# Sequence alignment
# Sekwence alignment
# Sequence alinement

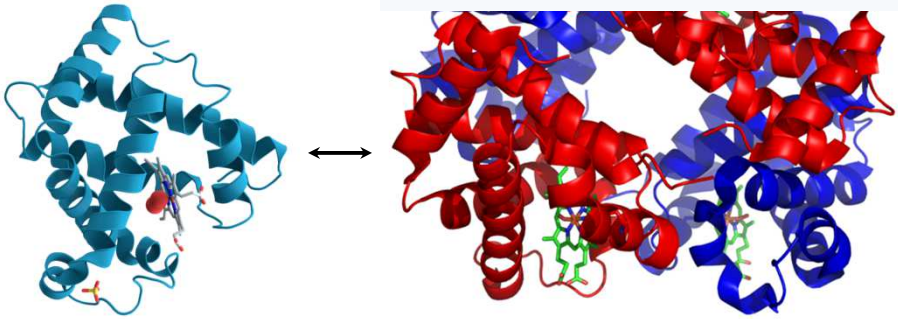
(Drawing heavily from Durbin *et al*., *Biological Sequence Analysis*)

**Systems Biology / Bioinformatics**
**Edward Marcotte, Univ of Texas at Austin**

**Typically, to be "biologically related" means to share a common ancestor. In biology, we call this *homologous*.**

**Two proteins sharing a common ancestor are said to be *homologs*.** Homology often implies structural similarity & sometimes (not always) sequence similarity. **A statistically significant sequence or structural similarity can be used to infer homology (common ancestry).**

e.g., Myoglobin & Hemoglobin

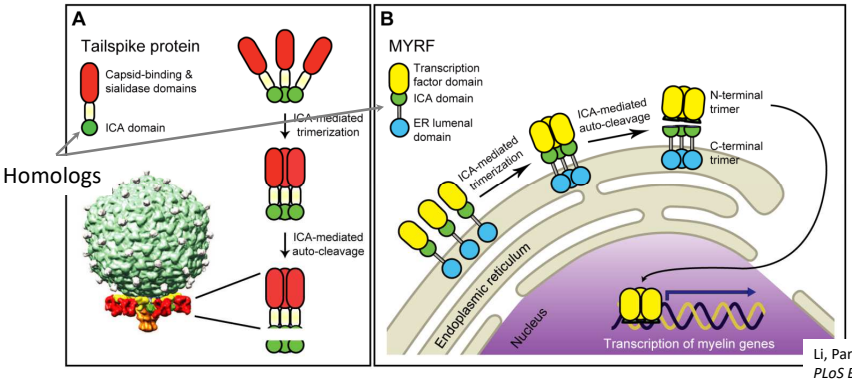


"X-Ray data suggest that the globin chain has the same configuration in the myoglobins and haemoglobins of all vertebrates. "
Kendrew, Perutz, HC Watson. JMB (1965) 13, 669-678

http://en.wikipedia.org/wiki/File:Myoglobin.png & File:1GZX_Haemoglobin.png

**In practice, searching for sequence or structural similarity is one of the most powerful computational approaches to discover a gene's function. We can often gain insight about a protein from its homologs.**

For example, my lab discovered that myelinating the neurons in your brain reuses the same biochemical mechanism that phage use to make capsids. The key breakthrough was recognizing that the human and phage proteins contained homologous domains.



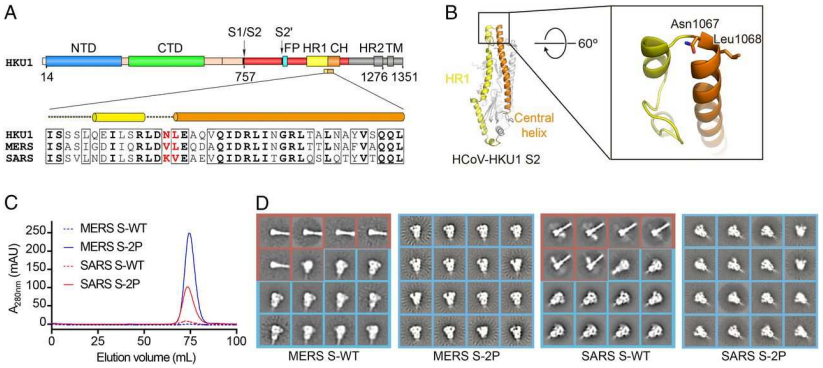

Li, Park, Marcotte *PLoS Biology* (2013)

**& here's the "trillion dollar" paper from the McLellan lab that the SARS-CoV-2 vaccines are designed from based on homology to MERS and SARS spike antigens**

**Immunogenicity and structures of a rationally designed prefusion MERS-CoV spike antigen**

Jesper Pallesen, Nianshuang Wang, Kizzmekia S. Corbett, Daniel Wrapp, Robert N. Kirchdoerfer, Hannah L. Turner, Christopher A. Cottrell, Michelle M. Becker, Lingshu Wang, Wei Shi, Wing-Pui Kong, Erica L. Andres, Arminja N. Kettenbach, Mark R. Denison, James D. Chappell, Barney S. Graham, Andrew B. Ward, and Jason S. McLellan

PNAS August 29, 2017 114 (35) E7348-E7357; first published August 14, 2017; https://doi.org/10.1073/pnas.1707304114



**Sequence alignment algorithms such as BLAST, PSI-BLAST, FASTA, MMSeqs2, the Needleman–Wunsch & Smith-Waterman algorithms are arguably some of the most important driver technologies of modern biology and underlie the sequencing revolution.**

**So, let's start learning bioinformatics algorithms by learning how to align two protein sequences.**

**Live demo:**

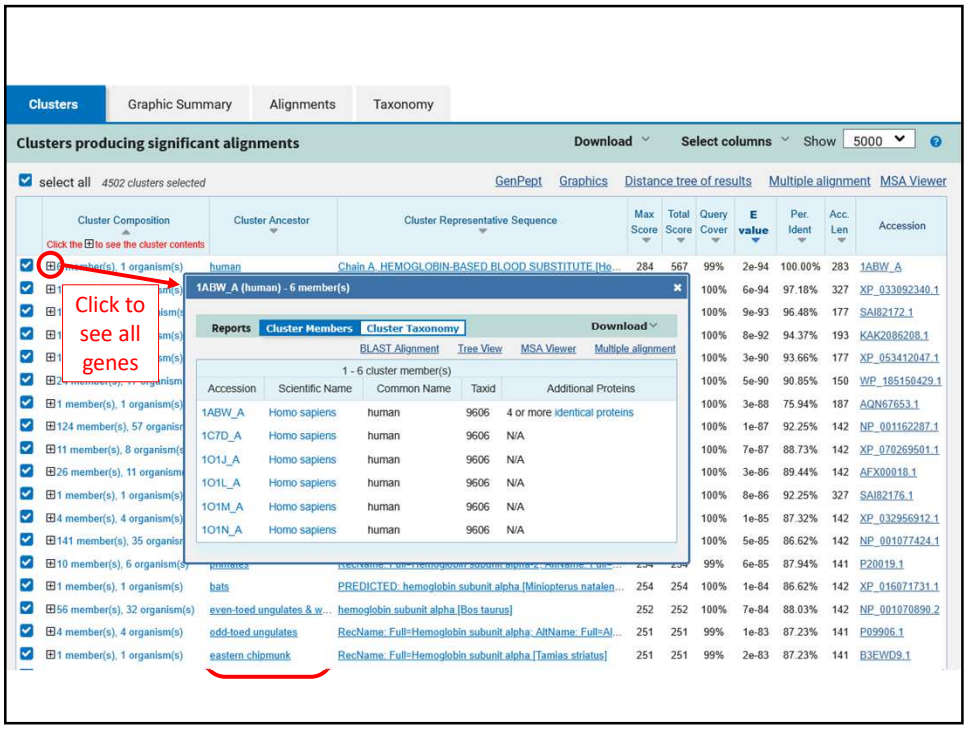
http://blast.ncbi.nlm.nih.gov/Blast.cgi?PROGRAM=blastp&BLAST_PROGRAMS=blastp&PAGE_TYPE=BlastSearch&SHOW_DEFAULTS=on&LINK_LOC=blasthome

