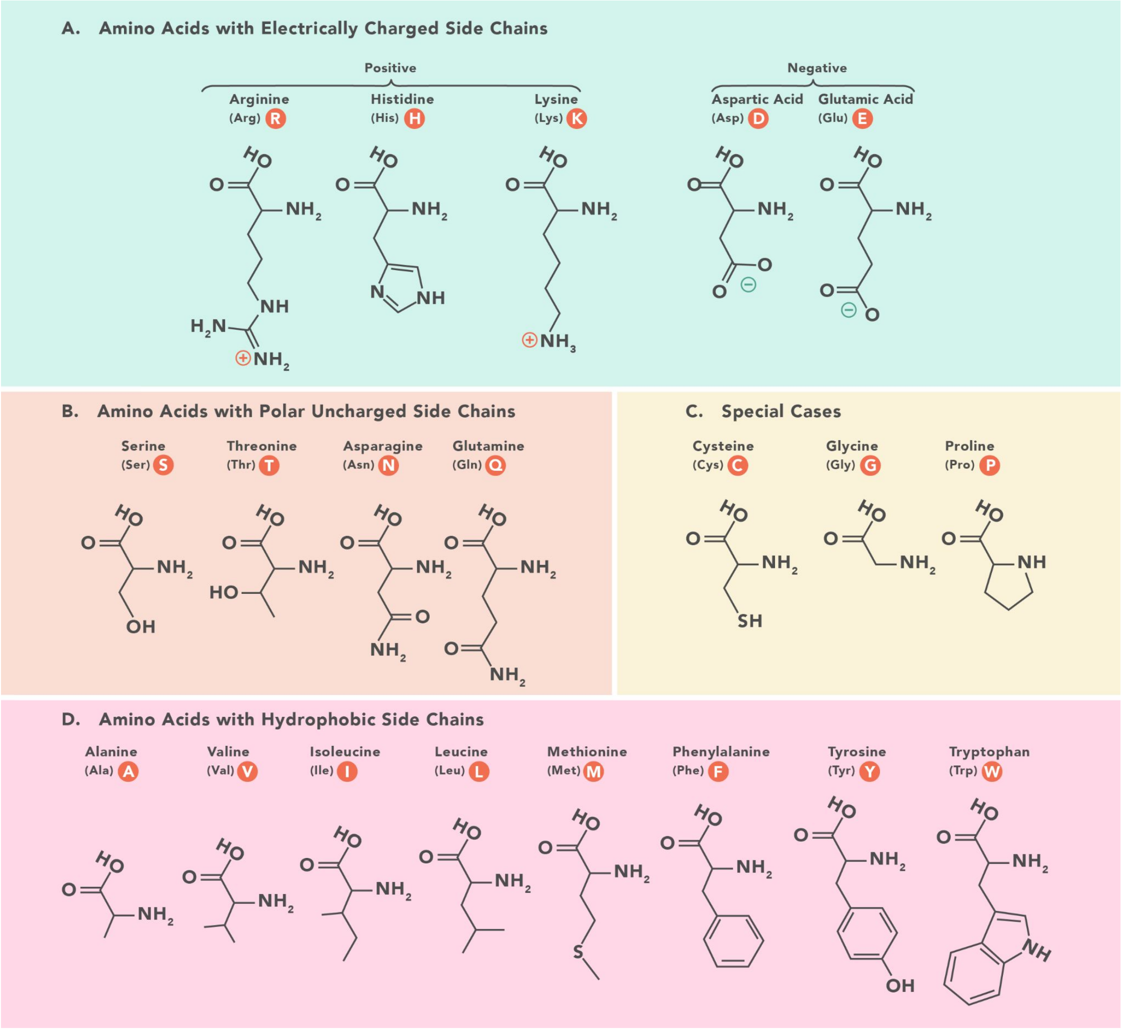
G	T	G	T	A	C	A	-	G
G	T	A	T	A	C	A	A	G

Pairwise Alignment

Hemoglobin Homologous

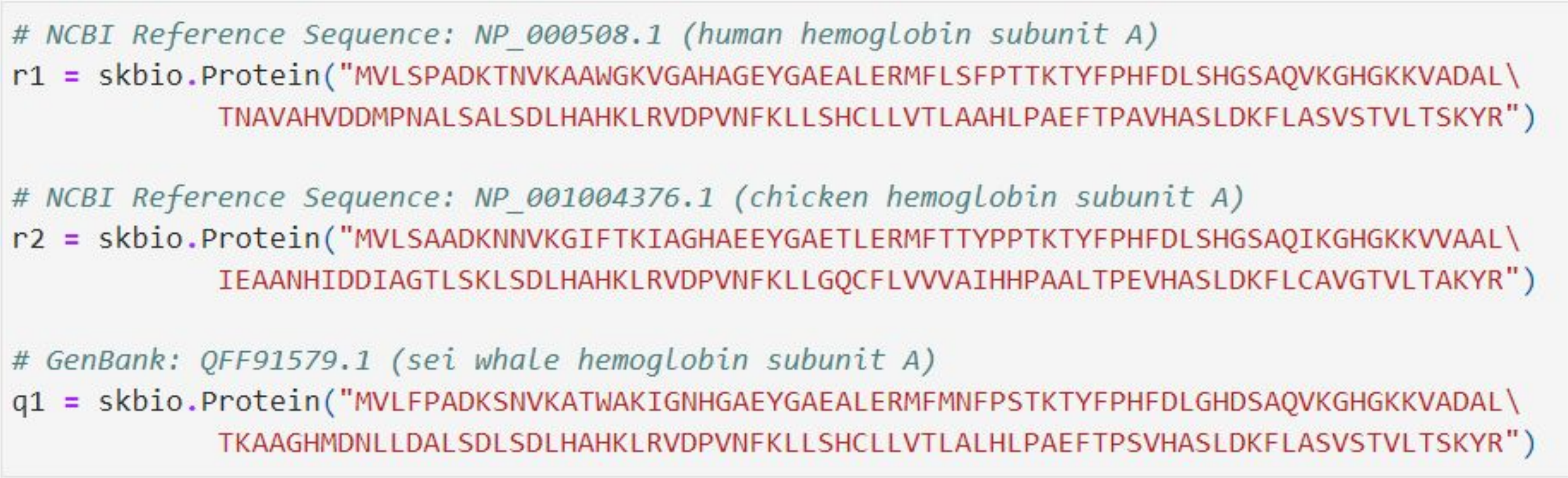


© 2016 Healthwise



Pairwise Alignment

Hemoglobin Homologous

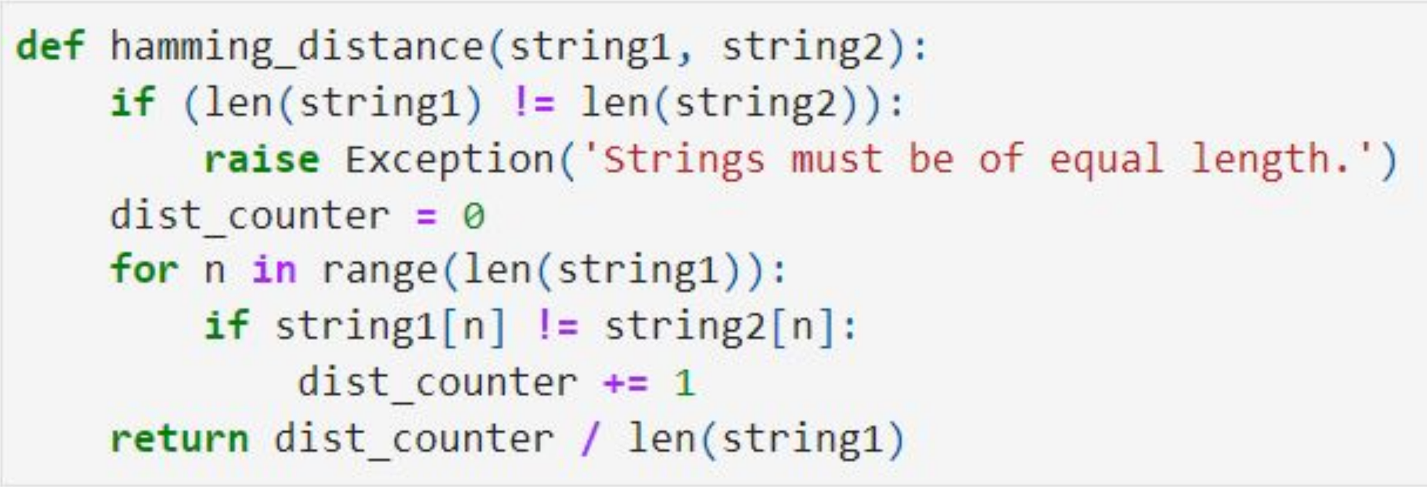


```
# NCBI Reference Sequence: NP_000508.1 (human hemoglobin subunit A)
r1 = skbio.Protein("MVLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHGKKVADAL\
                   TNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR")

# NCBI Reference Sequence: NP_001004376.1 (chicken hemoglobin subunit A)
r2 = skbio.Protein("MVLSAADKNNVKGIFTKIAGHAEEYGAETLERMFTTYPPTKTYFPHFDLSHGSAQIKGHGKKVVAAL\
                   IEAANHIDDIAGTLSKLSDLHAHKLRVDPVNFKLLGQCFLVVVAIHHPAALTPEVHASLDKFLCAVGTVLTAKYR")

# GenBank: QFF91579.1 (sei whale hemoglobin subunit A)
q1 = skbio.Protein("MVLFPADKSNVKATWAKIGNHGAEYGAEALERMFMNFPSTKTYFPHFDLGHDSAQVKGHGKKVADAL\
                   TKAAGHMDNLLDALSDLSDLHAHKLRVDPVNFKLLSHCLLVTLALHLPAEFTPSVHASLDKFLASVSTVLTSKYR")
```

Pairwise Alignment

The Hamming Distance

Hamming distance is the number of symbols or positions of two strings at which their corresponding characters are different

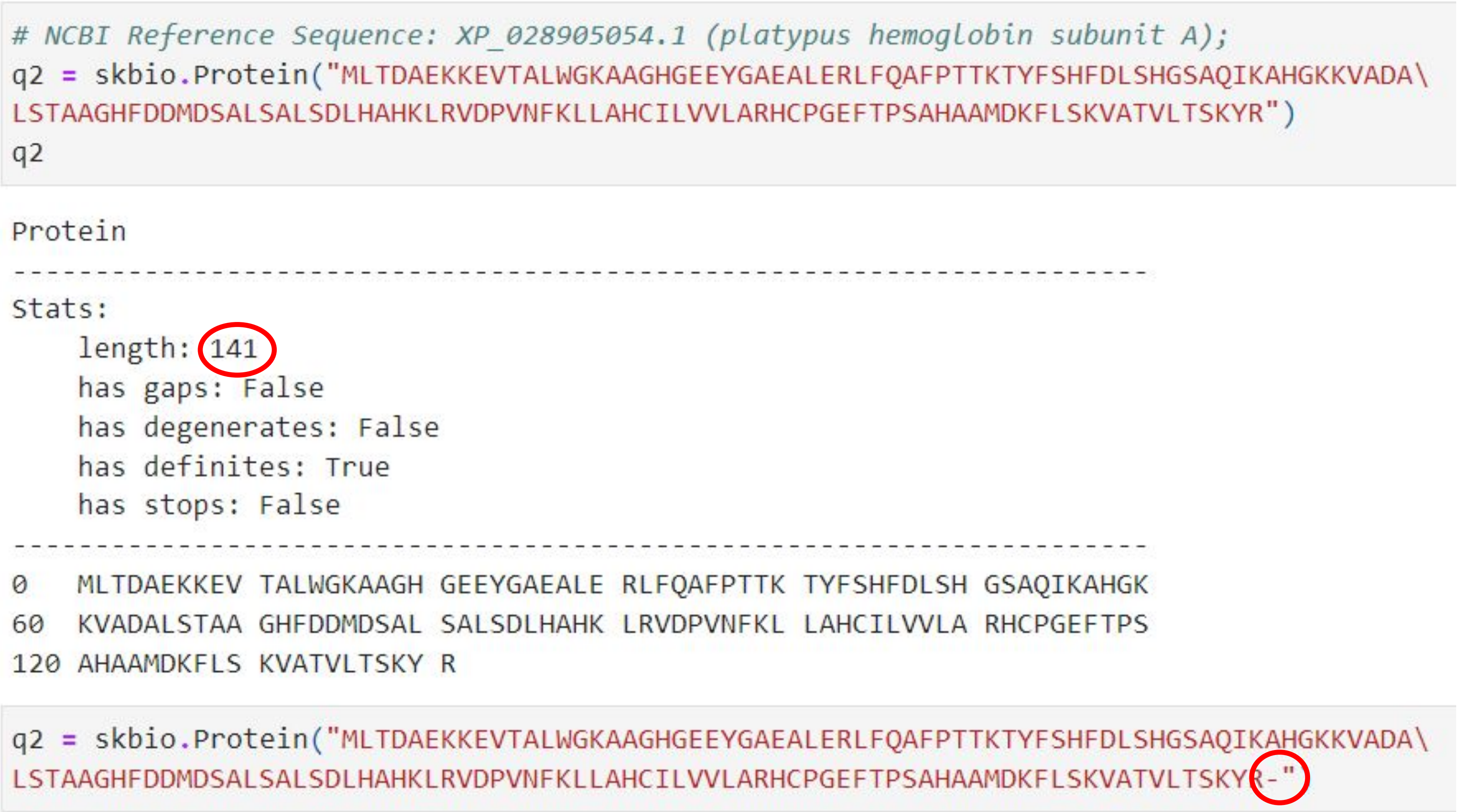


```
def hamming_distance(string1, string2):
    if (len(string1) != len(string2)):
        raise Exception('Strings must be of equal length.')
    dist_counter = 0
    for n in range(len(string1)):
        if string1[n] != string2[n]:
            dist_counter += 1
    return dist_counter / len(string1)
```

The Hamming distance between r1 and q1 is: 0.1690
The Hamming distance between r2 and q1 is: 0.3169

Pairwise Alignment

The Hamming Distance



```
# NCBI Reference Sequence: XP_028905054.1 (platypus hemoglobin subunit A);
q2 = skbio.Protein("MLTDAEKKEVTALWGKAAGHGEEYGAEALERLFQAFPTTKTYFSHFDLSHGSAQIKAHGKKVADA\
LSTAAGHFDDMDSALSALSDLHAHKLRVDPVNFKLLAHCILVVLARHCPGEFTPSAHAAMDKFLSKVATVLTSKYR")
q2
```

```
Protein
---------------------------------------------------------------------
Stats:
    length: 141
    has gaps: False
    has degenerates: False
    has definites: True
    has stops: False
---------------------------------------------------------------------
0   MLTDAEKKEV TALWGKAAGH GEEYGAEALE RLFQAFPTTK TYFSHFDLSH GSAQIKAHGK
60  KVADALSTAA GHFDDMDSAL SALSDLHAHK LRVDPVNFKL LAHCILVVLA RHCPGEFTPS
120 AHAAMDKFLS KVATVLTSKY R
```

