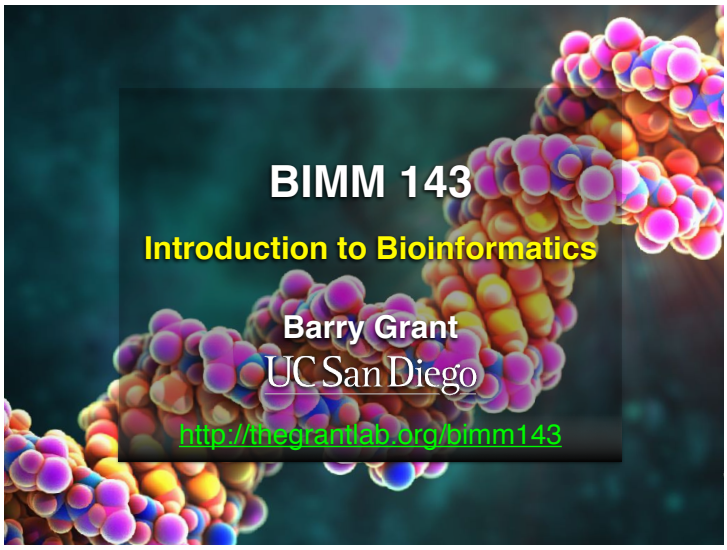


Recap From Last Time:

- Bioinformatics is computer aided biology.
  - Deals with the collection, archiving, organization, and interpretation of a wide range of biological data.
- There are a large number of bioinformatics databases (see handout!).
- The NCBI and EBI are major online bioinformatics service providers.
- Introduced via **hands-on session** the BLAST, Entrez, GENE, OMIM, UniProt, Muscle and PDB bioinformatics tools and databases.
  - Muddy point assessment (see results)
- Also covered: Course structure; Supporting course website, Ethics code, and Introductions…

Today's Menu

Classifying Databases	Primary, secondary and composite Bioinformatics databases
Using Databases	**Vignette** demonstrating how major Bioinformatics databases intersect
Major Biomolecular Formats	How nucleotide and protein sequence and structure data are represented
Alignment Foundations	**Introducing the *why* and *how* of comparing sequences**
Alignment Algorithms	**Hands-on** exploration of alignment algorithms and applications

Primary, secondary & composite databases

Bioinformatics databases can be usefully classified into *primary*, *secondary* and *composite* according to their data source.

- **Primary databases** (or *archival databases*) consist of data derived experimentally.
  - **GenBank**: NCBI's primary nucleotide sequence database.
  - **PDB:** Protein X-ray crystal and NMR structures.
- **Secondary databases** (or *derived databases*) contain information derived from a primary database.
  - **RefSeq**: non redundant set of curated reference sequences primarily from GenBank
  - **PFAM**: protein sequence families primarily from UniProt and PDB
- **Composite databases** (or *metadatabases*) join a variety of different primary and secondary database sources.
  - **OMIM**: catalog of human genes, genetic disorders and related literature
  - **GENE**: molecular data and literature related to genes with extensive links to other databases.

DATABASE VIGNETTE

You have just come out a seminar about gastric cancer and one of your co-workers asks:

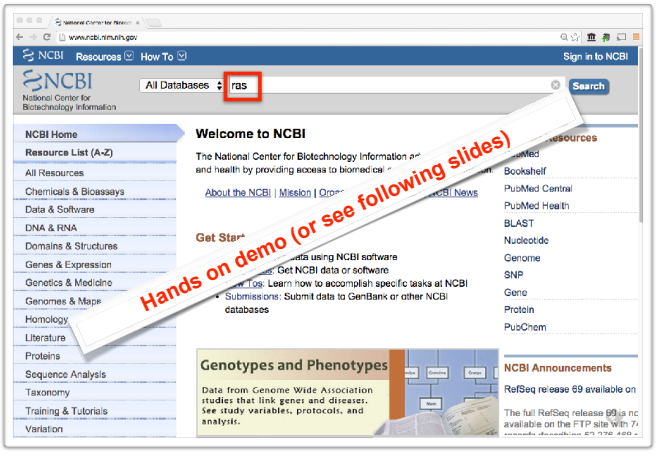
> *"What do you know about that 'Kras' gene the speaker kept taking about?"*

You have some recollection about hearing of 'Ras' before. How would you find out more?