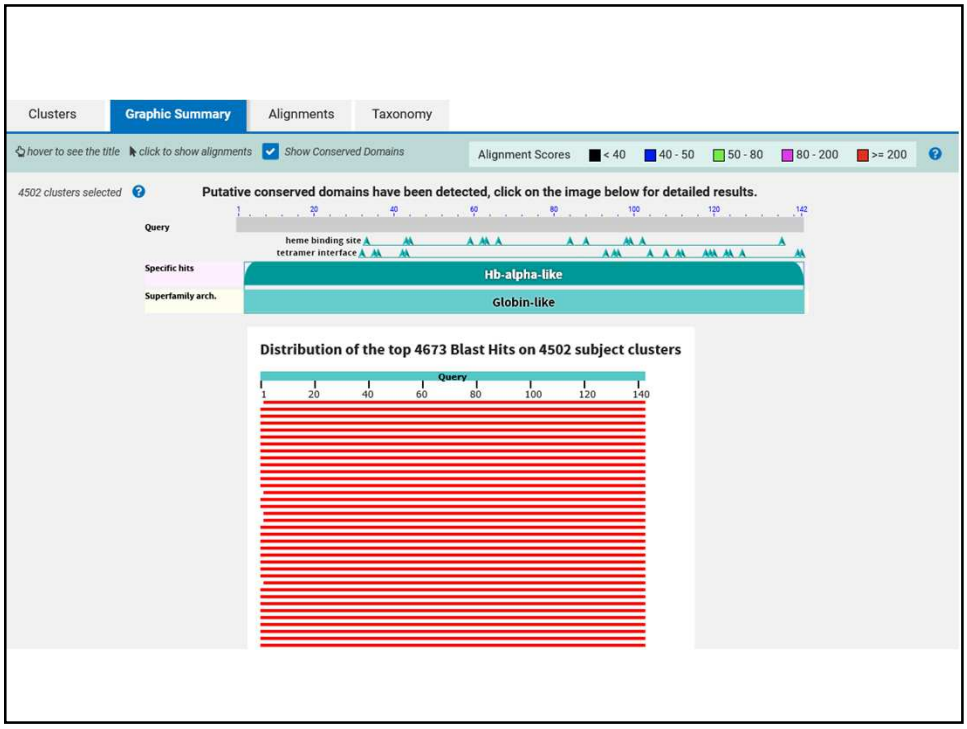
MVLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLSHGSAQVKGHG
KKVADALTNAVAHVDDMPNALSALSDLHAHKLRVDPVNFKLLSHCLLVTLAAHLPAEFTP
AVHASLDKFLASVSTVLTSKYR

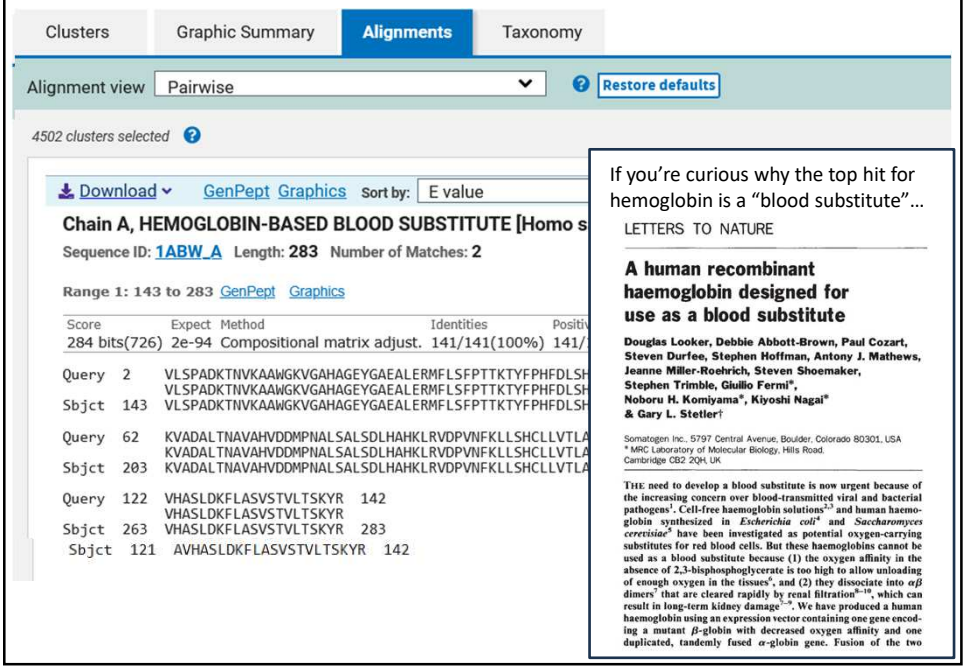
The next few slides show the data from searching this dbase
(#'s may be a bit different from the live version):

Title: clustered nr
Description: ClusteredNR is derived from the protein nr database by clustering sequences at 90% identity and 90% length. more...
Molecule Type: Protein
Update date: 2025/12/27
Number of sequences: 470,748,714

BLAST searches a pre-clustered dataset to give more taxonomic diversity to your results


Clusters
Graphic Summary
Alignments
Taxonomy
Clusters producing significant alignments
Download
Select columns
Show 5000
select all 4502 clusters selected
GenPept
Graphics
Distance tree of results
Multiple alignment
MSA Viewer
Cluster Composition
Click the ⊞ to see the cluster contents
Cluster Ancestor
Cluster Representative Sequence
Max Score
Total Score
Query Cover
E value
Per. Ident
Acc. Len
Accession
Click to see all genes
1ABW_A (human) - 6 member(s)
Reports
Cluster Members
Cluster Taxonomy
Download
BLAST Alignment
Tree View
MSA Viewer
Multiple alignment
1 - 6 cluster member(s)
Accession
Scientific Name
Common Name
Taxid
Additional Proteins
1ABW_A Homo sapiens human 9606 4 or more identical proteins
1C7D_A Homo sapiens human 9606 N/A
1O1J_A Homo sapiens human 9606 N/A
1O1L_A Homo sapiens human 9606 N/A
1O1M_A Homo sapiens human 9606 N/A
1O1N_A Homo sapiens human 9606 N/A
human
Chain A, HEMOGLOBIN-BASED BLOOD SUBSTITUTE [Ho...
284 567 99% 2e-94 100.00% 283 1ABW_A
100% 6e-94 97.18% 327 XP_033092340.1
100% 9e-93 96.48% 177 SAI82172.1
100% 8e-92 94.37% 193 KAK2086208.1
100% 3e-90 93.66% 177 XP_053412047.1
100% 5e-90 90.85% 150 WP_185150429.1
100% 3e-88 75.94% 187 AQN67653.1
100% 1e-87 92.25% 142 NP_001162287.1
100% 7e-87 88.73% 142 XP_070269501.1
100% 3e-86 89.44% 142 AFX00018.1
100% 8e-86 92.25% 327 SAI82176.1
100% 1e-85 87.32% 142 XP_032956912.1
100% 5e-85 86.62% 142 NP_001077424.1
99% 6e-85 87.94% 141 P20019.1
bats
PREDICTED: hemoglobin subunit alpha [Miniopterus natalen...
254 254 100% 1e-84 86.62% 142 XP_016071731.1
even-toed ungulates & w...
hemoglobin subunit alpha [Bos taurus]
252 252 100% 7e-84 88.03% 142 NP_001070890.2
odd-toed ungulates
RecName: Full=Hemoglobin subunit alpha; AltName: Full=Al...
251 251 99% 1e-83 87.23% 141 P09906.1
eastern chipmunk
RecName: Full=Hemoglobin subunit alpha [Tamias striatus]
251 251 99% 2e-83 87.23% 141 B3EWD9.1


Clusters
Graphic Summary
Alignments
Taxonomy
hover to see the title
click to show alignments
Show Conserved Domains
Alignment Scores
< 40
40 - 50
50 - 80
80 - 200
>= 200
4502 clusters selected
Putative conserved domains have been detected, click on the image below for detailed results.
Query
heme binding site
tetramer interface
Specific hits
Hb-alpha-like
Superfamily arch.
Globin-like
Distribution of the top 4673 Blast Hits on 4502 subject clusters
Query