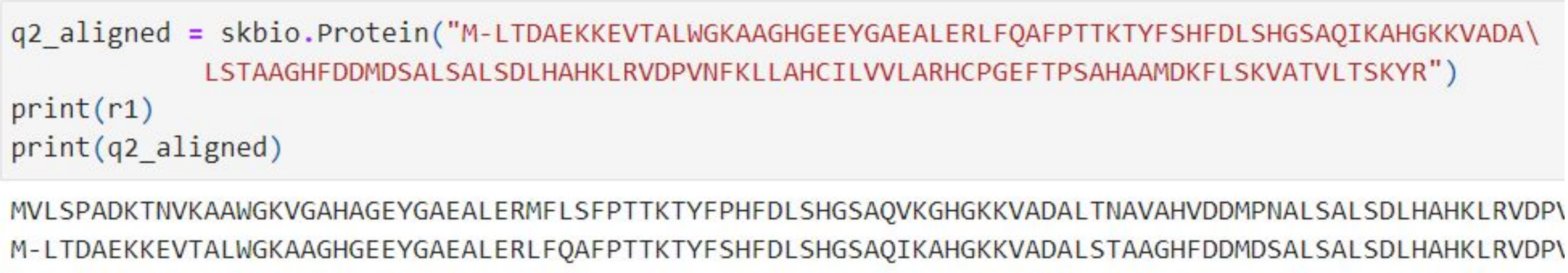
```
q2 = skbio.Protein("MLTDAEKKEVTALWGKAAGHGEEYGAEALERLFQAFPTTKTYFSHFDLSHGSAQIKAHGKKVADA\
LSTAAGHFDDMDSALSALSDLHAHKLRVDPVNFKLLAHCILVVLARHCPGEFTPSAHAAMDKFLSKVATVLTSKYR-")
```

Pairwise Alignment

The Hamming Distance

The Hamming distance between r1 and q2 is: 0.90845

The Hamming distance between r2 and q2 is: 0.92254



```python
q2_aligned = skbio.Protein("M-LTDAEKKEVTALWGKAAGHGEEYGAEALERLFQAFPTTKTYFSHFDLSHGSAQIKAHGKKVADA\
                LSTAAGHFDDMDSALSALSDLHAHKLRVDPVNFKLLAHCILVVLARHCPGEFTPSAHAAMDKFLSKVATVLTSKYR")
print(r1)
print(q2_aligned)
```

```
MVLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHGKKVADALTNAVAHVDDMPNALSALSDLHAHKLRVDP
M-LTDAEKKEVTALWGKAAGHGEEYGAEALERLFQAFPTTKTYFSHFDLSHGSAQIKAHGKKVADALSTAAGHFDDMDSALSALSDLHAHKLRVDP
```

The Hamming distance between r1 and q2_aligned is: 0.27465

The Hamming distance between r2 and q2_aligned is: 0.34507

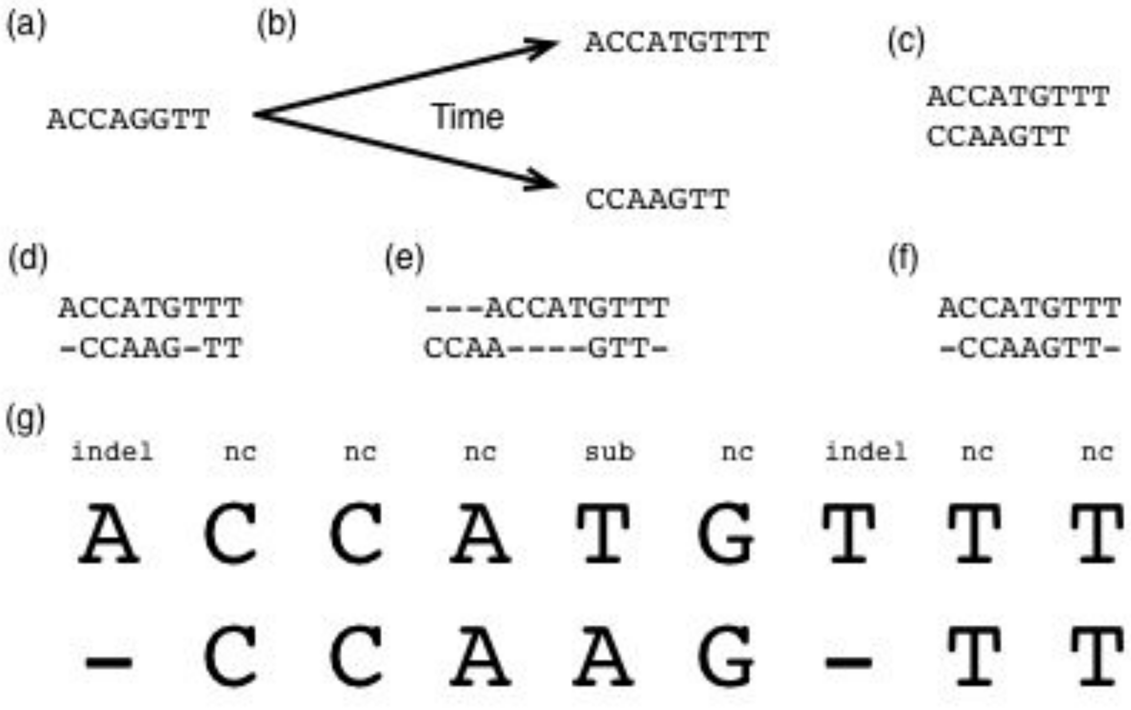
What Is Sequence Alignment?

Mutations:

Substitutions, where one DNA base is replaced with another

Insertions, where one or more contiguous DNA bases are inserted into a sequence

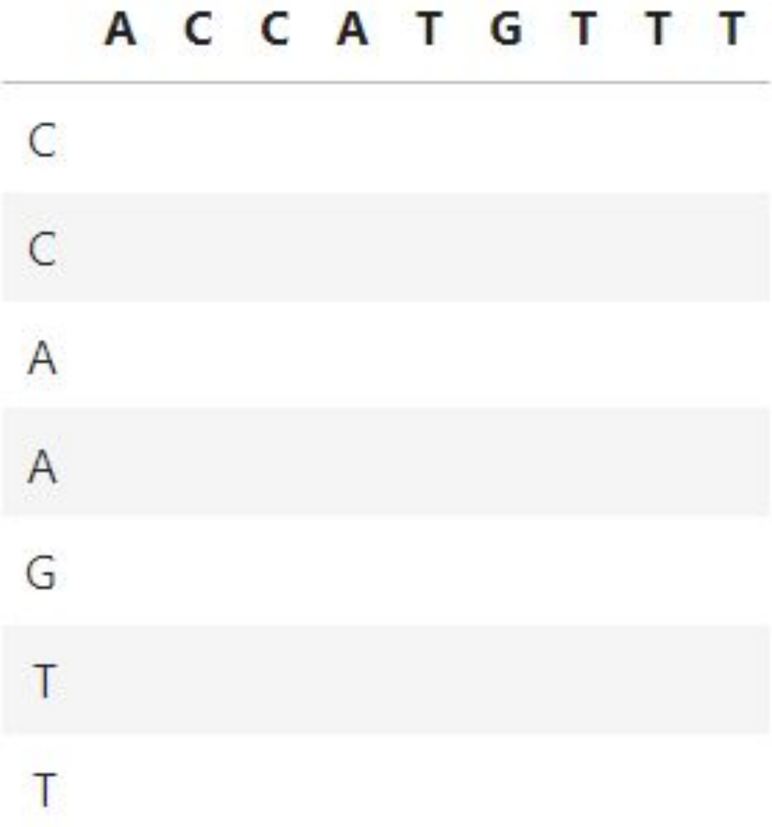
Deletions, where one or more contiguous DNA bases are deleted from a sequence.