- Google?
- Library?
- **Bioinformatics databases at NCBI and EBI!**

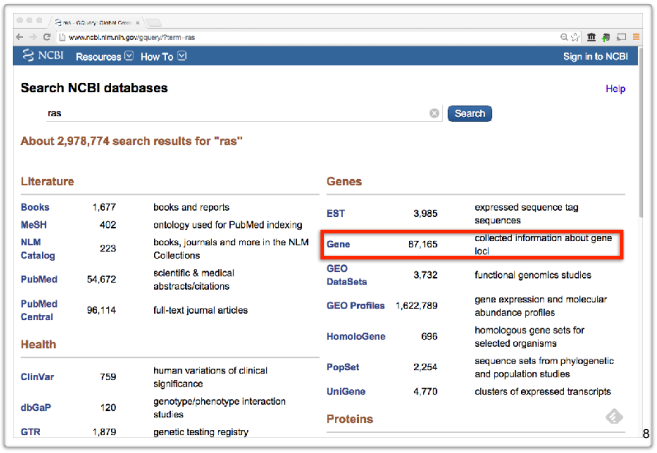
http://www.ncbi.nlm.nih.gov/

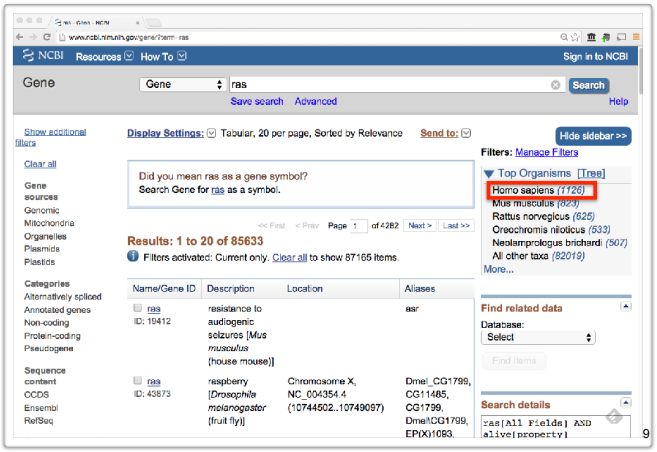
http://www.ncbi.nlm.nih.gov/

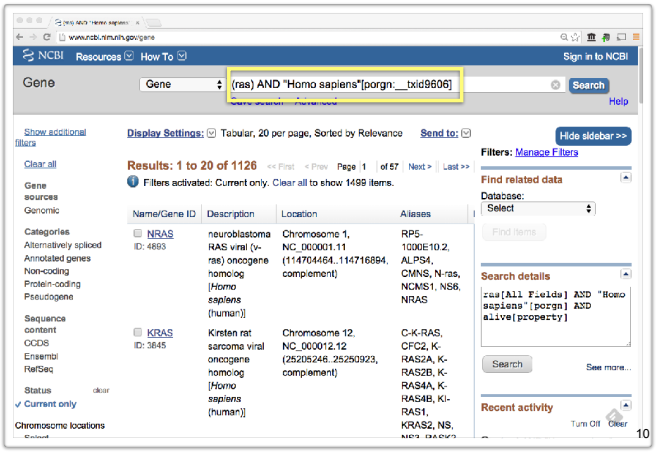


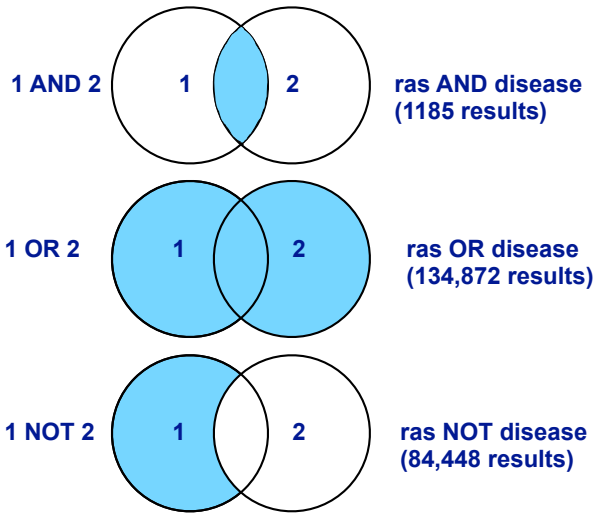
Example Vignette Questions:

- What chromosome location and what genes are in the vicinity of a given query gene? NCBI **GENE**
- What can you find out about molecular functions, biological processes, and prominent cellular locations? EBI **GO**
- What amino acid positions in the protein are responsible for ligand binding? EBI **UniProt**
- What variants of this gene are associated with gastric cancer and other human diseases? NCBI **OMIN**
- What is known about the protein family, its species distribution, number in humans and residue-wise conservation? EBI **PFAM**
- Are high resolution protein structures available to examine the details of these mutations? How might we explain their potential molecular effects? RCSB **PDB**