If you're curious why the top hit for hemoglobin is a "blood substitute"…



LETTERS TO NATURE

**A human recombinant haemoglobin designed for use as a blood substitute**

**Douglas Looker, Debbie Abbott-Brown, Paul Cozart, Steven Durfee, Stephen Hoffman, Antony J. Mathews, Jeanne Miller-Roehrich, Steven Shoemaker, Stephen Trimble, Giulio Fermi*, Noboru H. Komiyama*, Kiyoshi Nagai* & Gary L. Stetler†**

Somatogen Inc., 5797 Central Avenue, Boulder, Colorado 80301, USA
* MRC Laboratory of Molecular Biology, Hills Road, Cambridge CB2 2QH, UK

THE need to develop a blood substitute is now urgent because of the increasing concern over blood-transmitted viral and bacterial pathogens[1]. Cell-free haemoglobin solutions[2,3] and human haemoglobin synthesized in *Escherichia coli*[4] and *Saccharomyces cerevisiae*[5] have been investigated as potential oxygen-carrying substitutes for red blood cells. But these haemoglobins cannot be used as a blood substitute because (1) the oxygen affinity in the absence of 2,3-bisphosphoglycerate is too high to allow unloading of enough oxygen in the tissues[6], and (2) they dissociate into $\alpha\beta$ dimers[7] that are cleared rapidly by renal filtration[8–10], which can result in long-term kidney damage[7–9]. We have produced a human haemoglobin using an expression vector containing one gene encoding a mutant $\beta$-globin with decreased oxygen affinity and one duplicated, tandemly fused $\alpha$-globin gene. Fusion of the two

# Protein sequence alignment

**Two biologically related proteins with similar sequences:**

```
FlgA1 EAGNVKLKRGRLDTLPPRTVLDINQLVDAISLRDLSPDQPIQLTQFRQAWRVKAGQRVNVIASGD
        ++K+K+GRLDTLPP  +L+ N    A+SLR ++  QP+     R+ W +KAGQ V V+A G+
FlgA2 TLQDIKMKQGRLDTLPPGALLEPNFAQGAVSLRQINAGQPLTRNMLRRLWIIKAGQDVQVLALGE
```

**Also biologically related (& fold up into the same 3D protein structure):**

```
FlgA1 EAGNVKLKRGRLDTLPPRTVLDINQLVDAISLRDLSPDQPIQLTQFRQAWRVKAGQRVNVIASGD
       A   +          P        +L   I+ R L P + I     R+AW V+ G  V V
FlgA3 LAALKQVTLIAGKHKPDAMATHAEELQGKIAKRTLLPGRYIPTAAIREAWLVEQGAAVQVFFIAG
```

**But these are biologically unrelated (& fold up into unrelated structures):**

```
FlgA1 AGNVKLKRGRLDTLPPRTVLDINQLVDAISLRDLSPDQPIQLTQFRQA-WRVKAGQRVNVIASGD
      AG+V  K G +  + PRT  ++       I+     P  PI    +++A WRV A + V V+  GD
HvcPP AGHV--KNGTMRIVGPRTCSNVWNGTFPINATTTGPSIPIPAPNYKKALWRVSATEYVEVVRVGD
```

(FYI, we'll draw examples from Durbin *et al.*, *Biological Sequence Analysis*, Ch. 1 & 2).

**To align two sequences, we need to perform 3 steps:**

1. **We need some way to decide which alignments are better than others.**
   **For this, we'll invent a way to give the alignments a "score" indicating their quality.**

2. **Align the two proteins so that they get the best possible score.**

3. **Decide if the score is "good enough" for us to believe the alignment is biologically significant.**

**To align two sequences, we need to perform 3 steps:**

1. We need some way to decide which alignments are better than others.
   For this, **we'll invent a way to give the alignments a "score" indicating their quality.**

2. Align the two proteins so that they get the best possible score.

3. Decide if the score is "good enough" for us to believe the alignment is biologically significant.

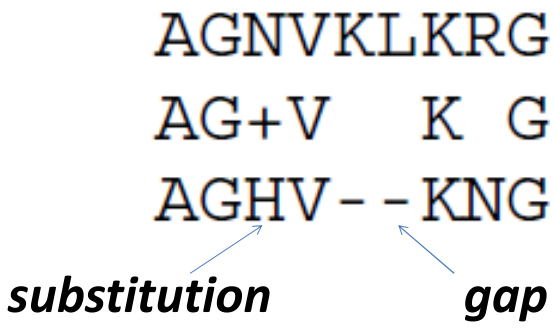
We'll treat mutations as independent events.

This allows us to create an ***additive scoring scheme.***
The score for a sequence alignment will be the sum of the scores for aligning each of the individual positions in two sequences.

**What kind of mutations should we expect?**

***Substitutions***, ***insertions*** and ***deletions***.

Insertions and deletions can be treated as equivalent events by considering one or the other sequence as the reference, and are usually called ***gaps***.



Let's consider two models:

First, a **random** model, where amino acids in the sequences occur independently at some given frequencies.

The probability of observing an alignment between *x* and *y* is just the product of the frequencies (q) with which we find each amino acid.

We can write this as:

What does the capital pi mean?

$$P(x, y \mid R) = \prod_i q_{x_i} \prod_j q_{y_j}$$

What's this mean?

What's this mean?

*i* is just a counter indicating the sequence position

Here's our pair of proteins from before:

```
FlgA1 EAGNVKLKRGRLDTLPPRTVLDINQLVDAISLRDLSPDQPIQLTQFRQAWRVKAGQRVNVIASGD
        ++K+K+GRLDTLPP  +L+ N    A+SLR ++  QP+     R+ W +KAGQ V V+A G+
FlgA2 TLQDIKMKQGRLDTLPPGALLEPNFAQGAVSLRQINAGQPLTRNMLRRLWIIKAGQDVQVLALGE
```

So, our random model is:

$$P(x, y \mid R) = \prod_i q_{x_i} \prod_j q_{y_j} = \underbrace{\text{f(E)*f(A)*f(G)*...*f(G)*f(D)}}\underbrace{\text{*f(T)*f(L)*f(Q)*...*f(G)*f(E)}}$$

frequencies of each amino acid in protein 1 & 2

Second, a **match** model, where amino acids at a given position in the alignment arise from some common ancestor with a probability given by the joint probability $p_{ab}$.

So, under this model, the probability of the alignment is the product of the probabilities of seeing the individual amino acids aligned.

We can write that as:

What does the capital pi mean again?

$$P(x, y \mid M) = \prod_i p_{x_i y_i}$$

What's this mean?

What's this mean?

Here's our pair of proteins from before:

```
FlgA1 EAGNVKLKRGRLDTLPPRTVLDINQLVDAISLRDLSPDQPIQLTQFRQAWRVKAGQRVNVIASGD
        ++K+K+GRLDTLPP  +L+ N    A+SLR ++  QP+     R+ W +KAGQ V V+A G+
FlgA2 TLQDIKMKQGRLDTLPPGALLEPNFAQGAVSLRQINAGQPLTRNMLRRLWIIKAGQDVQVLALGE
```

So, our match model is:

$$P(x, y \mid M) = \prod_i p_{x_i y_i}$$ = f(E aligned with T)*f(A aligned with L)*…f(D aligned with E)

frequencies of the aligned residue pairs

To decide which model better describes an alignment, we'll take the ratio:

$$\frac{P(x,y \mid M)}{P(x,y \mid R)} = \frac{\prod_i p_{x_i y_i}}{\prod_i q_{x_i} \prod_j q_{y_j}} = \prod_i \frac{p_{x_i y_i}}{q_{x_i} q_{y_i}}$$

What did these mean again?

Such a ratio of probabilities under 2 different models is called an ***odds ratio***.

Where else have you heard odds ratios used?

Basically: if the ratio > 1, model *M* is more probable
if < 1, model *R* is more probable.