Simple Align

ACCATGTTT

CCAAGTT

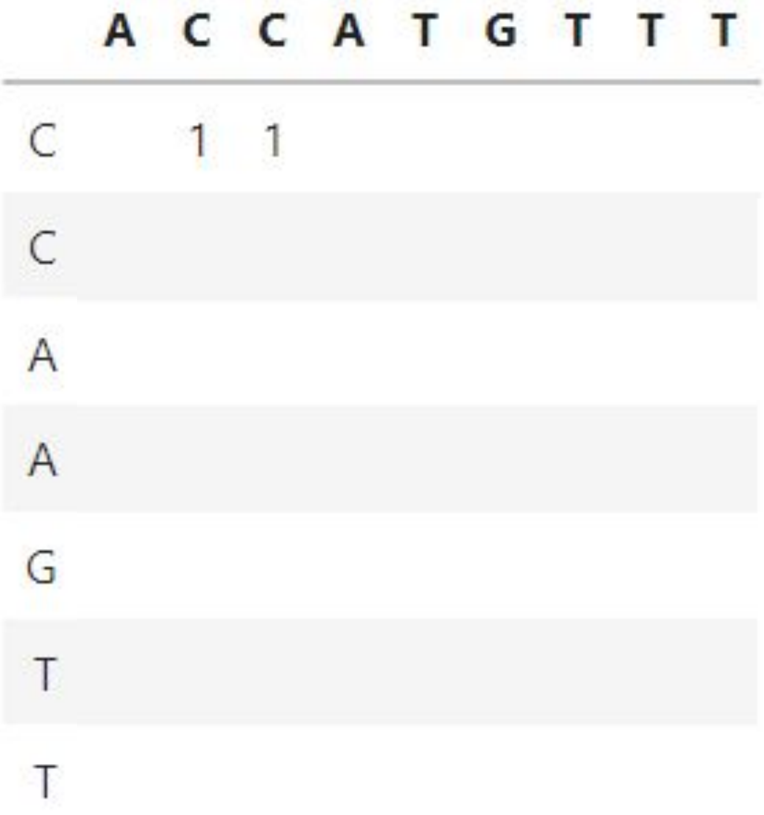


	A	C	C	A	T	G	T	T	T
C									
C									
A									
A									
G									
T									
T									

Simple Align

ACCATGTTT

CCAAGTT

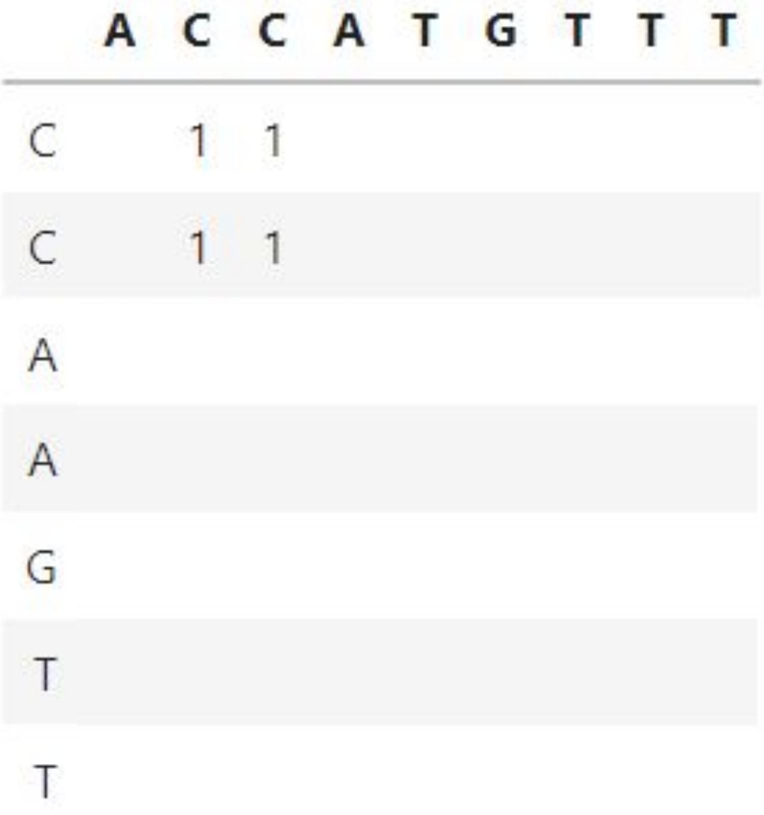


	A	C	C	A	T	G	T	T	T
C		1	1						
C									
A									
A									
G									
T									
T									

Simple Align

ACCATGTTT

CCAAGTT

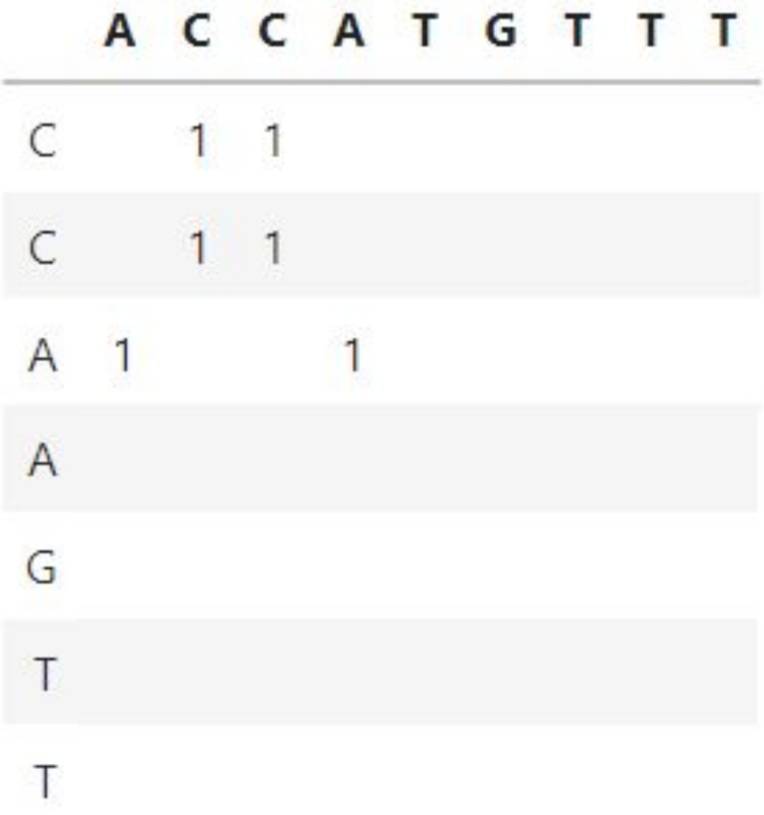


	A	C	C	A	T	G	T	T	T
C		1	1						
C		1	1						
A									
A									
G									
T									
T									

Simple Align

ACCATGTTT

CCAAGTT

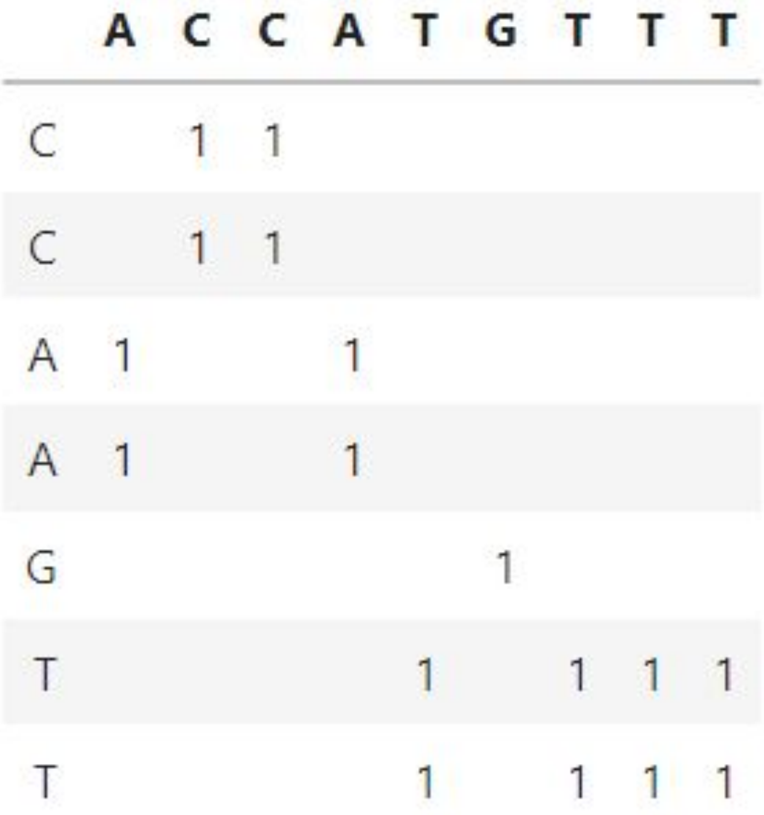


	A	C	C	A	T	G	T	T	T
C		1	1						
C		1	1						
A	1			1					
A									
G									
T									
T									

Simple Align

ACCATGTTT

CCAAGTT

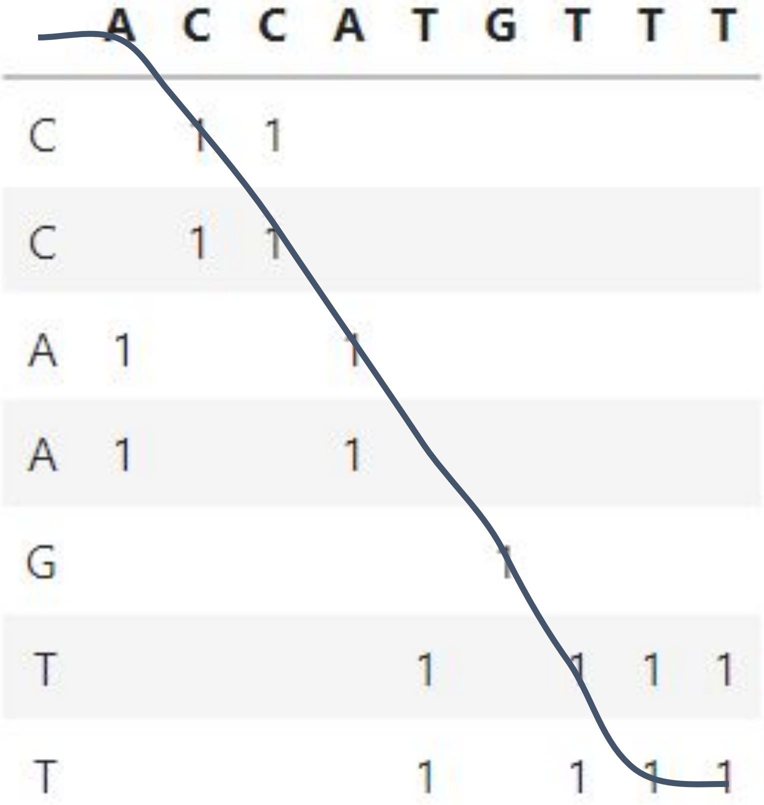


	A	C	C	A	T	G	T	T	T
C		1	1						
C		1	1						
A	1			1					
A	1			1					
G						1			
T					1		1	1	1
T					1		1	1	1

Simple Align

ACCATGTTT

CCAAGTT



	A	C	C	A	T	G	T	T	T
C		1	1						
C		1	1						
A	1			1					
A	1			1					
G						1			
T					1		1	1	1
T					1		1	1	1

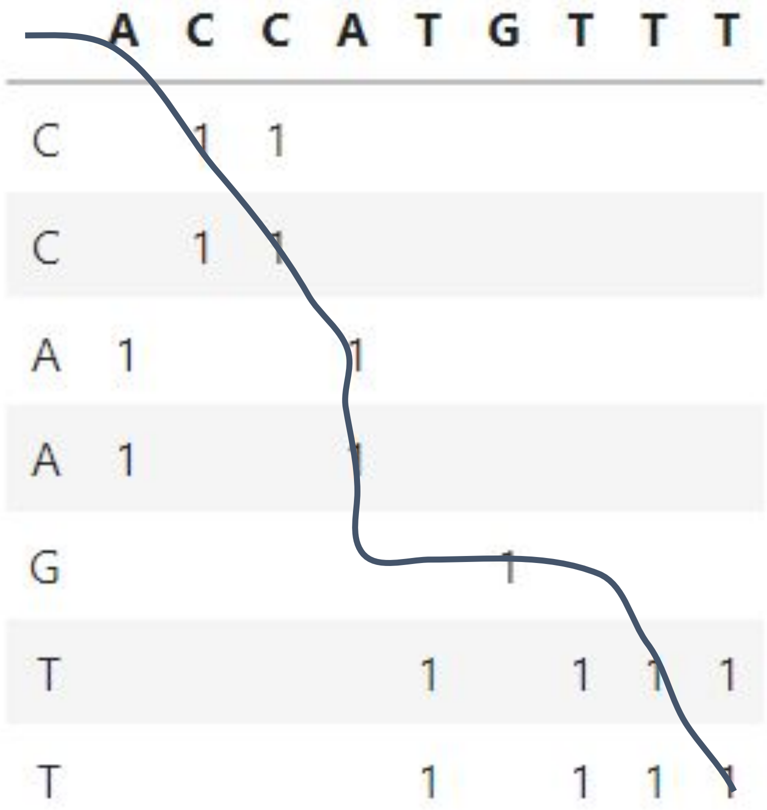
ACCATGTTT

-CCAAGTT -

Simple Align

ACCATGTTT

CCAAGTT



	A	C	C	A	T	G	T	T	T
C		1	1						
C		1	1						
A	1			1					
A	1			1					
G						1			
T					1		1	1	1
T					1		1	1	1

ACCA- -TGTTT

-CCAAG- - -TT

Simple Align

ACCATGTTT

CCAAGTT

ACCATGTTT

-CCAAGTT -

S = -1+1+1-1+1+1+1-1=4