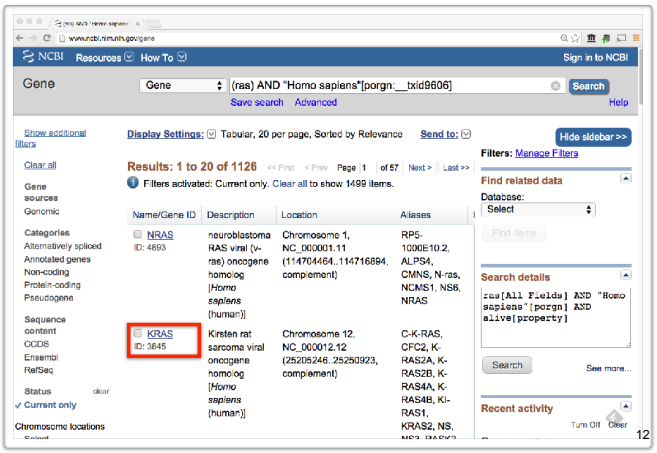




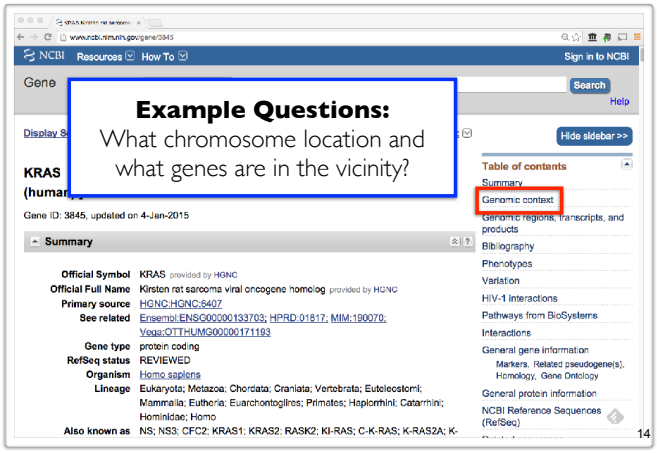




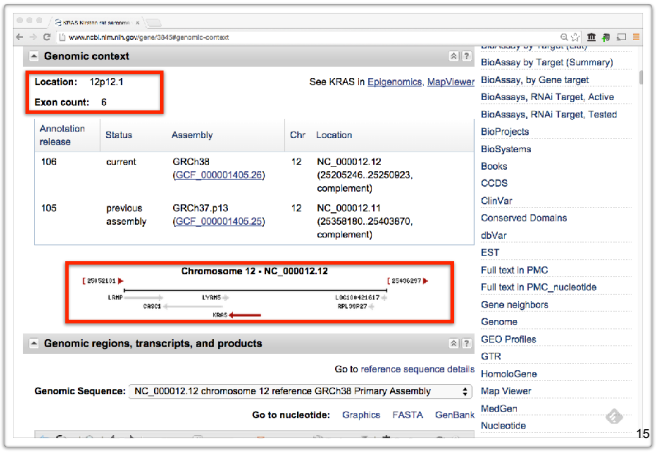
11



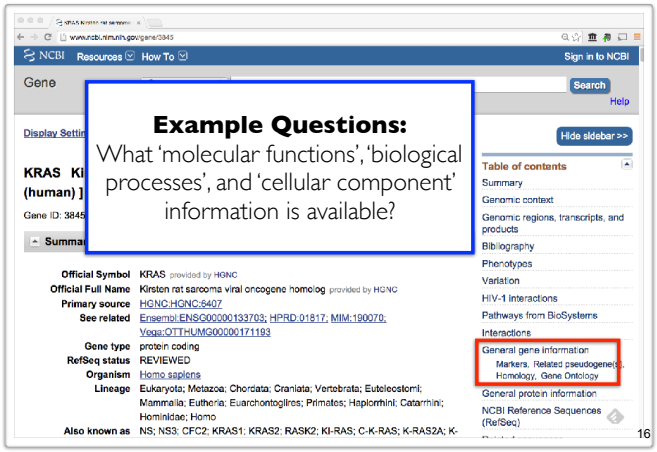
13



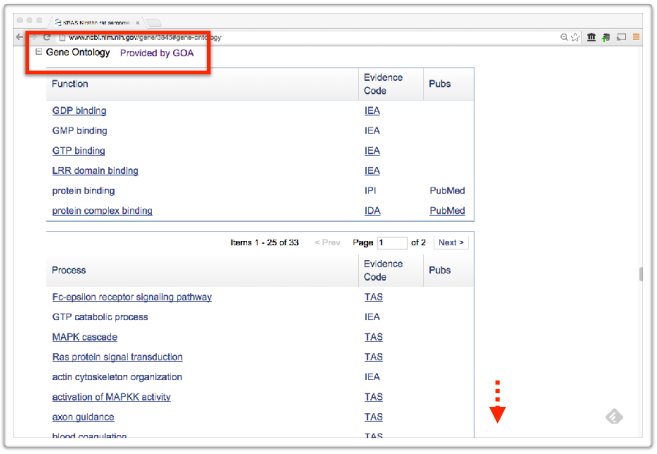
14



15

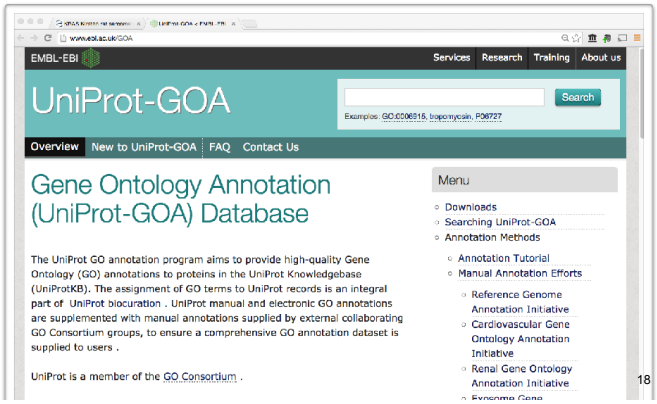


16



GO: Gene Ontology

GO provides a controlled vocabulary of terms for describing gene product characteristics and gene product annotation data



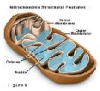
18

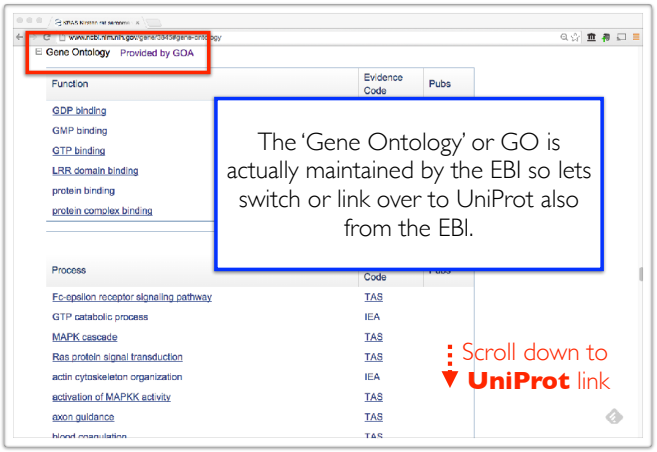
Why do we need Ontologies?

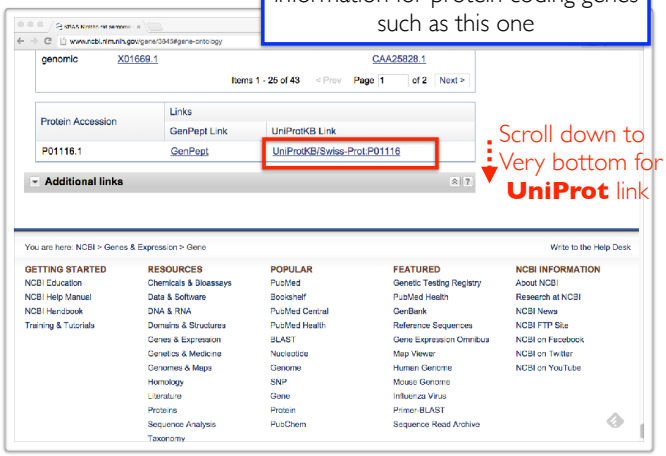
- Annotation is essential for capturing the understanding and knowledge associated with a sequence or other molecular entity
- Annotation is traditionally recorded as “free text”, which is easy to read by humans, but has a number of disadvantages, including:
  - Difficult for computers to parse
  - Quality varies from database to database
  - Terminology used varies from annotator to annotator
- Ontologies are annotations using standard vocabularies that try to address these issues
- GO is integrated with UniProt and many other databases including a number at NCBI