Now, to convert this to an additive score *S*, we can simply take the logarithm of the odds ratio (called the ***log odds ratio***):

$$S = \sum_i s(x_i, y_i)$$

This is just the score for aligning one amino acid with another amino acid:

$$s(a,b) = \log\left(\frac{p_{ab}}{p_a p_b}\right)$$

Here written *a* and *b* rather than $x_i$ and $y_i$ to emphasize that this score reflects the *inherent preference* of the two amino acids (*a* and *b*) to be aligned.

**Almost done with step 1…**

The last trick:
Take a big set of pre-aligned protein sequence alignments (that are correct!) and measure all of the pairwise amino acid substitution scores (the $s(a,b)$'s). Put them in a 20x20 ***amino acid substitution matrix :***

| | A | R | N | D | C | Q | E | G | H | I | L | K | M | F | P | S | T | W | Y | V |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 5 | -2 | -1 | -2 | -1 | -1 | -1 | 0 | -2 | -1 | -2 | -1 | -1 | -3 | -1 | 1 | 0 | -3 | -2 | 0 |
| R | -2 | 7 | -1 | -2 | -4 | 1 | 0 | -3 | 0 | -4 | -3 | 3 | -2 | -3 | -3 | -1 | -1 | -3 | -1 | -3 |
| N | -1 | -1 | 7 | 2 | -2 | 0 | 0 | 0 | 1 | -3 | -4 | 0 | -2 | -4 | -2 | 1 | 0 | -4 | -2 | -3 |
| D | -2 | -2 | 2 | 8 | -4 | 0 | 2 | -1 | -1 | -4 | -4 | -1 | -4 | -5 | -1 | 0 | -1 | -5 | -3 | -4 |
| C | -1 | -4 | -2 | -4 | 13 | -3 | -3 | -3 | -3 | -2 | -2 | -3 | -2 | -2 | -4 | -1 | -1 | -5 | -3 | -1 |
| Q | -1 | 1 | 0 | 0 | -3 | 7 | 2 | -2 | 1 | -3 | -2 | 2 | 0 | -4 | -1 | 0 | -1 | -1 | -1 | -3 |
| E | -1 | 0 | 0 | 2 | -3 | 2 | 6 | -3 | 0 | -4 | -3 | 1 | -2 | -3 | -1 | -1 | -1 | -3 | -2 | -3 |
| G | 0 | -3 | 0 | -1 | -3 | -2 | -3 | 8 | -2 | -4 | -4 | -2 | -3 | -4 | -2 | 0 | -2 | -3 | -3 | -4 |
| H | -2 | 0 | 1 | -1 | -3 | 1 | 0 | -2 | 10 | -4 | -3 | 0 | -1 | -1 | -2 | -1 | -2 | -3 | 2 | -4 |
| I | -1 | -4 | -3 | -4 | -2 | -3 | -4 | -4 | -4 | 5 | 2 | -3 | 2 | 0 | -3 | -3 | -1 | -3 | -1 | 4 |
| L | -2 | -3 | -4 | -4 | -2 | -2 | -3 | -4 | -3 | 2 | 5 | -3 | 3 | 1 | -4 | -3 | -1 | -2 | -1 | 1 |
| K | -1 | 3 | 0 | -1 | -3 | 2 | 1 | -2 | 0 | -3 | -3 | 6 | -2 | -4 | -1 | 0 | -1 | -3 | -2 | -3 |
| M | -1 | -2 | -2 | -4 | -2 | 0 | -2 | -3 | -1 | 2 | 3 | -2 | 7 | 0 | -3 | -2 | -1 | -1 | 0 | 1 |
| F | -3 | -3 | -4 | -5 | -2 | -4 | -3 | -4 | -1 | 0 | 1 | -4 | 0 | 8 | -4 | -3 | -2 | 1 | 4 | -1 |
| P | -1 | -3 | -2 | -1 | -4 | -1 | -1 | -2 | -2 | -3 | -4 | -1 | -3 | -4 | 10 | -1 | -1 | -4 | -3 | -3 |
| S | 1 | -1 | 1 | 0 | -1 | 0 | -1 | 0 | -1 | -3 | -3 | 0 | -2 | -3 | -1 | 5 | 2 | -4 | -2 | -2 |
| T | 0 | -1 | 0 | -1 | -1 | -1 | -1 | -2 | -2 | -1 | -1 | -1 | -1 | -2 | -1 | 2 | 5 | -3 | -2 | 0 |
| W | -3 | -3 | -4 | -5 | -5 | -1 | -3 | -3 | -3 | -3 | -2 | -3 | -1 | 1 | -4 | -4 | -3 | 15 | 2 | -3 |
| Y | -2 | -1 | -2 | -3 | -3 | -1 | -2 | -3 | 2 | -1 | -1 | -2 | 0 | 4 | -3 | -2 | -2 | 2 | 8 | -1 |
| V | 0 | -3 | -3 | -4 | -1 | -3 | -3 | -4 | -4 | 4 | 1 | -3 | 1 | -1 | -3 | -2 | 0 | -3 | -1 | 5 |

This is the **BLOSUM50** matrix.
(The numbers are scaled & rounded off to the nearest integer):

What's the score for aspartate (D) aligning with itself?
How about aspartate with phenylalanine (F)? Why?

| | A | R | N | D | C | Q | E | G | H | I | L | K | M | F | P | S | T | W | Y | V |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 5 | -2 | -1 | -2 | -1 | -1 | -1 | 0 | -2 | -1 | -2 | -1 | -1 | -3 | -1 | 1 | 0 | -3 | -2 | 0 |
| R | -2 | 7 | -1 | -2 | -4 | 1 | 0 | -3 | 0 | -4 | -3 | 3 | -2 | -3 | -3 | -1 | -1 | -3 | -1 | -3 |
| N | -1 | -1 | 7 | 2 | -2 | 0 | 0 | 0 | 1 | -3 | -4 | 0 | -2 | -4 | -2 | 1 | 0 | -4 | -2 | -3 |
| D | -2 | -2 | 2 | 8 | -4 | 0 | 2 | -1 | -1 | -4 | -4 | -1 | -4 | -5 | -1 | 0 | -1 | -5 | -3 | -4 |
| C | -1 | -4 | -2 | -4 | 13 | -3 | -3 | -3 | -3 | -2 | -2 | -3 | -2 | -2 | -4 | -1 | -1 | -5 | -3 | -1 |
| Q | -1 | 1 | 0 | 0 | -3 | 7 | 2 | -2 | 1 | -3 | -2 | 2 | 0 | -4 | -1 | 0 | -1 | -1 | -1 | -3 |
| E | -1 | 0 | 0 | 2 | -3 | 2 | 6 | -3 | 0 | -4 | -3 | 1 | -2 | -3 | -1 | -1 | -1 | -3 | -2 | -3 |
| G | 0 | -3 | 0 | -1 | -3 | -2 | -3 | 8 | -2 | -4 | -4 | -2 | -3 | -4 | -2 | 0 | -2 | -3 | -3 | -4 |
| H | -2 | 0 | 1 | -1 | -3 | 1 | 0 | -2 | 10 | -4 | -3 | 0 | -1 | -1 | -2 | -1 | -2 | -3 | 2 | -4 |
| I | -1 | -4 | -3 | -4 | -2 | -3 | -4 | -4 | -4 | 5 | 2 | -3 | 2 | 0 | -3 | -3 | -1 | -3 | -1 | 4 |
| L | -2 | -3 | -4 | -4 | -2 | -2 | -3 | -4 | -3 | 2 | 5 | -3 | 3 | 1 | -4 | -3 | -1 | -2 | -1 | 1 |
| K | -1 | 3 | 0 | -1 | -3 | 2 | 1 | -2 | 0 | -3 | -3 | 6 | -2 | -4 | -1 | 0 | -1 | -3 | -2 | -3 |
| M | -1 | -2 | -2 | -4 | -2 | 0 | -2 | -3 | -1 | 2 | 3 | -2 | 7 | 0 | -3 | -2 | -1 | -1 | 0 | 1 |
| F | -3 | -3 | -4 | -5 | -2 | -4 | -3 | -4 | -1 | 0 | 1 | -4 | 0 | 8 | -4 | -3 | -2 | 1 | 4 | -1 |
| P | -1 | -3 | -2 | -1 | -4 | -1 | -1 | -2 | -2 | -3 | -4 | -1 | -3 | -4 | 10 | -1 | -1 | -4 | -3 | -3 |
| S | 1 | -1 | 1 | 0 | -1 | 0 | -1 | 0 | -1 | -3 | -3 | 0 | -2 | -3 | -1 | 5 | 2 | -4 | -2 | -2 |
| T | 0 | -1 | 0 | -1 | -1 | -1 | -1 | -2 | -2 | -1 | -1 | -1 | -1 | -2 | -1 | 2 | 5 | -3 | -2 | 0 |
| W | -3 | -3 | -4 | -5 | -5 | -1 | -3 | -3 | -3 | -3 | -2 | -3 | -1 | 1 | -4 | -4 | -3 | 15 | 2 | -3 |
| Y | -2 | -1 | -2 | -3 | -3 | -1 | -2 | -3 | 2 | -1 | -1 | -2 | 0 | 4 | -3 | -2 | -2 | 2 | 8 | -1 |
| V | 0 | -3 | -3 | -4 | -1 | -3 | -3 | -4 | -4 | 4 | 1 | -3 | 1 | -1 | -3 | -2 | 0 | -3 | -1 | 5 |

Using this matrix, **we can score any alignment as the sum of scores of individual pairs of amino acids**.