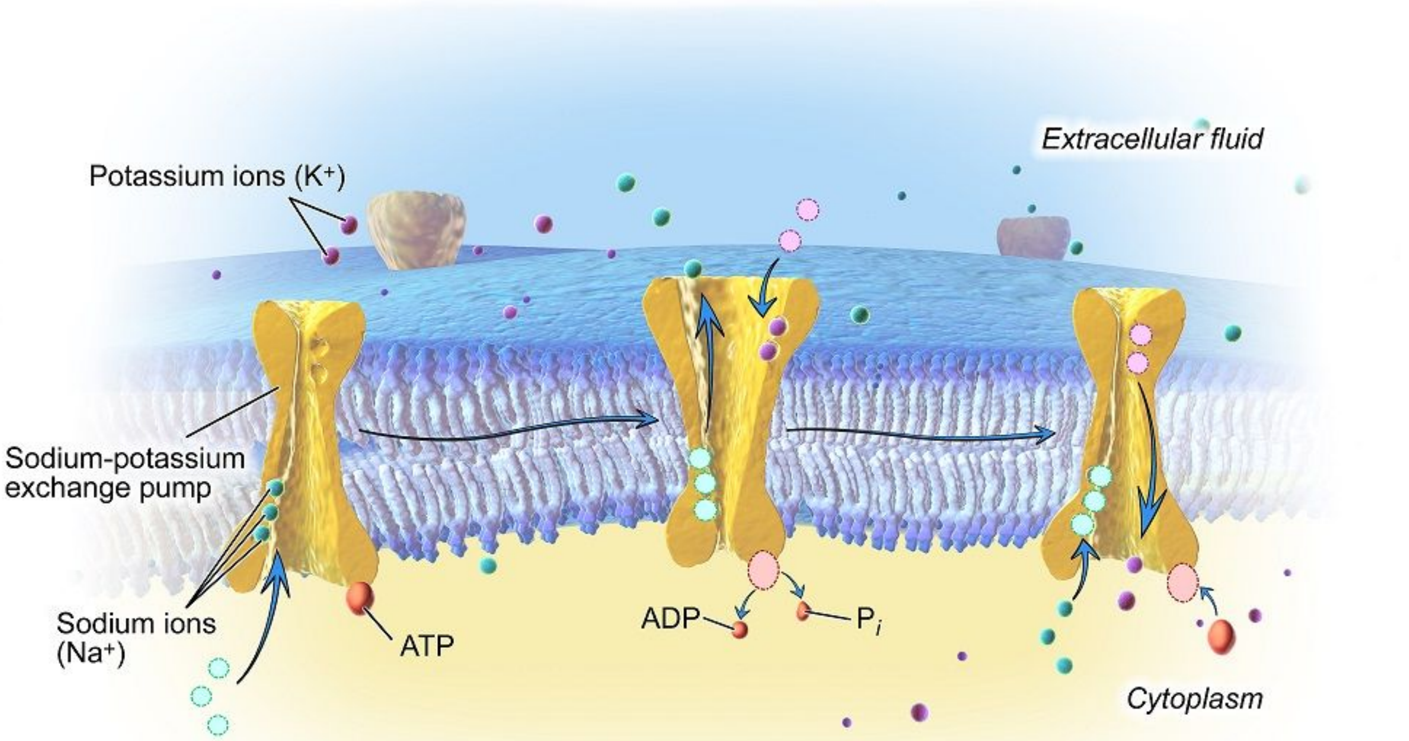
ACCA- -TGTTT

-CCAAG- - -TT

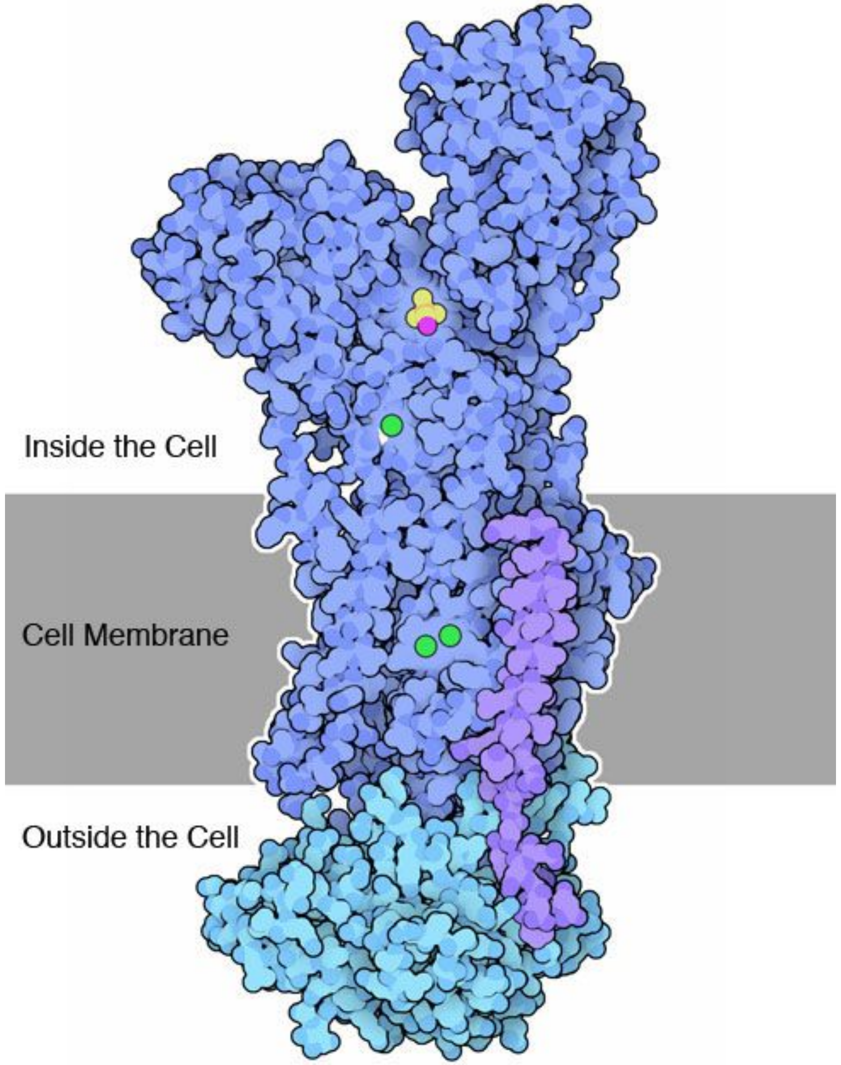
S = -1+1+1+1-1-1-1-1-1+1+1=-1

Simple Align

Too Simplistic



The Sodium-Potassium Exchange Pump

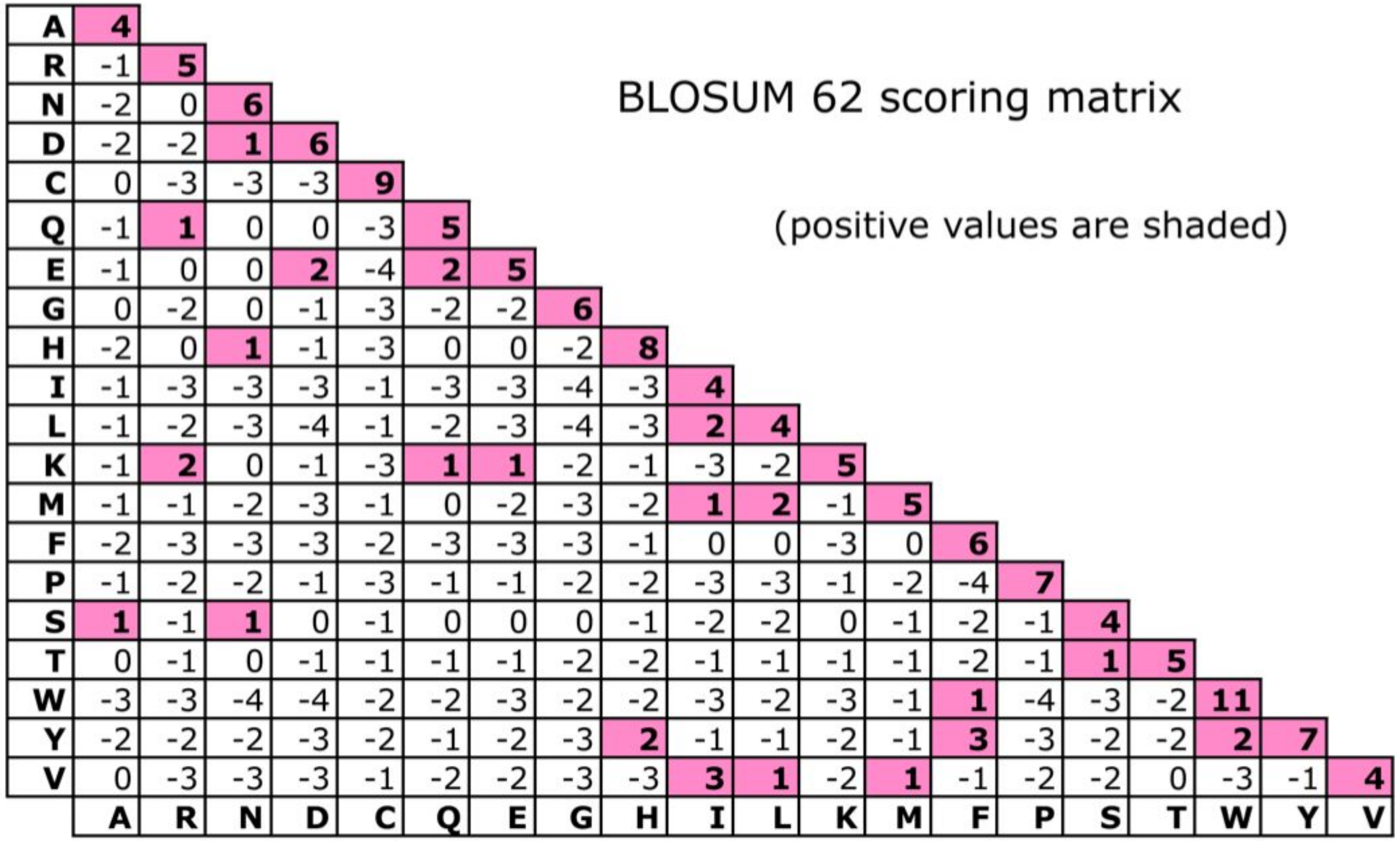


Simple Align

Too Simplistic

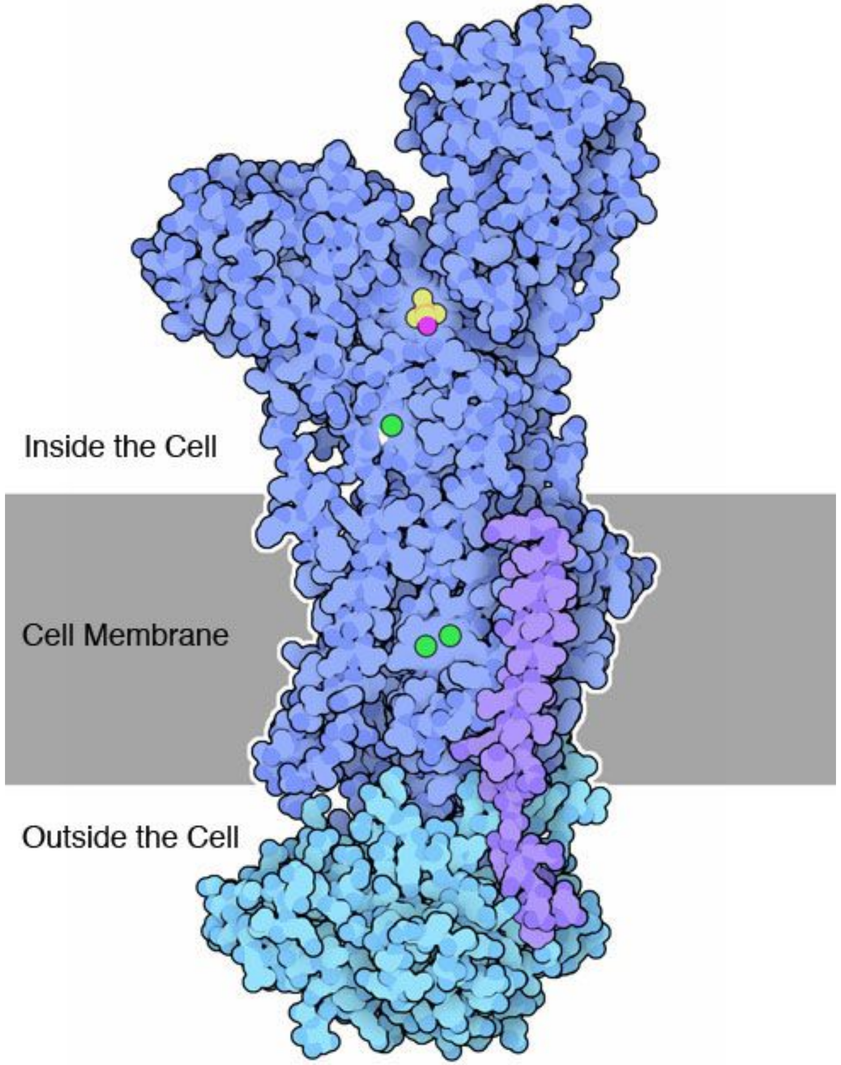
BLOSUM 62 scoring matrix

(positive values are shaded)

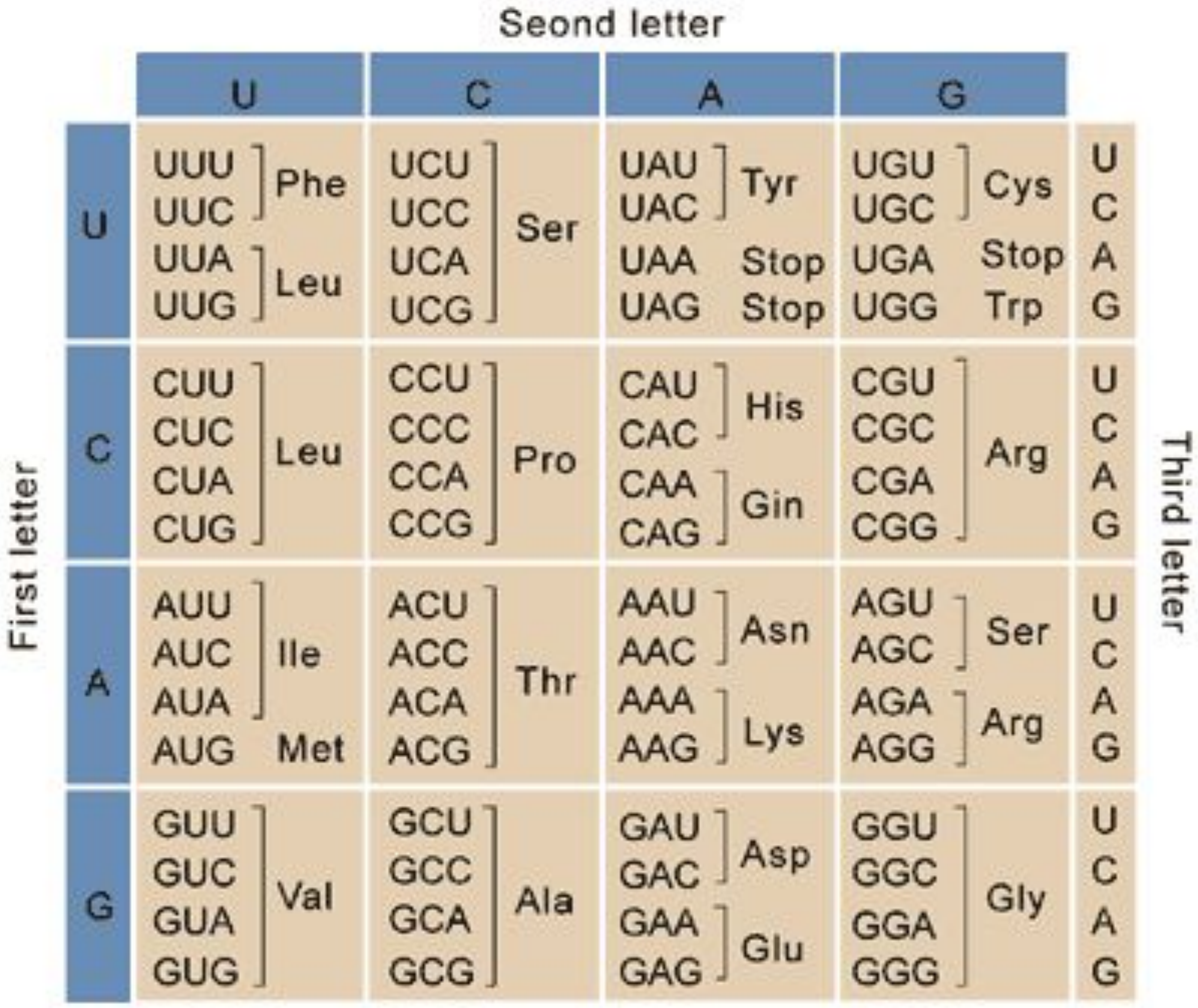


A	**4**																			
R	-1	**5**																		
N	-2	0	**6**																	
D	-2	-2	**1**	**6**																
C	0	-3	-3	-3	**9**															
Q	-1	**1**	0	0	-3	**5**														
E	-1	0	0	**2**	-4	**2**	**5**													
G	0	-2	0	-1	-3	-2	-2	**6**												
H	-2	0	**1**	-1	-3	0	0	-2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

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.

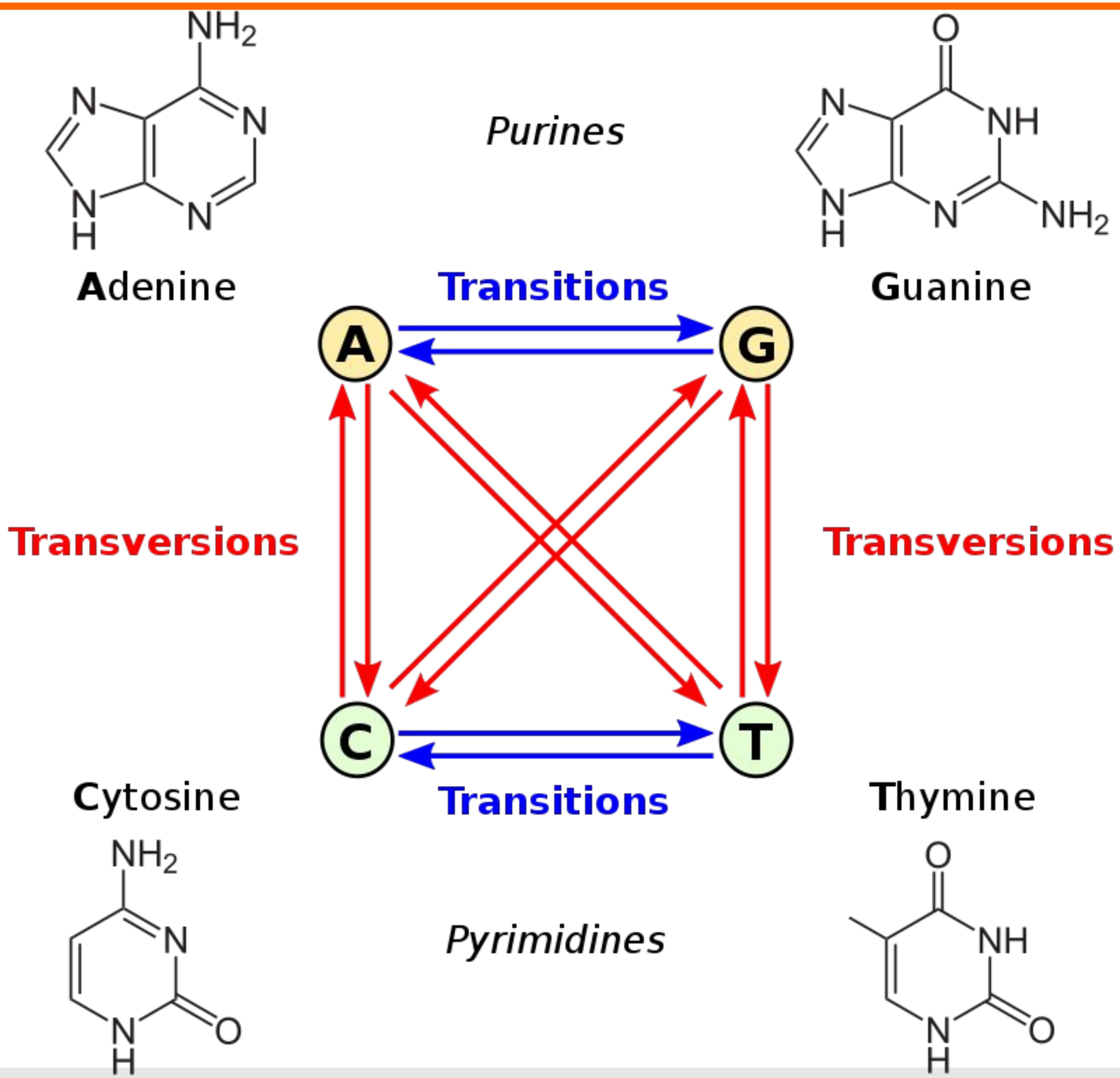


Simple Align



	Seond letter								
First letter	U		C		A		G		Third letter
U	UUU	Phe	UCU	Ser	UAU	Tyr	UGU	Cys	U
	UUC		UCC		UAC		UGC		C
	UUA	Leu	UCA		UAA	Stop	UGA	Stop	A
	UUG		UCG		UAG	Stop	UGG	Trp	G
C	CUU	Leu	CCU	Pro	CAU	His	CGU	Arg	U
	CUC		CCC		CAC		CGC		C
	CUA		CCA		CAA	Gln	CGA		A
	CUG		CCG		CAG		CGG		G
A	AUU	Ile	ACU	Thr	AAU	Asn	AGU	Ser	U
	AUC		ACC		AAC		AGC		C
	AUA		ACA		AAA	Lys	AGA	Arg	A
	AUG	Met	ACG		AAG		AGG		G
G	GUU	Val	GCU	Ala	GAU	Asp	GGU	Gly	U
	GUC		GCC		GAC		GGC		C
	GUA		GCA		GAA	Glu	GGA		A
	GUG		GCG		GAG		GGG		G