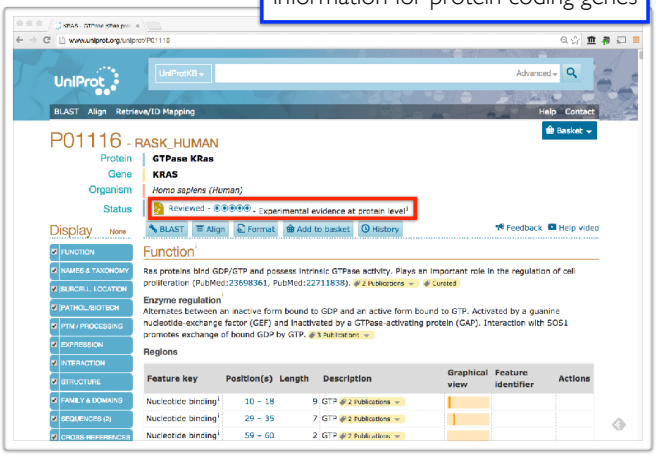
GO Ontologies

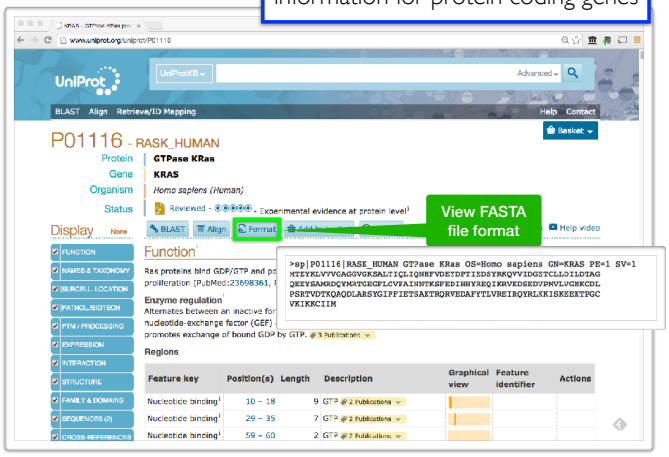
- There are three ontologies in GO:
  - **Biological <u>Process</u>**
    A commonly recognized series of events
    e.g. cell division, mitosis,
  - **Molecular <u>Function</u>**
    An elemental activity, task or job
    e.g. kinase activity, insulin binding
  - **Cellular <u>Component</u>**
    Where a gene product is located
    e.g. mitochondrion, mitochondrial membrane



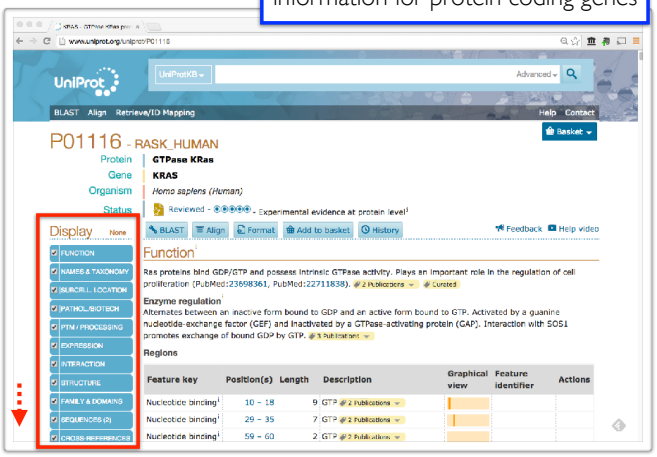




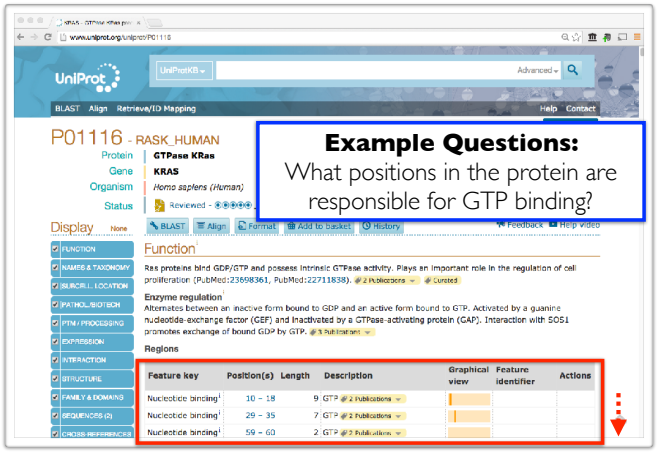




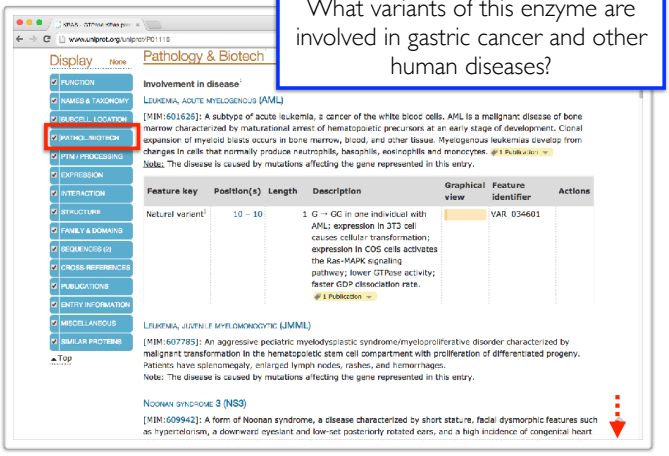
UniProt will detail much more information for protein coding genes