For example, the top alignment in our earlier example:

```
FlgA1 EAGNVKLKRGRLDTLPPRTVLDINQLVDAISLRDLSPDQPIQLTQFRQAWRVKAGQRVNVIASGD
        ++K+K+GRLDTLPP  +L+ N    A+SLR ++  QP+     R+ W +KAGQ V V+A G+
FlgA2 TLQDIKMKQGRLDTLPPGALLEPNFAQGAVSLRQINAGQPLTRNMLRRLWIIKAGQDVQVLALGE
```

gets the score:

S(FlgA1,FlgA2) = – 1 – 2 – 2 + 2 + 4 + 6 + ... = 186

We also need to penalize **gaps**. For now, let's just use a constant penalty ***d*** for each amino acid gap in an alignment, *i. e.*:

the penalty for a gap of length g = -g*d

## PAM vs. BLOSUM

Margaret Dayhoff (1925-1983)
Developed point accepted mutation matrices = PAM matrices
(& also made the 1 letter aa codes!)

Steve and Jorja Henikoff
Developed BLOSUM matrices

Calibrated for different evolutionary times
PAM-*n* = *n* substitutions per 100 residues
e.g. matrices from PAM1 to PAM250
measure PAM1,
calculate higher PAMs from that

Explicit model of evolution
(calculated using a phylogenetic tree)

Calibrated for different % identity sequences
BLOSUM-*n* = for sequences of about *n* % identity
averages substitution probabilities over sequence clusters, gives better estimates for highly divergent cases

Implicit model of evolution
(calculated from blocks of aligned sequences)

**To align two sequences, we need to perform 3 steps:**

1. We need some way to decide which alignments are better than others.
   For this, we'll invent a way to give the alignments a "score" indicating their quality.

2. **Align the two proteins so that they get the best possible score.**

3. Decide if the score is "good enough" for us to believe the alignment is biologically significant.

**A sense of scale:**

**There are $\binom{2n}{n} \approx \frac{2^{2n}}{\sqrt{\pi n}}$ possible global alignments between two sequences of length n if we use gaps**

**So, with 2 sequences of length 100, that's $> 10^{60}$ possible alignments**

We'll use something called ***dynamic programming.***

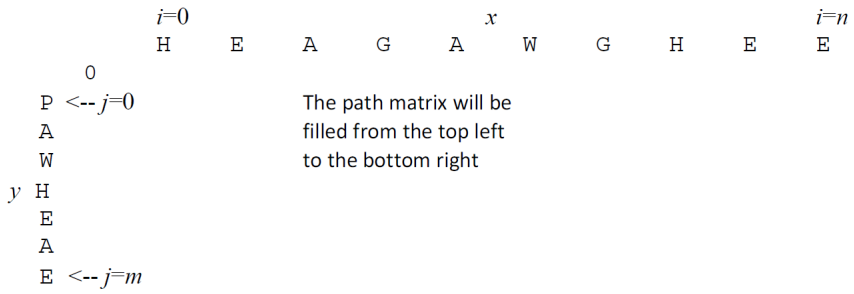
This is **mathematically guaranteed** to find the best scoring alignment, and uses ***recursion.*** This means problems are broken into sub-problems, which are in turn broken into sub-problems, etc, until the simplest sub-problems can be solved.

We're going to find the best ***local*** alignment—the best matching internal alignment—without forcing <u>all</u> of the amino acids to align (i.e. to match ***globally***).

i.e., this → ATGCAT
ATGCAT

Not this → ACGTTATGCATGACGTA
-C---ATGCAT----T-

**Here's the main idea:**
We'll make a ***path matrix***, showing the possible alignments and their scores. There are simple rules for how to fill in the matrix. ***<u>This will test all possible alignments & give us the top-scoring alignment between the two sequences</u>***.



**Here are the rules:**

For a given square in the matrix $F(i,j)$, we look at the squares to its left $F(i\text{-}1,j)$ , top $F(i,j\text{-}1)$ , and top-left $F(i\text{-}1,j\text{-}1)$. Each should have a score.

We consider **3 possible events** & **choose the one scoring the highest**:

(1) $x_i$ is aligned to $y_j$ $\quad\quad$ $F(i\text{-}1,j\text{-}1) + s(x_i,y_j)$

(2) $x_i$ is aligned to a gap $\quad\quad$ $F(i\text{-}1,j) - d$

(3) $y_j$ is aligned to a gap $\quad\quad$ $F(i,j\text{-}1) - d$

For this example, we'll use $d = 8$. We also set the left-most & top-most entries to zero.

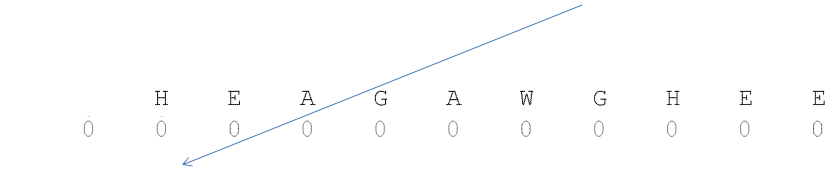
**Just two more rules:**

**If the score is negative, set it equal to zero.**

At each step, we also keep track of which event was chosen by **drawing an arrow from the cell we just filled back to the cell which contributed its score to this one.**

**That's it! Just repeat this to fill the entire matrix.**

Here we go! Start with the borders & the first entry.



| | | H | E | A | G | A | W | G | H | E | E |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| P | 0 | 0 | | | | | | | | | |
| A | 0 | | | | | | | | | | |
| W | 0 | | | | | | | | | | |
| H | 0 | | | | | | | | | | |
| E | 0 | | | | | | | | | | |
| A | 0 | | | | | | | | | | |
| E | 0 | | | | | | | | | | |

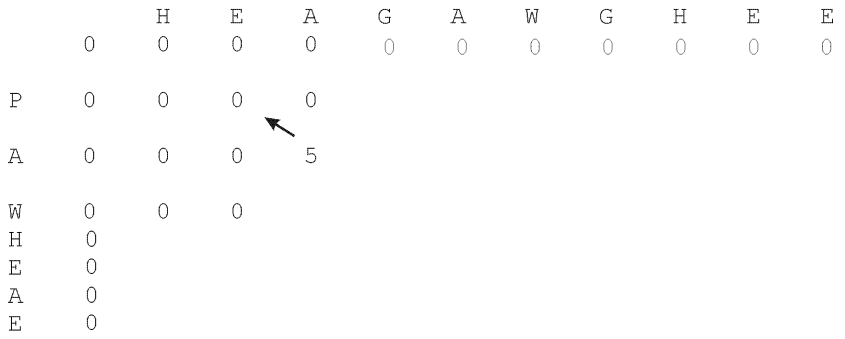
Why is this zero?
What's the score from our BLOSSUM matrix for substituting H for P?