Simple Align



Adenine

Purines

Guanine

Transitions

Transversions

Transversions

Transitions

Cytosine

Pyrimidines

Thymine

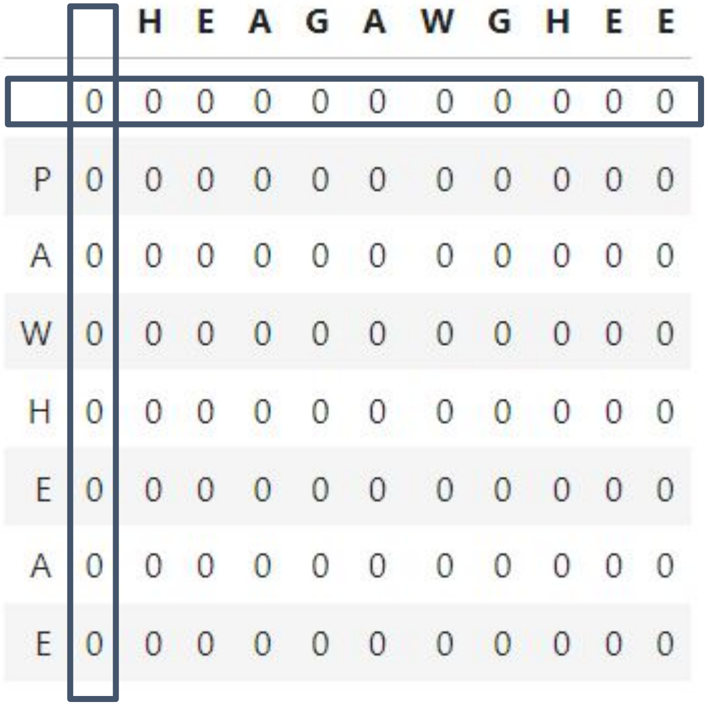
Needleman-Wunsch Alignment

HEAGAWGHEE

PAWHEAE

F - The Dynamic Programming Matrix

T - The Traceback Matrix

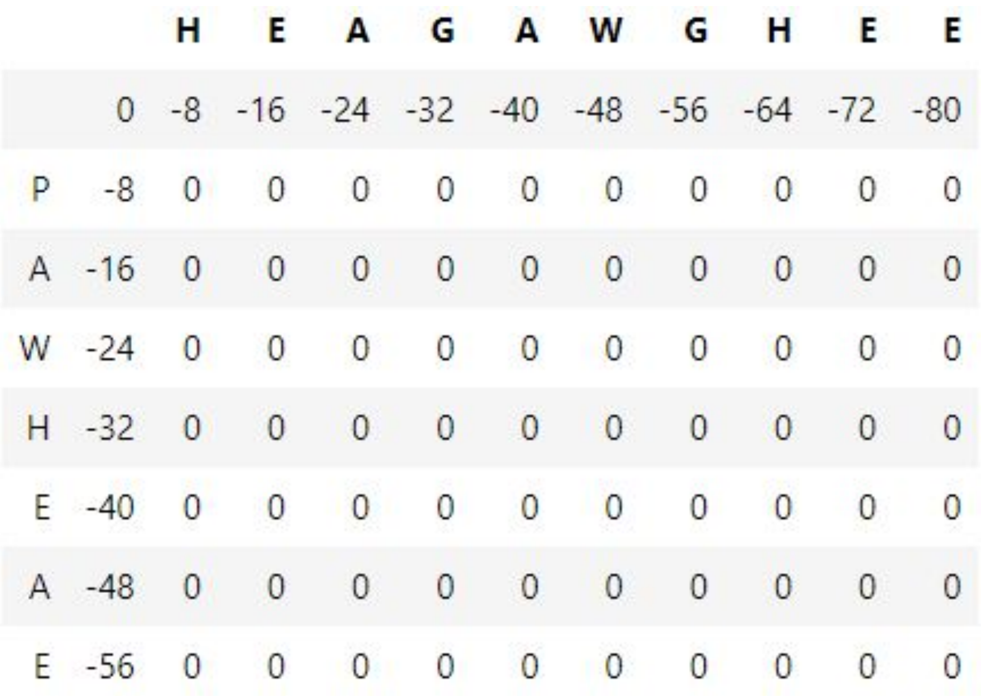


		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0	0

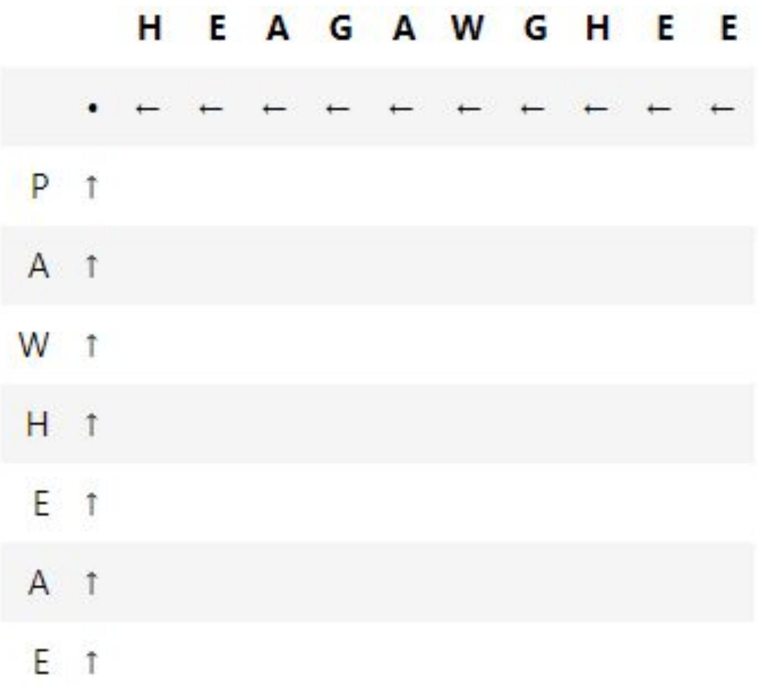
Needleman-Wunsch Alignment

$$F(0,0)=0$$
$$F(i,0)=F(i-1,0)-d$$
$$F(0,j)=F(0,j-1)-d$$

$$T(0,0)=.$$
$$T(i,0)=\leftarrow$$
$$T(0,j)=\uparrow$$



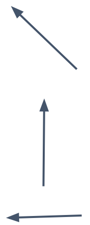
		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	0	0	0	0	0	0	0	0	0	0
A	-16	0	0	0	0	0	0	0	0	0	0
W	-24	0	0	0	0	0	0	0	0	0	0
H	-32	0	0	0	0	0	0	0	0	0	0
E	-40	0	0	0	0	0	0	0	0	0	0
A	-48	0	0	0	0	0	0	0	0	0	0
E	-56	0	0	0	0	0	0	0	0	0	0

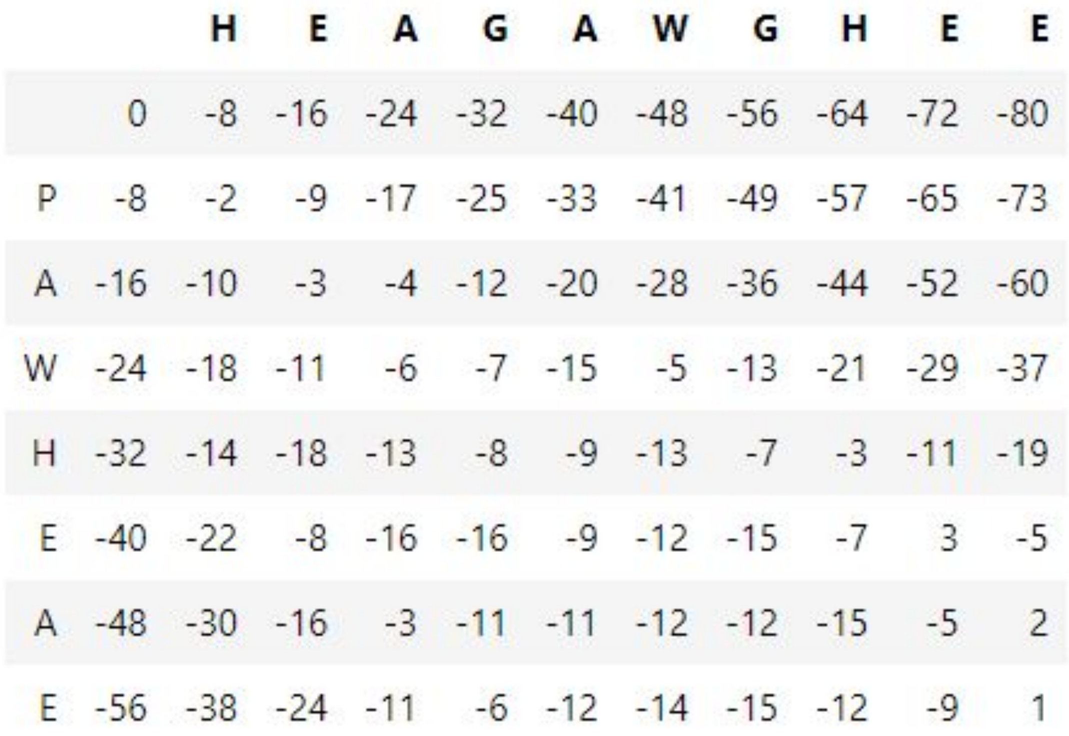


		H	E	A	G	A	W	G	H	E	E
	•	←	←	←	←	←	←	←	←	←	←
P	↑										
A	↑										
W	↑										
H	↑										
E	↑										
A	↑										
E	↑										

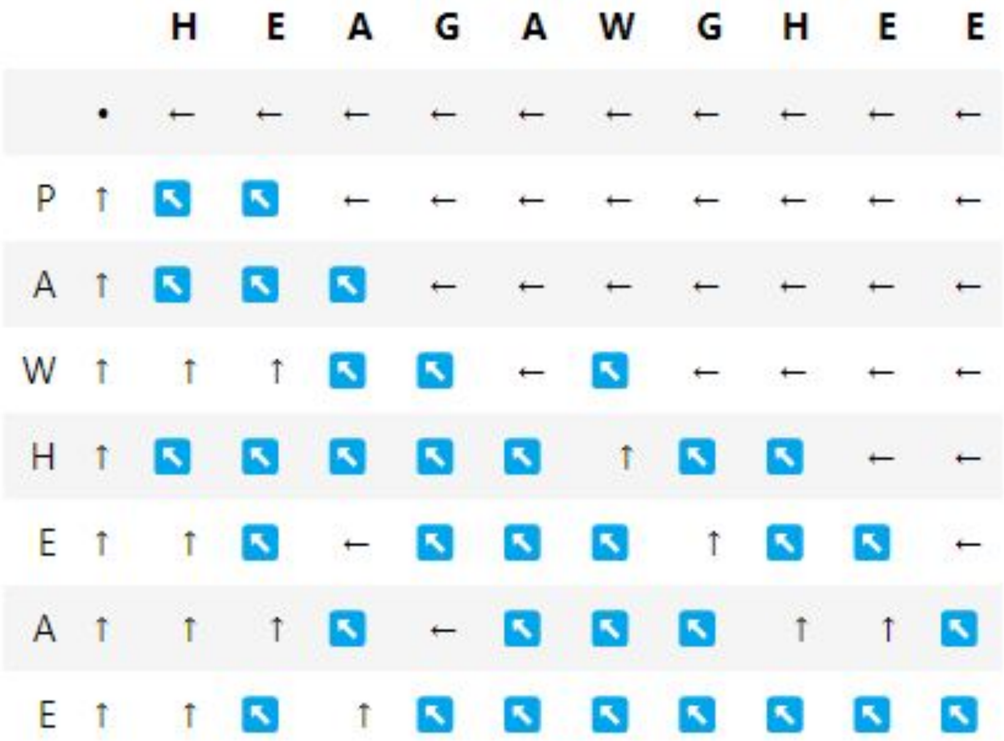
Needleman-Wunsch Alignment

$$\max \begin{cases} F(i-1, j-1) + s(ci, cj) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$



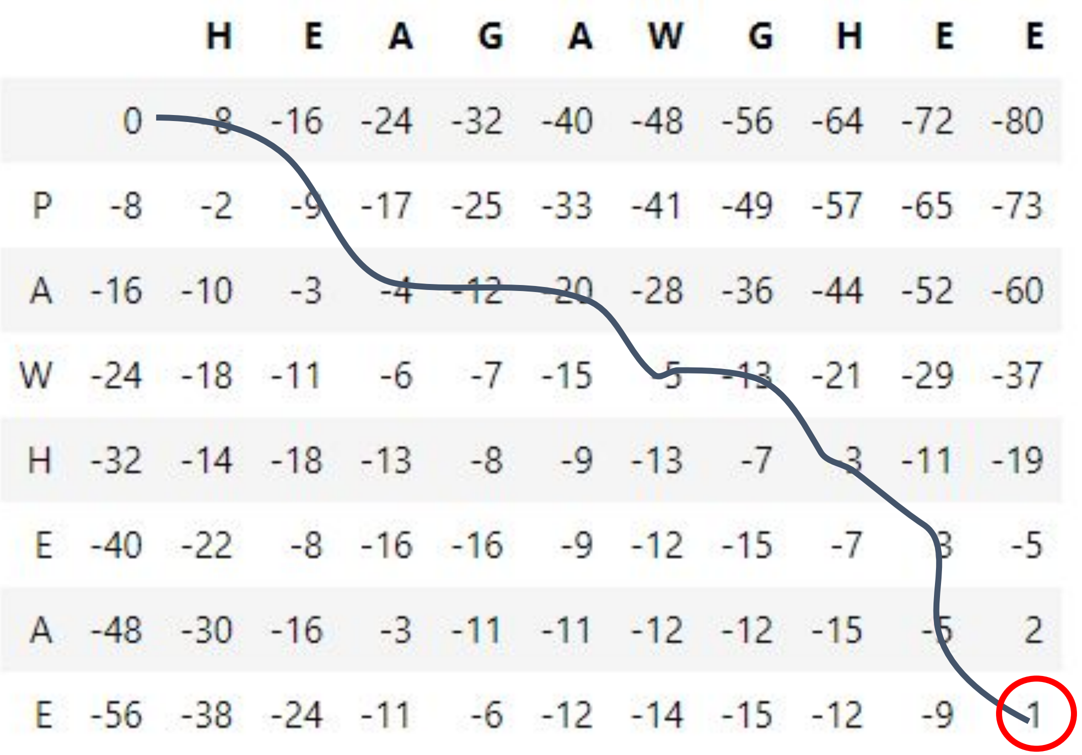


		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	-17	-25	-33	-41	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	-20	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	-5	-13	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	-3	-11	-19
E	-40	-22	-8	-16	-16	-9	-12	-15	-7	3	-5
A	-48	-30	-16	-3	-11	-11	-12	-12	-15	-5	2
E	-56	-38	-24	-11	-6	-12	-14	-15	-12	-9	1