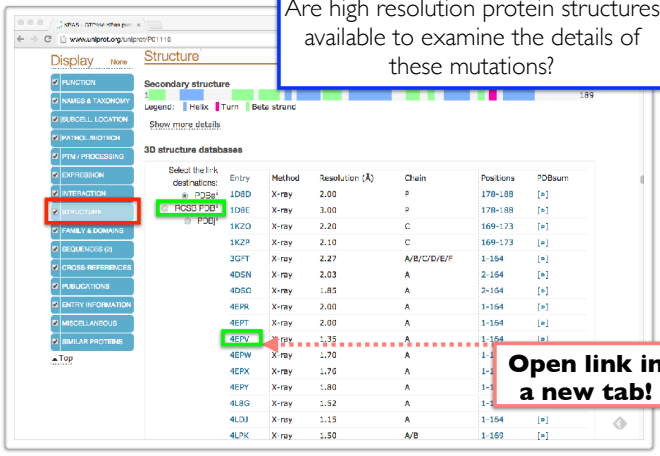
Example Questions:
What positions in the protein are responsible for GTP binding?



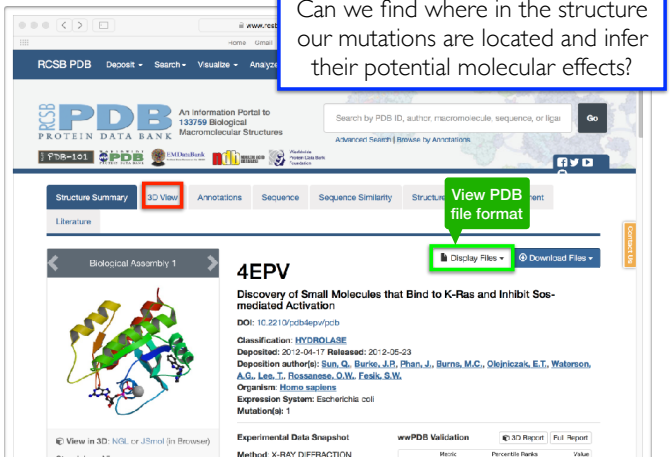
Example Questions:
What variants of this enzyme are involved in gastric cancer and other human diseases?



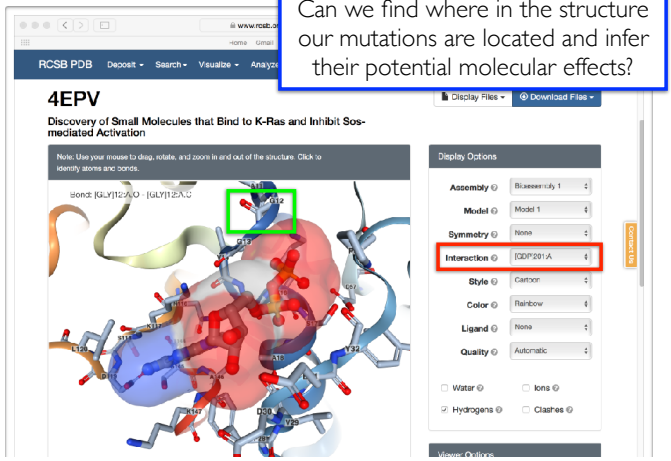
Example Questions:
Are high resolution protein structures available to examine the details of these mutations?



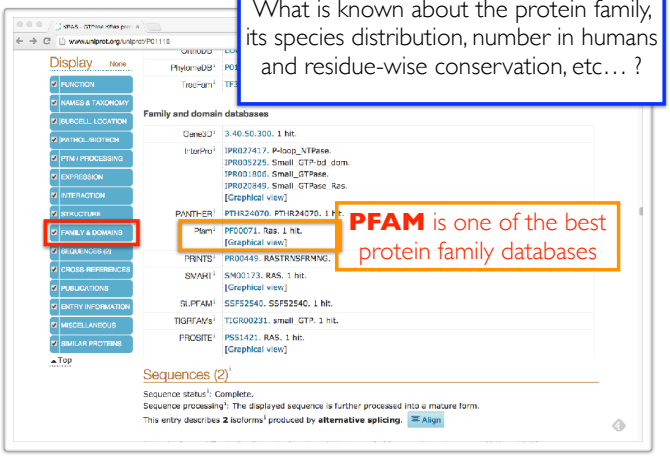
Lets view the 3D structure:
Can we find where in the structure our mutations are located and infer their potential molecular effects?



Lets view the 3D structure:
Can we find where in the structure our mutations are located and infer their potential molecular effects?