Next round!

| | | 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 | | | | | | | | |
| A | 0 | 0 | 0 | | | | | | | | |
| W | 0 | | | | | | | | | | |
| H | 0 | | | | | | | | | | |
| E | 0 | | | | | | | | | | |
| A | 0 | | | | | | | | | | |
| E | 0 | | | | | | | | | | |

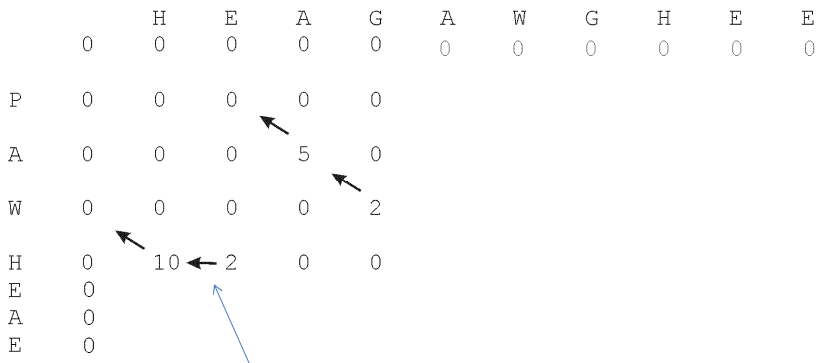
Terrible! Again, none of the possible give positive scores.
We have to go a bit further in before we find a positive score…

A few more rounds, and a positive score at last!



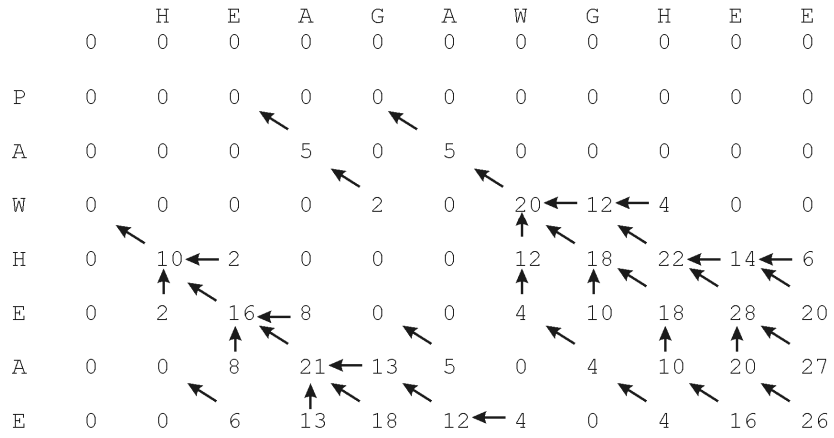

How did we get this one?

& a few more rounds…




What does this mean?

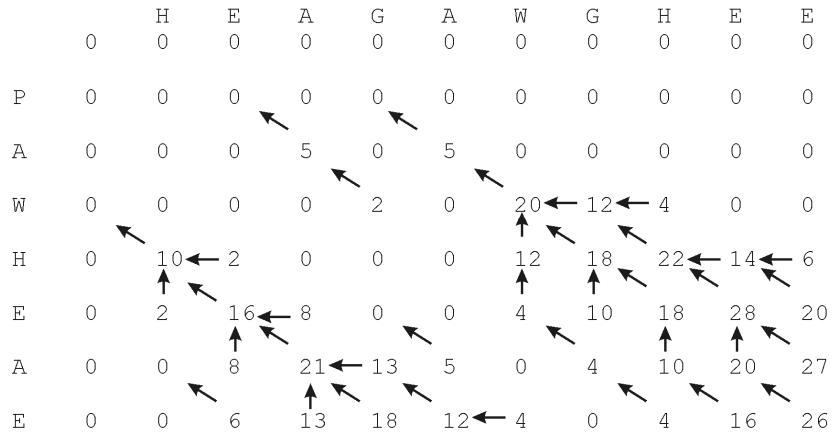
The whole thing filled in!




Now, find the optimal alignment using a ***traceback*** process:

**Look for the highest score, then follow the arrows back.**

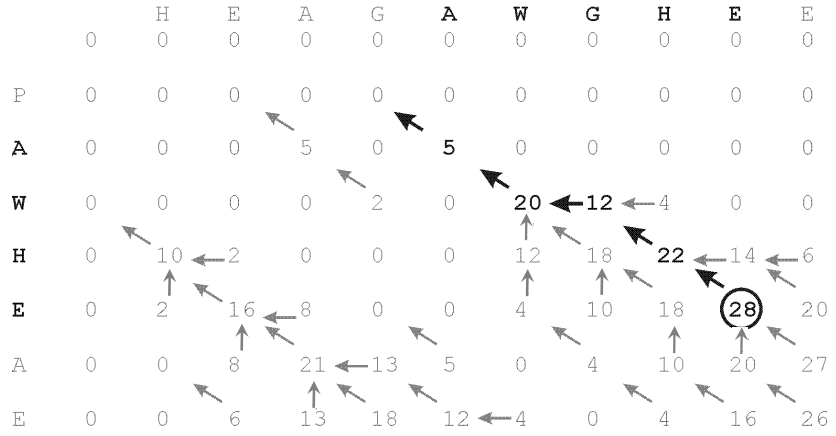
**The alignment "grows" from right to left**



This gives the following alignment:

```
AWGHE
AW-HE
```

(Note: for gaps, the arrow points to the sequence that gets the gap)




**To align two sequences, we need to perform 3 steps:**

1. We need some way to decide which alignments are better than others.
   For this, we'll invent a way to give the alignments a "score" indicating their quality.

2. Align the two proteins so that they get the best possible score.

3. **Decide if the score is "good enough" for us to believe the alignment is biologically significant.**

This algorithm always gives the best alignment.

Every pair of sequences can be aligned in some fashion.

**So, when is a score "good enough"?**

How can we figure this out?

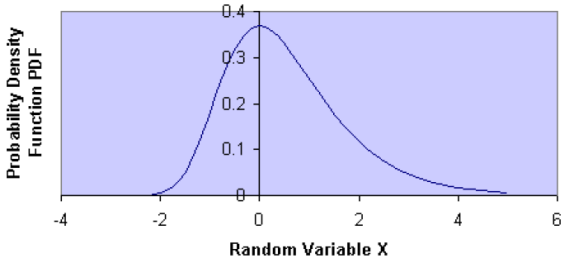
Here's one approach:

**Shuffle one sequence. Calculate the best alignment & its score.**
**Repeat 1000 times.**

If we never see a score as high as the real one, we say the real score has <1 in a 1000 chance of happening just by luck.

But if we want something that only occurs < 1 in a million, we'd have to shuffle 1,000,000 times…

Luckily, alignment scores follow a well-behaved distribution, the ***extreme value distribution***, so we can do a few trials & fit to this.



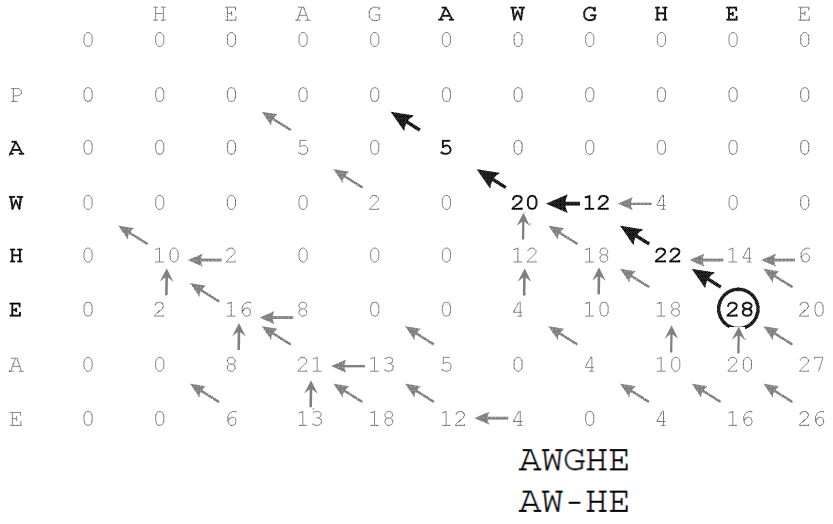

# random trials & their average score

$$p(\text{max score} \le X) \approx e^{-kNe^{\lambda(X-\mu)}}$$

This p-value gives the significance of your alignment. But, if we search a database and perform many alignments, we still need something more (next time).