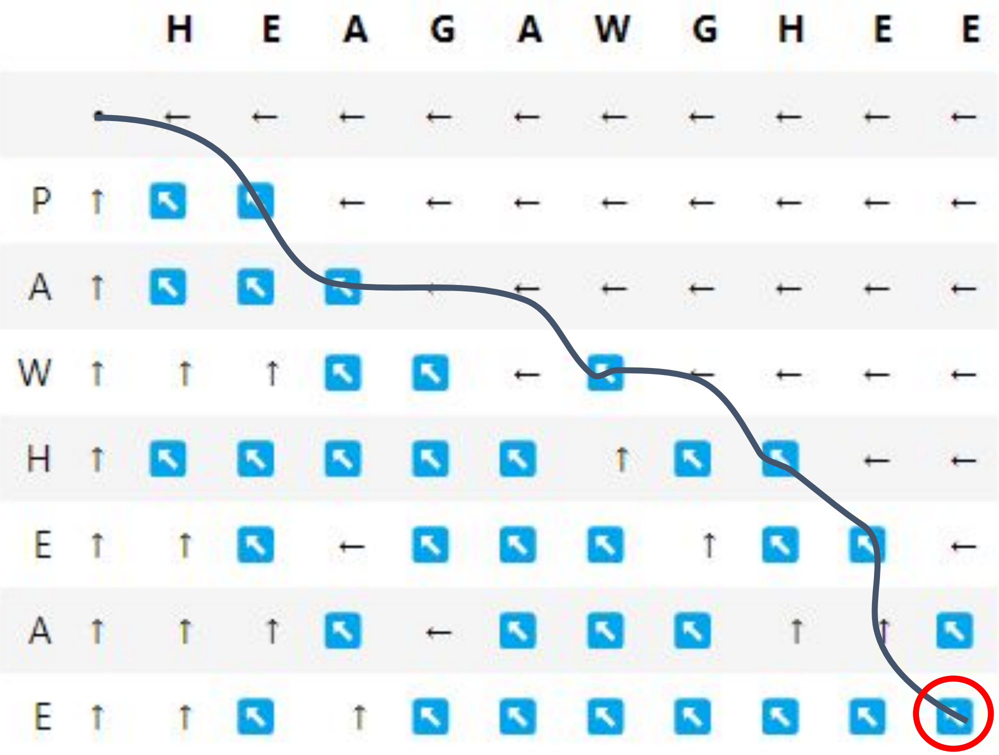


		H	E	A	G	A	W	G	H	E	E
	•	←	←	←	←	←	←	←	←	←	←
P	↑	↖	↖	←	←	←	←	←	←	←	←
A	↑	↖	↖	↖	←	←	←	←	←	←	←
W	↑	↑	↑	↖	↖	←	↖	←	←	←	←
H	↑	↖	↖	↖	↖	↖	↑	↖	↖	←	←
E	↑	↑	↖	←	↖	↖	↖	↑	↖	↖	←
A	↑	↑	↑	↖	←	↖	↖	↖	↑	↑	↖
E	↑	↑	↖	↑	↖	↖	↖	↖	↖	↖	↖

Needleman-Wunsch Alignment



		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	-17	-25	-33	-41	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	-20	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	-5	-13	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	-3	-11	-19
E	-40	-22	-8	-16	-16	-9	-12	-15	-7	3	-5
A	-48	-30	-16	-3	-11	-11	-12	-12	-15	-5	2
E	-56	-38	-24	-11	-6	-12	-14	-15	-12	-9	1



		H	E	A	G	A	W	G	H	E	E
		←	←	←	←	←	←	←	←	←	←
P	↑	↖	↖	←	←	←	←	←	←	←	←
A	↑	↖	↖	↖	←	←	←	←	←	←	←
W	↑	↑	↑	↖	↖	←	↖	←	←	←	←
H	↑	↖	↖	↖	↖	↖	↑	↖	↖	←	←
E	↑	↑	↖	←	↖	↖	↖	↑	↖	↖	←
A	↑	↑	↑	↖	←	↖	↖	↖	↑	↑	↖
E	↑	↑	↖	↑	↖	↖	↖	↖	↖	↖	↖

HEAGAWGHE - E
- PA - -W - HEAE
Score = 1

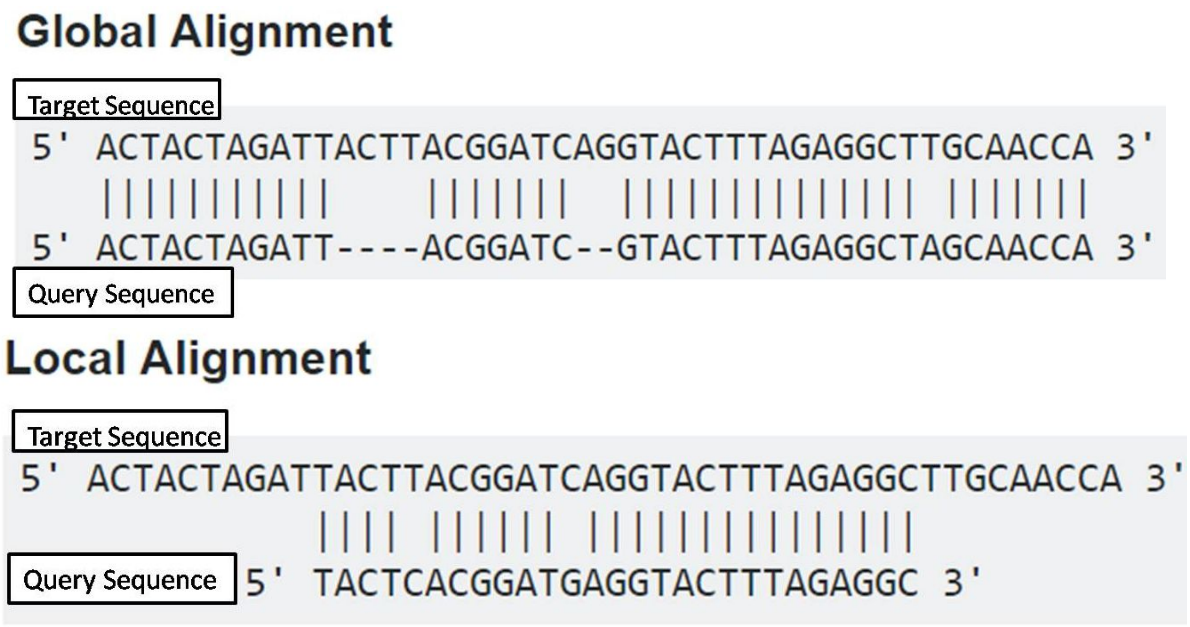
Local vs. Global Alignment

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 Alignment

```
Target Sequence
5' ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA 3'
   |||||||||||    |||||||  |||||||||||||| |||||||
5' ACTACTAGATT----ACGGATC--GTACTTTAGAGGCTAGCAACCA 3'
Query Sequence
```

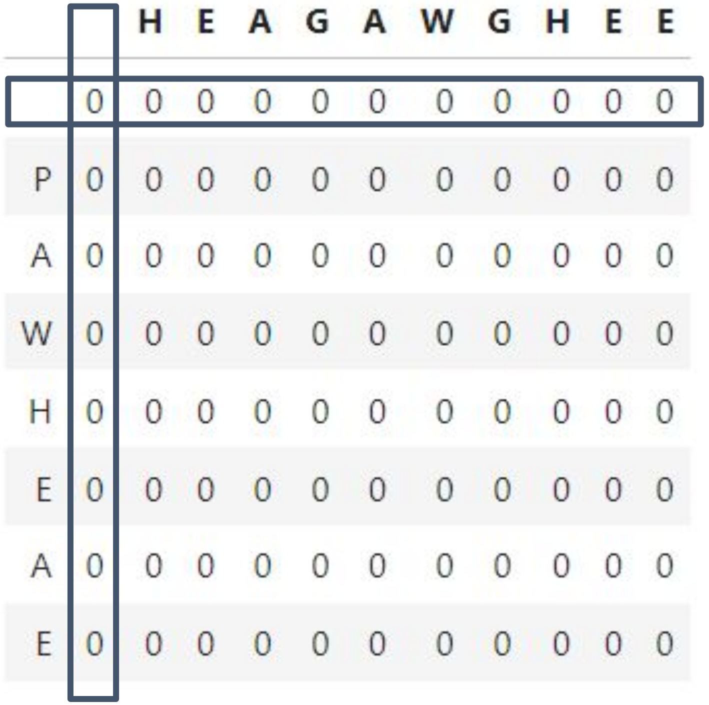
Local Alignment

```
Target Sequence
5' ACTACTAGATTACTTACGGATCAGGTACTTTAGAGGCTTGCAACCA 3'
             |||| |||||| |||||||||||||||
Query Sequence 5' TACTCACGGATGAGGTACTTTAGAGGC 3'
```

Smith-Waterman Local Alignment

HEAGAWGHEE

PAWHEAE

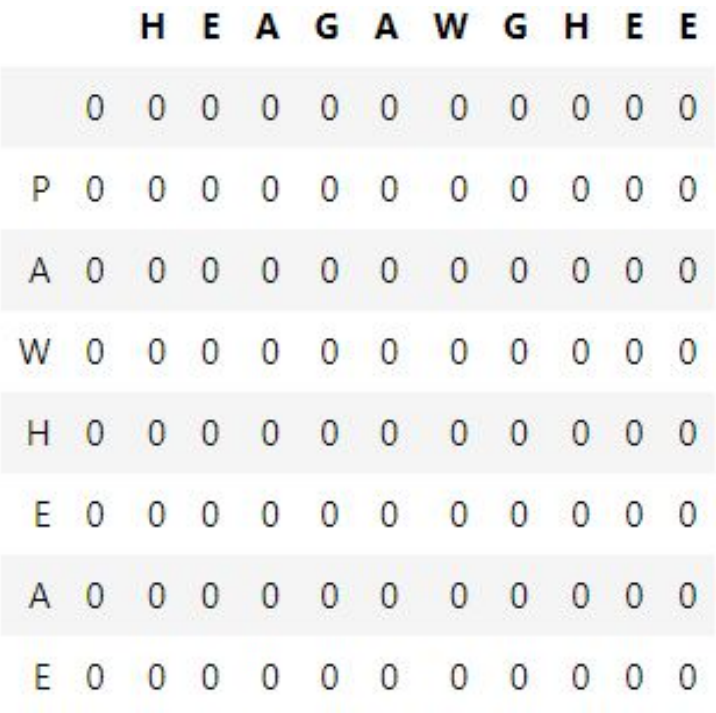


		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0	0

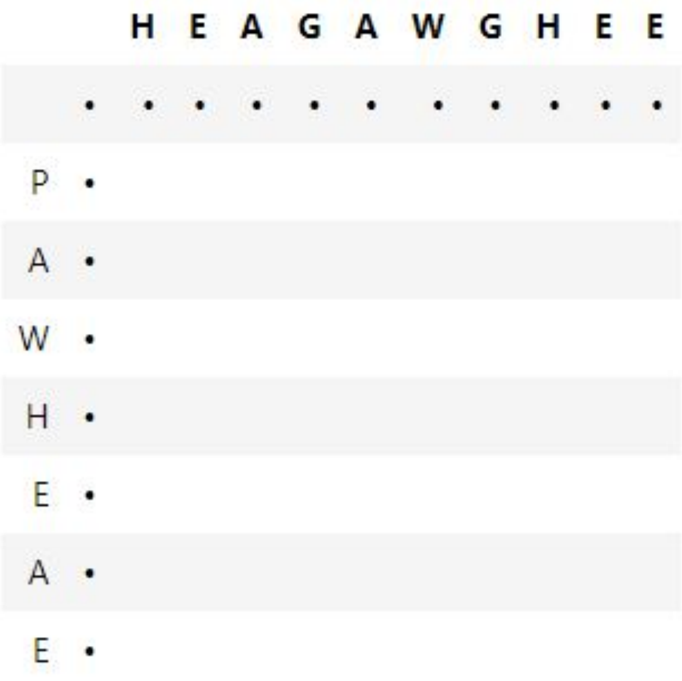
Smith-Waterman Local Alignment

$$F(0,0)=0$$
$$F(i,0)=0$$
$$F(0,j)=0$$

$$T(0,0)=.$$
$$T(i,0)=.$$
$$T(0,j)=.$$



		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0
W	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	0	0	0	0	0	0	0



		H	E	A	G	A	W	G	H	E	E
	•	•	•	•	•	•	•	•	•	•	•
P	•										
A	•										
W	•										
H	•										
E	•										
A	•										
E	•										

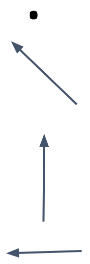
Smith-Waterman Local Alignment

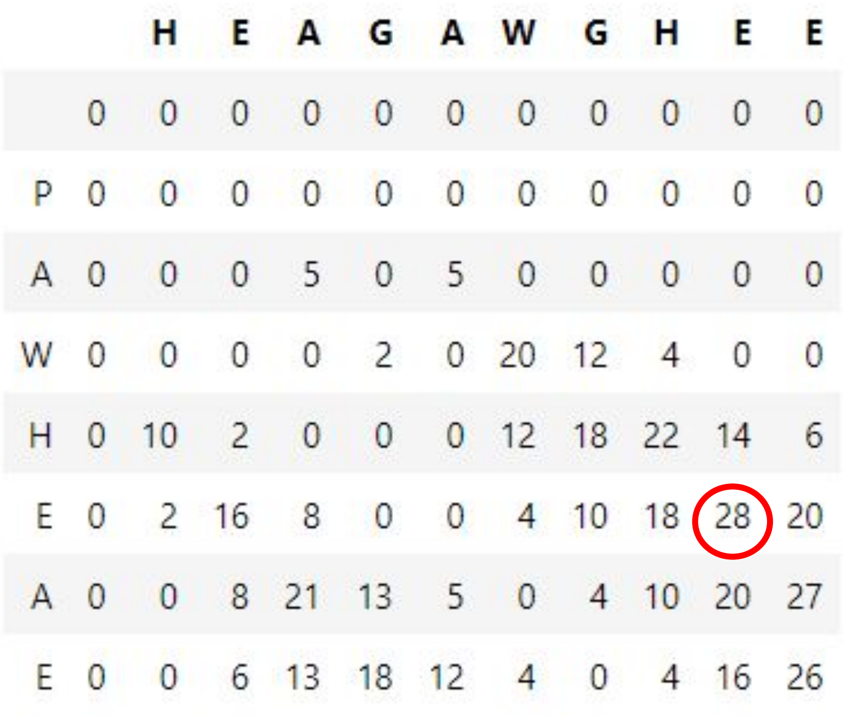
$$\max \begin{cases} 0 \\ F(i-1,j-1)+s(ci,cj) \\ F(i-1,j)-d \\ F(i,j-1)-d \end{cases}$$