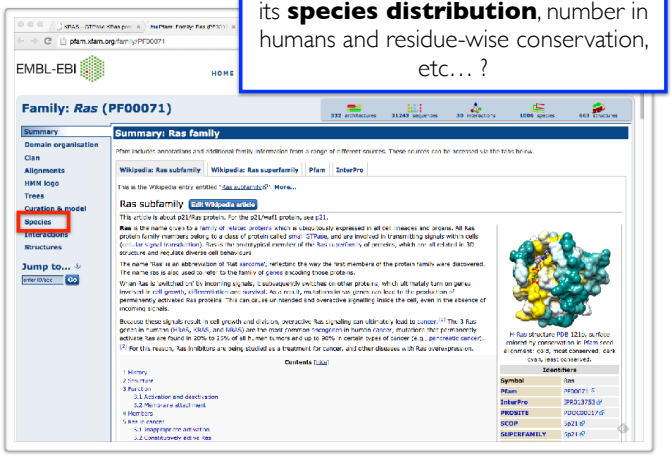


Back to UniProt:
What is known about the protein family, its species distribution, number in humans and residue-wise conservation, etc… ?

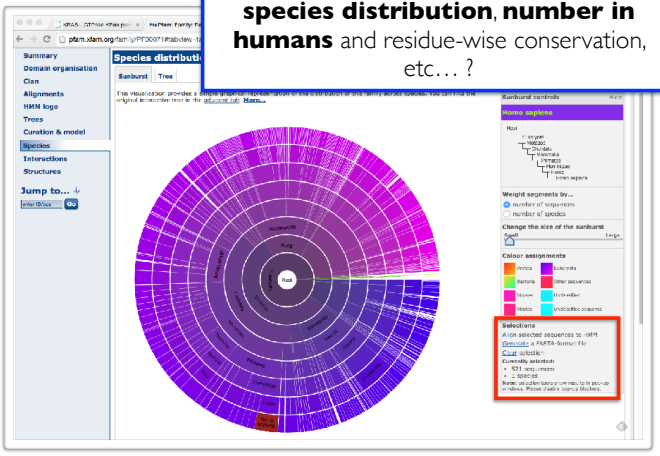


PFAM is one of the best protein family databases

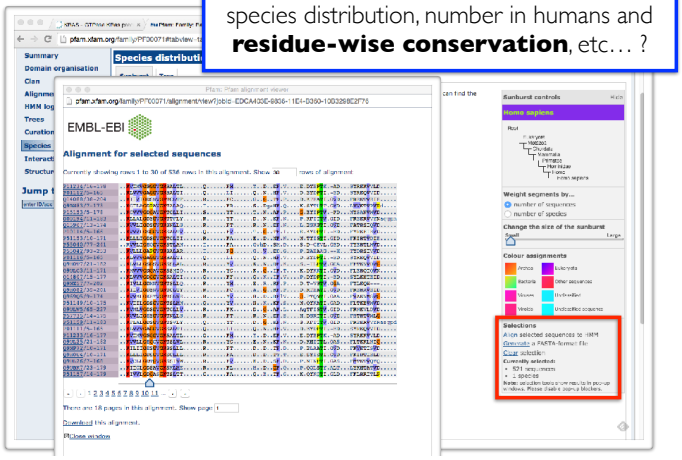
Example Questions:
What is known about the protein family, its **species distribution**, number in humans and residue-wise conservation, etc… ?



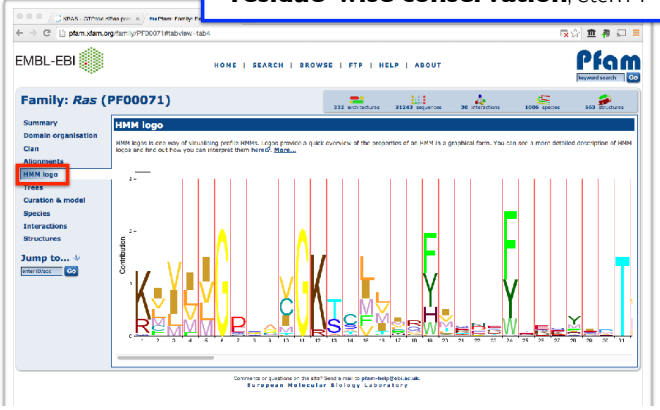
Example Questions:
What is known about the protein family, its **species distribution**, **number in humans** and residue-wise conservation, etc… ?