Describe the shape & can be fit from a few trials

Some extensions: Local vs. global alignments
How might you force the full sequences to align?



## Some extensions: Local vs. global alignments
## How might you force the full sequences to align?

A few tiny changes:

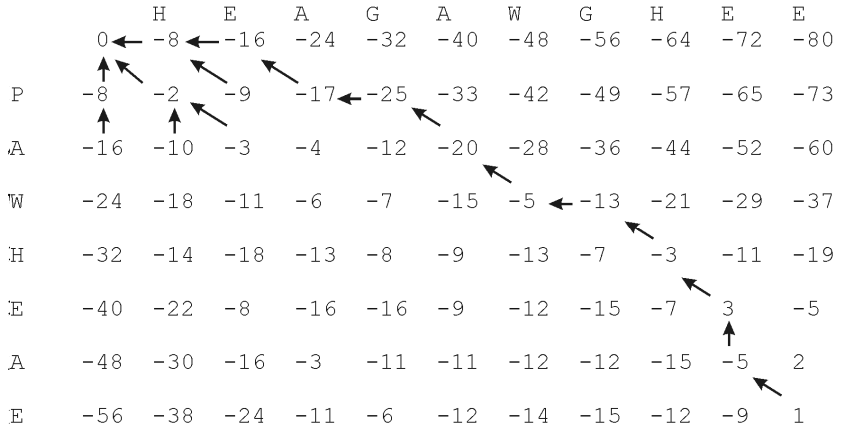
Initialize only the top left cell of the path matrix to zero
(not all top and left cells).

Leave the negative values (don't set them to zero).

The optimal alignment should start at the top left cell and finish at the bottom right cell of the path matrix.

Start the trace-back at the bottom right cell

## Some extensions: Local vs. global alignments
## How might you force the full sequences to align?



# How can you try this yourself using BioPython?

BioPython can perform a wide variety of sequence alignments, DNA/protein, local/global, dynamic programming, BLAST, different scoring schemes, etc, & is a great environment to learn and play with these approaches. Here's a minimal use case to start you off:

```
# Here's how to perform pairwise alignments using BioPython,
# excerpted from https://biopython.org/DIST/docs/tutorial/Tutorial.html

# To generate pairwise alignments, first create a PairwiseAligner object:
from Bio import Align
aligner = Align.PairwiseAligner()    # this will use a very minimal default scoring method
# However, BioPython knows about more sophisticated schemes
# e.g. uncomment the next line to use the BLASTN substitution matrix & gap penalties, which is good for nucleotides:
# aligner = Align.PairwiseAligner(scoring="blastn")
# other options include megablast (for nucs) and blastp (for proteins)

aligner.mode = "local"   # alternatively, use "global" for a global alignment
target = "AGAACTC"
query = "GAACT"
score = aligner.score(target, query)  # Use aligner.score to calculate the alignment score between 2 sequences:
print(score)

alignments = aligner.align(target, query)
for alignment in alignments:
    print(alignment)

# BioPython will perform Smith-Waterman for local alignments, Needleman-Wunsch for global
# you can confirm which algorithm you used by typing:
aligner.algorithm

```

```
5.0
target            1 GAACT 6
                  0 ||||| 5
query             0 GAACT 5

'Smith-Waterman'
```

## Using BioPython, you can change every aspect of the scoring & substitution matrices, as well as run BLAST locally or in the cloud.

e.g. here's the BLOSUM62 matrix, along w/ many others that BioPython knows about:

```
from Bio.Align import substitution_matrices
substitution_matrices.load()
['BENNER22', 'BENNER6', 'BENNER74', 'BLASTN', 'BLASTP', 'BLOSUM45', 'BLOSUM50', 'BLOSUM62', ..., 'TRANS']
matrix = substitution_matrices.load("BLOSUM62")
print(matrix)
```

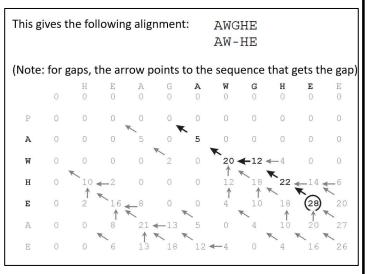
```
#  Matrix made by matblas from blosum62.iij
#  * column uses minimum score
#  BLOSUM Clustered Scoring Matrix in 1/2 Bit Units
#  Blocks Database = /data/blocks_5.0/blocks.dat
#  Cluster Percentage: >= 62
#  Entropy =   0.6979, Expected =  -0.5209
   A   R   N   D   C   Q   E   G   H   I   L   K   M   F   P   S   T   W   Y   V   B   Z   X   *
A  4.0 -1.0 -2.0 -2.0  0.0 -1.0 -1.0  0.0 -2.0 -1.0 -1.0 -1.0 -1.0 -2.0 -1.0  1.0  0.0 -3.0 -2.0  0.0 -2.0 -1.0  0.0 -4.0
R -1.0  5.0  0.0 -2.0 -3.0  1.0  0.0 -2.0  0.0 -3.0 -2.0  2.0 -1.0 -3.0 -2.0 -1.0 -1.0 -3.0 -2.0 -3.0 -1.0  0.0 -1.0 -4.0
N -2.0  0.0  6.0  1.0 -3.0  0.0  0.0  0.0  1.0 -3.0 -3.0  0.0 -2.0 -3.0 -2.0  1.0  0.0 -4.0 -2.0 -3.0  3.0  0.0 -1.0 -4.0
D -2.0 -2.0  1.0  6.0 -3.0  0.0  2.0 -1.0 -1.0 -3.0 -4.0 -1.0 -3.0 -3.0 -1.0  0.0 -1.0 -4.0 -3.0 -3.0  4.0  1.0 -1.0 -4.0
C  0.0 -3.0 -3.0 -3.0  9.0 -3.0 -4.0 -3.0 -3.0 -1.0 -1.0 -3.0 -1.0 -2.0 -3.0 -1.0 -1.0 -2.0 -2.0 -1.0 -3.0 -3.0 -2.0 -4.0
Q -1.0  1.0  0.0  0.0 -3.0  5.0  2.0 -2.0  0.0 -3.0 -2.0  1.0  0.0 -3.0 -1.0  0.0 -1.0 -2.0 -1.0 -2.0  0.0  3.0 -1.0 -4.0
E -1.0  0.0  0.0  2.0 -4.0  2.0  5.0 -2.0  0.0 -3.0 -3.0  1.0 -2.0 -3.0 -1.0  0.0 -1.0 -3.0 -2.0 -2.0  1.0  4.0 -1.0 -4.0
G  0.0 -2.0  0.0 -1.0 -3.0 -2.0 -2.0  6.0 -2.0 -4.0 -4.0 -2.0 -3.0 -3.0 -2.0  0.0 -2.0 -2.0 -3.0 -3.0 -1.0 -2.0 -1.0 -4.0
H -2.0  0.0  1.0 -1.0 -3.0  0.0  0.0 -2.0  8.0 -3.0 -3.0 -1.0 -2.0 -1.0 -2.0 -1.0 -2.0 -2.0  2.0 -3.0  0.0  0.0 -1.0 -4.0
I -1.0 -3.0 -3.0 -3.0 -1.0 -3.0 -3.0 -4.0 -3.0  4.0  2.0 -3.0  1.0  0.0 -3.0 -2.0 -1.0 -3.0 -1.0  3.0 -3.0 -3.0 -1.0 -4.0
L -1.0 -2.0 -3.0 -4.0 -1.0 -2.0 -3.0 -4.0 -3.0  2.0  4.0 -2.0  2.0  0.0 -3.0 -2.0 -1.0 -2.0 -1.0  1.0 -4.0 -3.0 -1.0 -4.0
K -1.0  2.0  0.0 -1.0 -3.0  1.0  1.0 -2.0 -1.0 -3.0 -2.0  5.0 -1.0 -3.0 -1.0  0.0 -1.0 -3.0 -2.0 -2.0  0.0  1.0 -1.0 -4.0
M -1.0 -1.0 -2.0 -3.0 -1.0  0.0 -2.0 -3.0 -2.0  1.0  2.0 -1.0  5.0  0.0 -2.0 -1.0 -1.0 -1.0 -1.0  1.0 -3.0 -1.0 -1.0 -4.0
F -2.0 -3.0 -3.0 -3.0 -2.0 -3.0 -3.0 -3.0 -1.0  0.0  0.0 -3.0  0.0  6.0 -4.0 -2.0 -2.0  1.0  3.0 -1.0 -3.0 -3.0 -1.0 -4.0
P -1.0 -2.0 -2.0 -1.0 -3.0 -1.0 -1.0 -2.0 -2.0 -3.0 -3.0 -1.0 -2.0 -4.0  7.0 -1.0 -1.0 -4.0 -3.0 -2.0 -2.0 -1.0 -2.0 -4.0
S  1.0 -1.0  1.0  0.0 -1.0  0.0  0.0  0.0 -1.0 -2.0 -2.0  0.0 -1.0 -2.0 -1.0  4.0  1.0 -3.0 -2.0 -2.0  0.0  0.0  0.0 -4.0
T  0.0 -1.0  0.0 -1.0 -1.0 -1.0 -1.0 -2.0 -2.0 -1.0 -1.0 -1.0 -1.0 -2.0 -1.0  1.0  5.0 -2.0 -2.0  0.0 -1.0 -1.0  0.0 -4.0
W -3.0 -3.0 -4.0 -4.0 -2.0 -2.0 -3.0 -2.0 -2.0 -3.0 -2.0 -3.0 -1.0  1.0 -4.0 -3.0 -2.0 11.0  2.0 -3.0 -4.0 -3.0 -2.0 -4.0
Y -2.0 -2.0 -2.0 -3.0 -2.0 -1.0 -2.0 -3.0  2.0 -1.0 -1.0 -2.0 -1.0  3.0 -3.0 -2.0 -2.0  2.0  7.0 -1.0 -3.0 -2.0 -1.0 -4.0
V  0.0 -3.0 -3.0 -3.0 -1.0 -2.0 -2.0 -3.0 -3.0  3.0  1.0 -2.0  1.0 -1.0 -2.0 -2.0  0.0 -3.0 -1.0  4.0 -3.0 -2.0 -1.0 -4.0
B -2.0 -1.0  3.0  4.0 -3.0  0.0  1.0 -1.0  0.0 -3.0 -4.0  0.0 -3.0 -3.0 -2.0  0.0 -1.0 -4.0 -3.0 -3.0  4.0  1.0 -1.0 -4.0
Z -1.0  0.0  0.0  1.0 -3.0  3.0  4.0 -2.0  0.0 -3.0 -3.0  1.0 -1.0 -3.0 -1.0  0.0 -1.0 -3.0 -2.0 -2.0  1.0  4.0 -1.0 -4.0
X  0.0 -1.0 -1.0 -1.0 -2.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -1.0 -2.0  0.0  0.0 -2.0 -1.0 -1.0 -1.0 -1.0 -1.0 -4.0
* -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0 -4.0  1.0
```