AWGHE

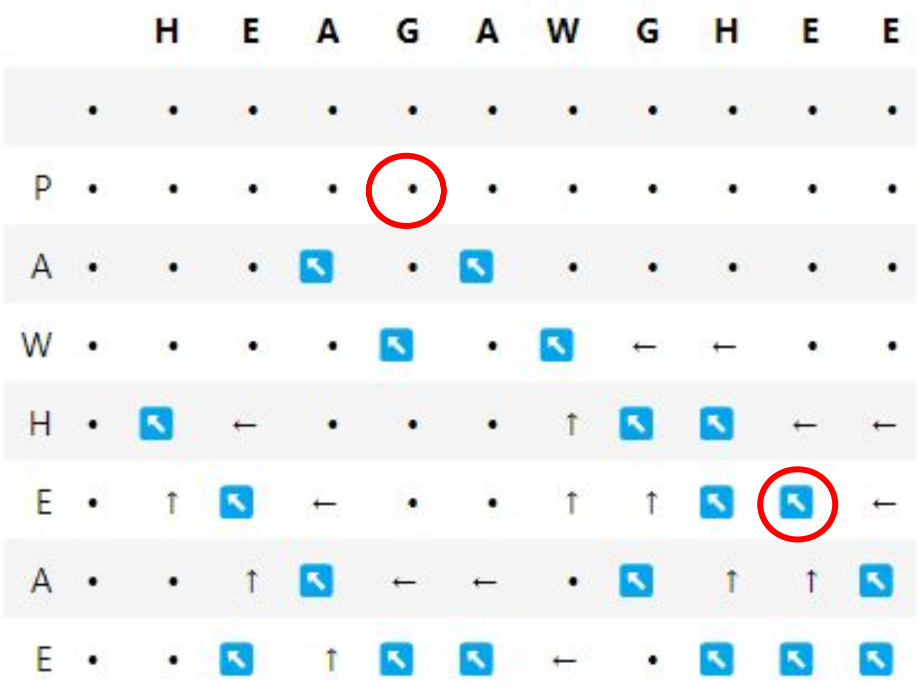
AW - HE

Score = 28





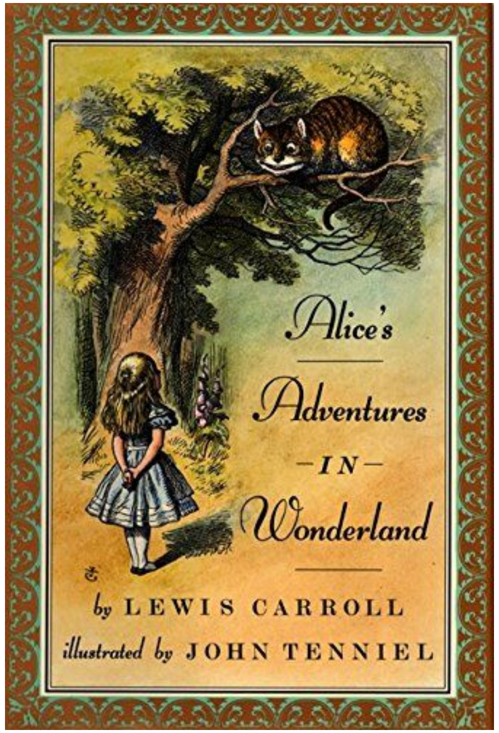
		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	5	0	5	0	0	0	0	0
W	0	0	0	0	2	0	20	12	4	0	0
H	0	10	2	0	0	0	12	18	22	14	6
E	0	2	16	8	0	0	4	10	18	28	20
A	0	0	8	21	13	5	0	4	10	20	27
E	0	0	6	13	18	12	4	0	4	16	26



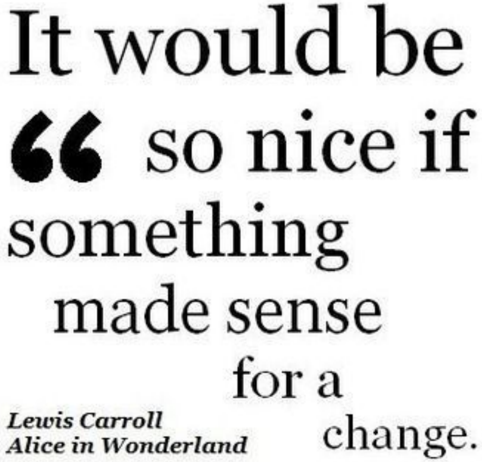
		H	E	A	G	A	W	G	H	E	E
	•	•	•	•	•	•	•	•	•	•	•
P	•	•	•	•	•	•	•	•	•	•	•
A	•	•	•	↖	•	↖	•	•	•	•	•
W	•	•	•	•	↖	•	↖	←	←	•	•
H	•	↖	←	•	•	•	↑	↖	↖	←	←
E	•	↑	↖	←	•	•	↑	↑	↖	↖	←
A	•	•	↑	↖	←	←	•	↖	↑	↑	↖
E	•	•	↖	↑	↖	↖	←	•	↖	↖	↖

Search

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

Search

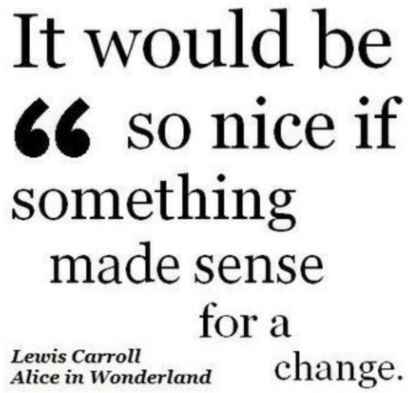
- How can we do it in a timely manner?
  - Brute force
  - Indexing
- Do we allow slight changes?

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?



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

Lewis Carroll
Alice in Wonderland

Sequence Mapping Challenges

Large DBs - millions to billions of nucleotides/AAs

Repetition - biological sequences tend to repeat

Noisy - sequencing errors and real biological variants

BLAST - Basic Local Alignment Search Tool

The most popular alignment tool

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

bioljerk.tumblr.com

BLAST - Algorithm

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

BLAST - Indexing

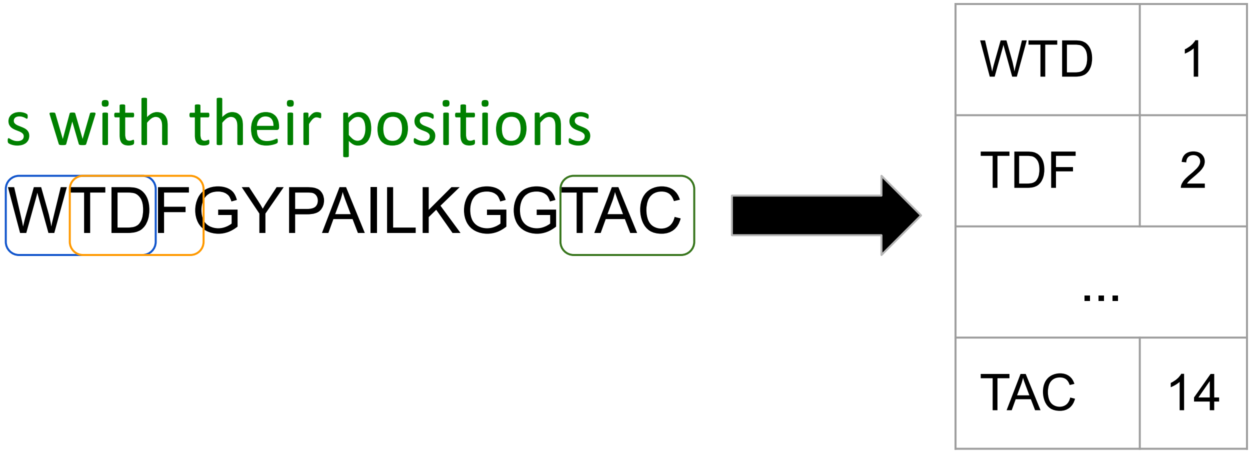
Only needed the first time a DB is used

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



WTDFGYPAILKGGTAC

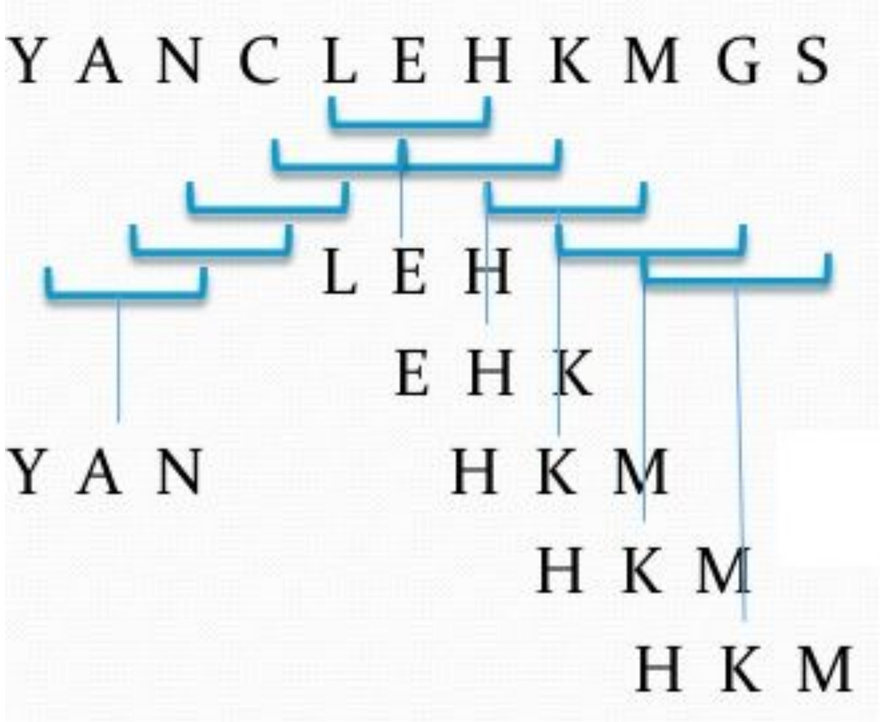
WTD	1
TDF	2
...	
TAC	14

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 Neighbour Words

1. For each word, find all neighbourhood words

   = words with one change

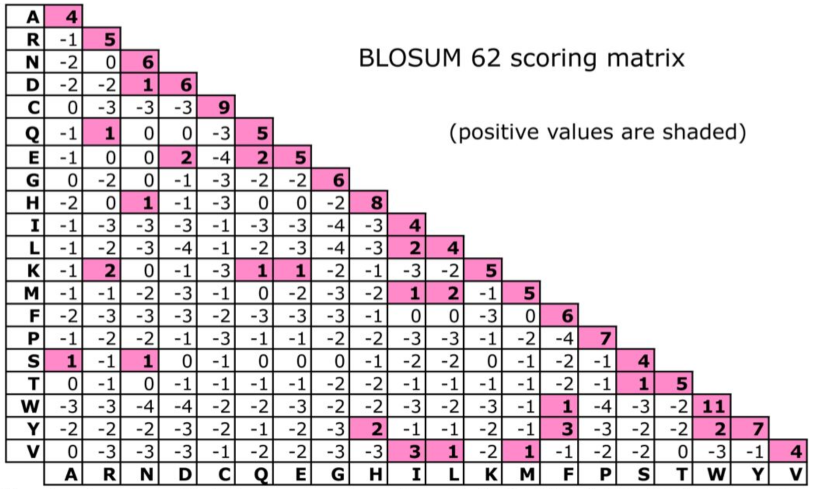
2. Use a scoring matrix to assign each neighbourhood word a score
3. Discard neighbourhood words with score < *T*

BLAST - Breaking Query to Words

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	**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**
	A	**R**	**N**	**D**	**C**	**Q**	**E**	**G**	**H**	**I**	**L**	**K**	**M**	**F**	**P**	**S**	**T**	**W**	**Y**	**V**

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
…	

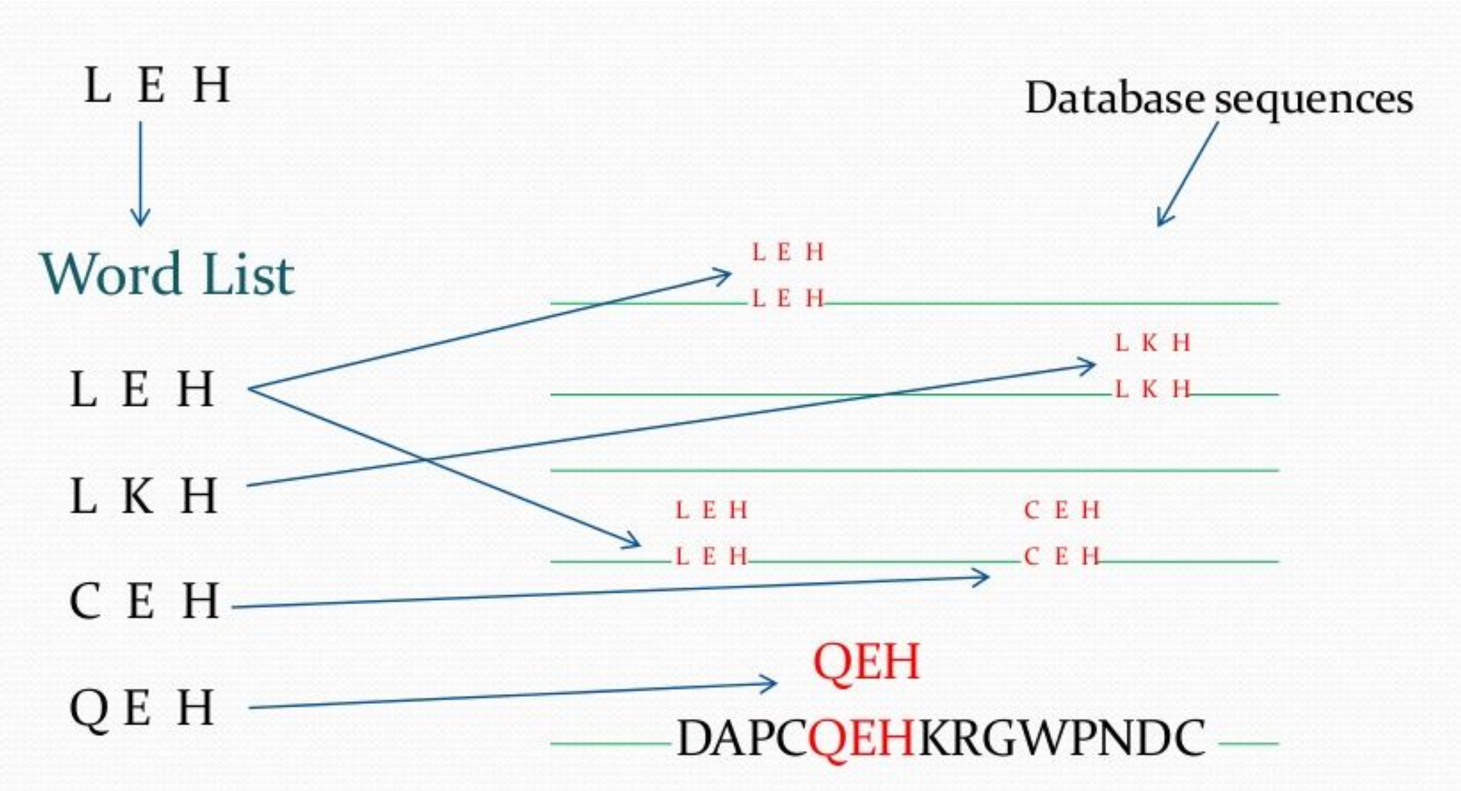
T=11

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

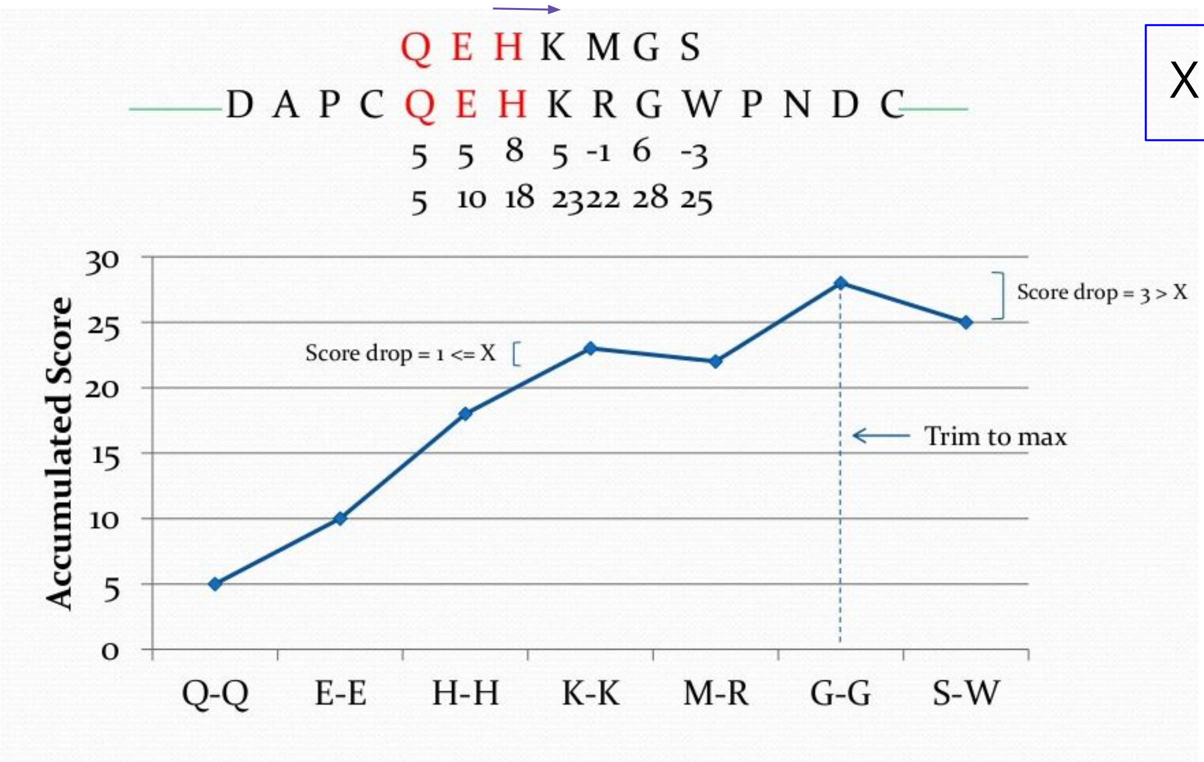


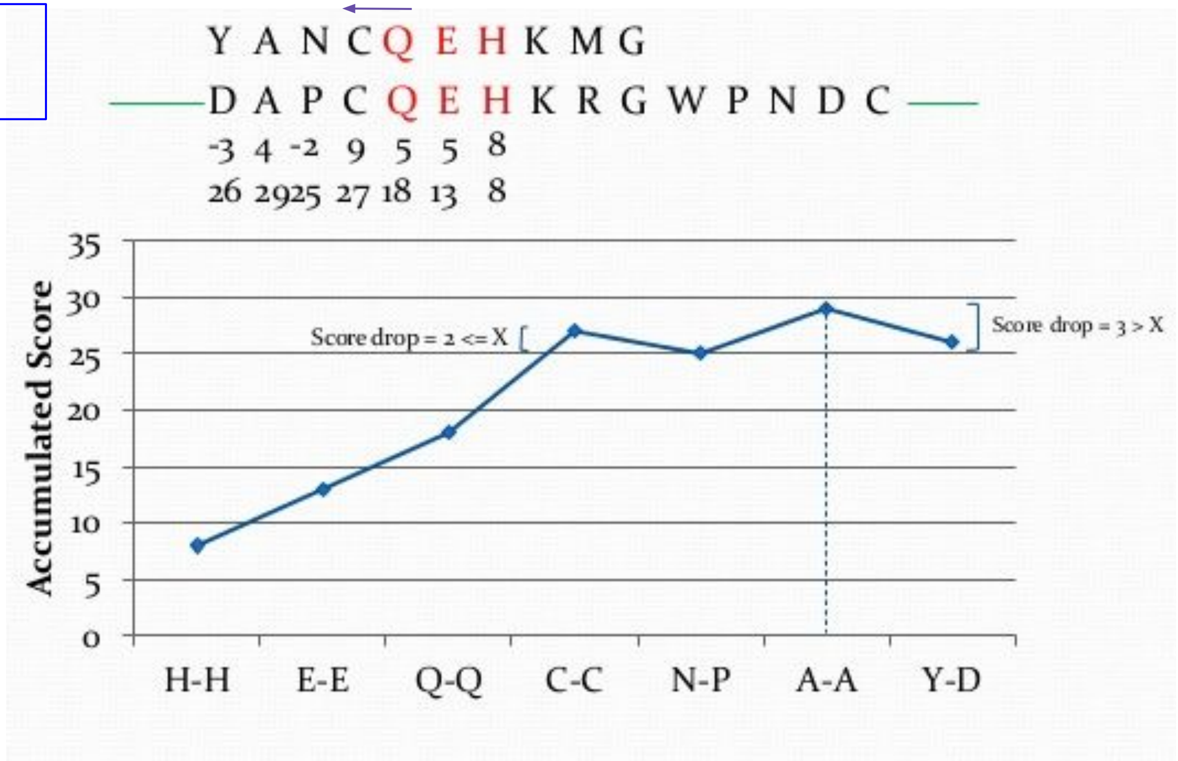
BLAST - Seed Elongation

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

Query:
YANC**LEH**KMGS

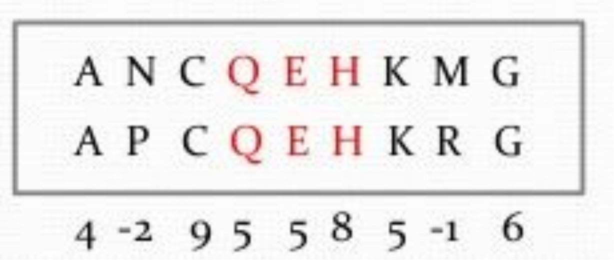
X = 2





BLAST - Scoring the Alignment

Calculate total alignment score



A	N	C	Q	E	H	K	M	G
A	P	C	Q	E	H	K	R	G
4	-2	9	5	5	8	5	-1	6

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

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

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

Smaller E ⟶ better hit

BLAST - Scoring the Alignment

W – word size (query and DB)

T – neighborhood words score cutoff

X – allowed score drop during seed elongation

S – HSP score cutoff

What would happen if we **increase** T?

Scale and Speed

We need to map millions to hundreds of millions of reads

Can we use Blast?

Blastn - ~100 reads / sec

Human genome - ~ 3Gb

Assume 100bp reads

How long to map x10 data to the human genome?

Hint: how many reads do we need?

Can We Use Blast in NGS?

Blastn - ~100 reads / sec

Human genome - ~ 3Gb

Assume 100bp reads

How long to map x10 data to the human genome?

Data required:
3 Gb x 10 = 30 Gb

Reads required:
30 Gb / 100 = 300 M reads

Time to map:
300 M reads / (100 reads/sec) = 3M sec = ~ **35 days**

BWA - Burrows-Wheeler Aligner

Specifically designed for mapping of short reads

Maps ~2,200 reads / sec (one CPU)

Allows parallel computing

Contains three algorithms - the most useful is **BWA-MEM**

BWA - Limitations

Only works for nucleotides (usually DNA, not RNA)

Less effective when:

- Queries are very long
- Reads are highly diverged from the reference
- Reads contain lots of sequencing errors

Usually offers a good accuracy-speed balance

BWA - Algorithm Overview

Step 1: Index the reference genome

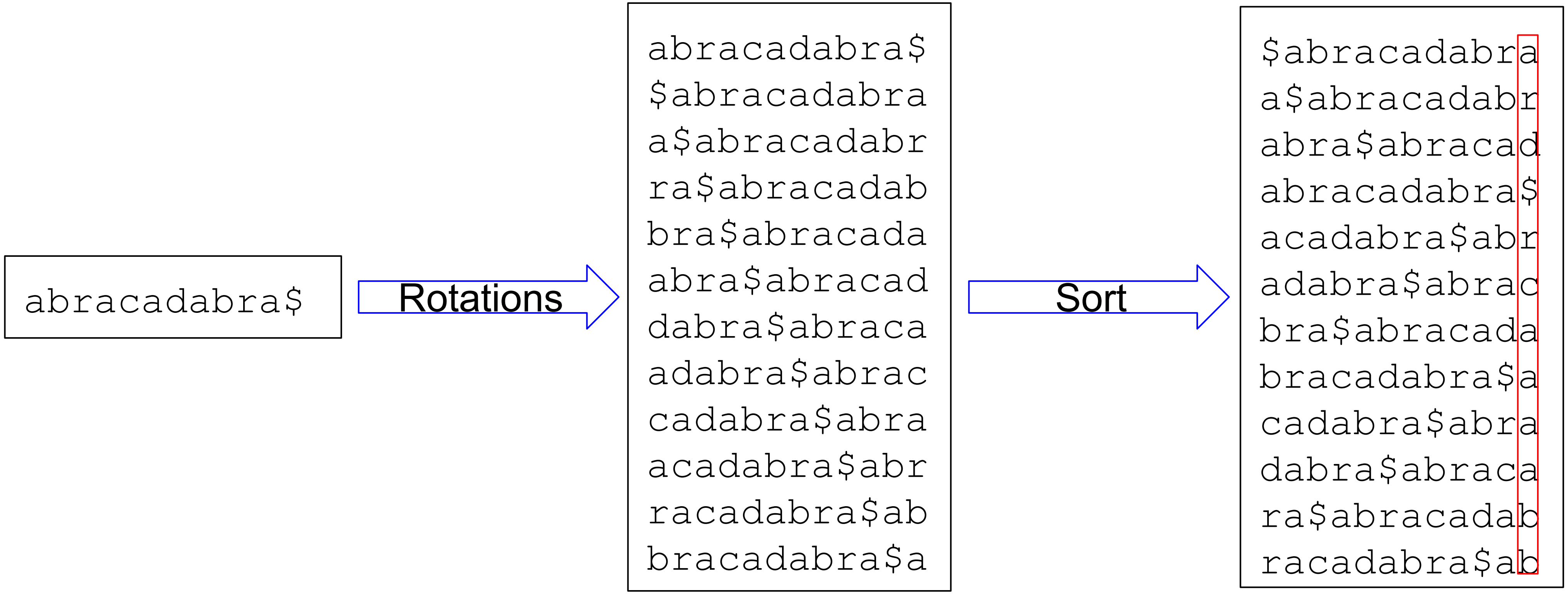
Step 2: Search for reads

Indexing is based on the **Burrows-Wheeler’s transformation**

Index allows easy searching:

- ○ Quick
- ○ Memory efficient

BWA - The Burrows Wheeler Transform



```
abracadabra$
```

Rotations

```
abracadabra$
$abracadabra
a$abracadabr
ra$abracadab
bra$abracada
abra$abracad
dabra$abraca
adabra$abrac
cadabra$abra
acadabra$abr
racadabra$ab
bracadabra$a
```

Sort

```
$abracadabra
a$abracadabr
abra$abracad
abracadabra$
acadabra$abr
adabra$abrac
bra$abracada
bracadabra$a
cadabra$abra
dabra$abraca
ra$abracadab
racadabra$ab
```

BWT(abracadabra$) = ard$rcaaaabb

BWA - The Burrows Wheeler Transform

BWT is **reversible** - we can get back from BWT(G) to G

BWT(G) tends to cluster the same characters together - easy to compress

BWT(abracadabra$) = ard$rcaaaabb

Using some additional data structures, BWT(G) can be searched efficiently

BWA - The Burrows Wheeler Transform

1. Create index of reference genome:

   Input: reference in fasta format

   ```
   $ bwa index genome.fasta
   ```

2. Map reads to reference:

   Input: reads file or pair (for PE data) in fastq format

   ```
   $ bwa mem genome.fasta reads_R1.fq reads_R2.fq -o aln.sam
   ```