Putting it all together, here's the example alignment we did manually

```python
from Bio import Align
from Bio.Align import substitution_matrices

aligner = Align.PairwiseAligner()
aligner.mode = "local"
matrix = substitution_matrices.load("BLOSUM50")

aligner.substitution_matrix = matrix
aligner.target_internal_open_gap_score = -8.000000
aligner.target_internal_extend_gap_score = -8.000000
aligner.target_left_open_gap_score = -8.000000
aligner.target_left_extend_gap_score = -8.000000
aligner.target_right_open_gap_score = -8.000000
aligner.target_right_extend_gap_score = -8.000000
aligner.query_internal_open_gap_score = -8.000000
aligner.query_internal_extend_gap_score = -8.000000
aligner.query_left_open_gap_score = -8.000000
aligner.query_left_extend_gap_score = -8.000000
aligner.query_right_open_gap_score = -8.000000
aligner.query_right_extend_gap_score = -8.000000

target = "HEAGAWGHEE"
query = "PAWHEAE"
score = aligner.score(target, query)
print(score)

alignments = aligner.align(target, query)
for alignment in alignments:
    print(alignment)
```

```
28.0
target            4 AWGHE 9
                  0 ||-|| 5
query             1 AW-HE 5
```

Here was our earlier version:

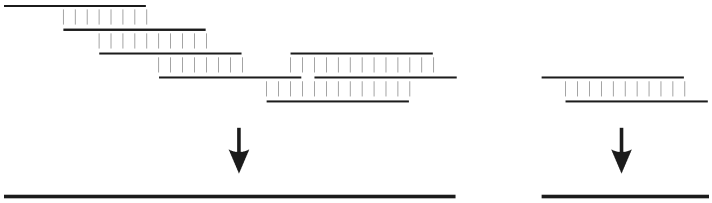



**You can read more about using BioPython for sequence analyses & get example code at:**

**https://biopython.org/DIST/docs/tutorial/Tutorial.html**

**Chapter 7 is all about how to perform pairwise sequence alignments**

Some extensions:
What about overlapping sequences?
e.g. as in 'shotgun sequencing' genomes where 'contigs' are built up from overlapping sequences




Some extensions:
What about overlapping sequences?

Modify global alignment to <u>not penalize overhangs:</u>

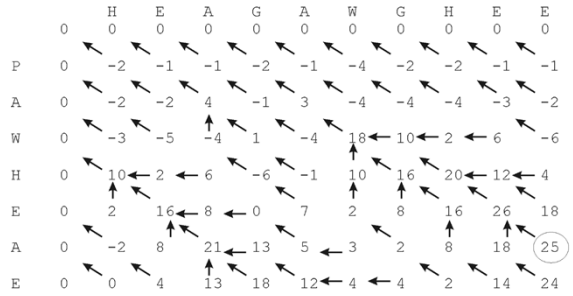
The optimal alignment should start at the <u>top</u> or <u>left</u> edge and finish at the <u>bottom</u> or <u>right</u> edge of the path matrix.

Set these boundary conditions :

$F(i,0) = 0$ for $i$=1 to $n$
$F(0,j) = 0$ for $j$=1 to $m$

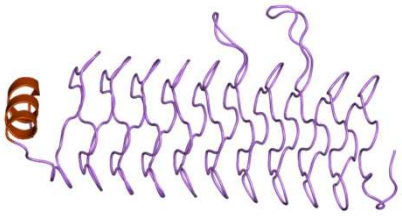
Start the traceback at the cell with the highest score on the <u>right</u> or <u>bottom</u> border

Some extensions:
What about overlapping sequences?
e.g. as in ‘shotgun sequencing’ genomes where
‘contigs’ are built up from overlapping sequences



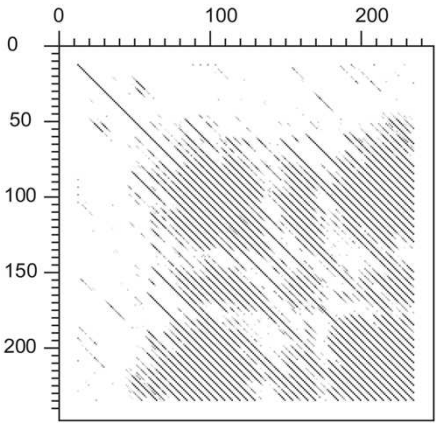

**(overhang = HEA)** **GAWGHEE**
**PAW-HEA** **(overhang = E)**

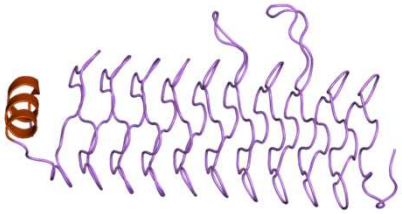
Some extensions:
How might you find repetitive sequences?



Structure of the pentapeptide repeat protein HetL
(from wiki, PMID18952182)

Align the sequence to itself and ignore the diagonal (optimal) alignment
→ High-scoring off-diagonal alignments will be repeats



Structure of the pentapeptide repeat protein HetL
(from wiki, PMID18952182)

Dot plot (quick visualization of sequence similarity) of the pentapeptide repeat protein HglK protein vs. itself
(http://en.wikipedia.org/wiki/Pentapeptide_repeat)