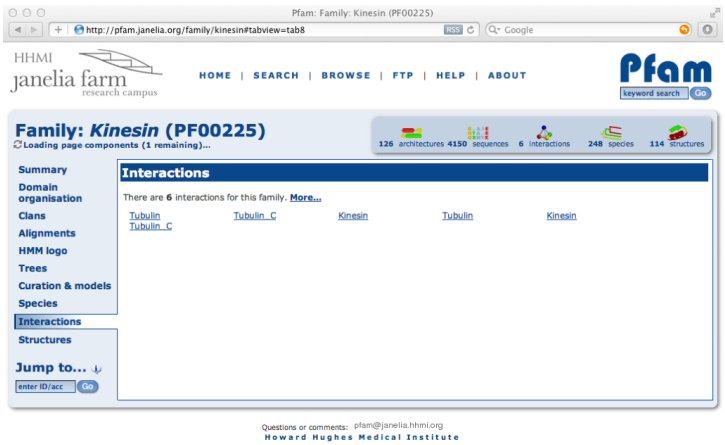


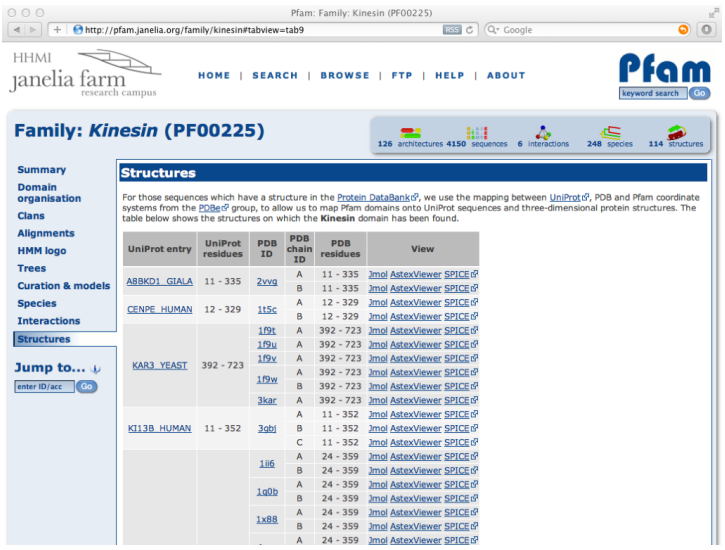
Example Questions:
What is known about the protein family, its species distribution, number in humans and **residue-wise conservation**, etc… ?

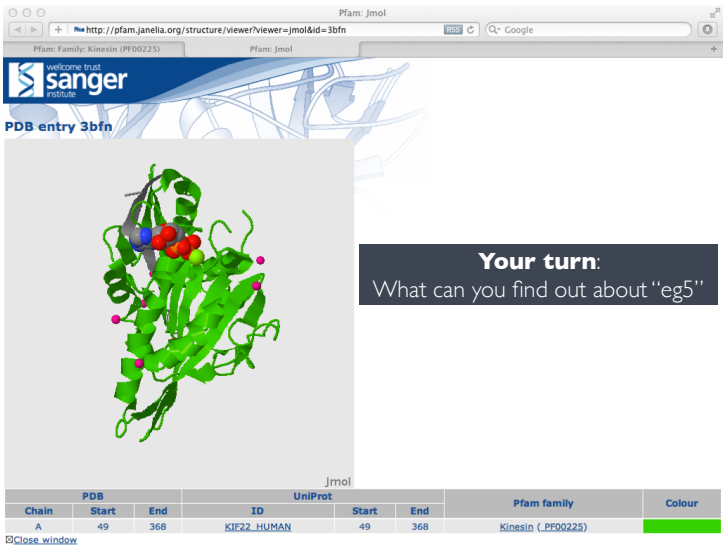


Example Questions:
What is known about the protein family, its species distribution, number in humans and **residue-wise conservation**, etc… ?









Today's Menu

Classifying Databases	Primary, secondary and composite Bioinformatics databases
Using Databases	**Vignette** demonstrating how major Bioinformatics databases intersect
Major Biomolecular Formats	How nucleotide and protein sequence and structure data are represented
Alignment Foundations	**Introducing the *why* and *how* of comparing sequences**
Alignment Algorithms	**Hands-on** exploration of alignment algorithms and applications

ALIGNMENT FOUNDATIONS

- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

ALIGNMENT FOUNDATIONS

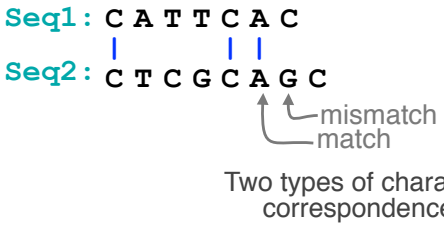
- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

Basic Idea: Display one sequence above another with spaces (termed **gaps**) inserted in both to reveal **similarity** of nucleotides or amino acids.

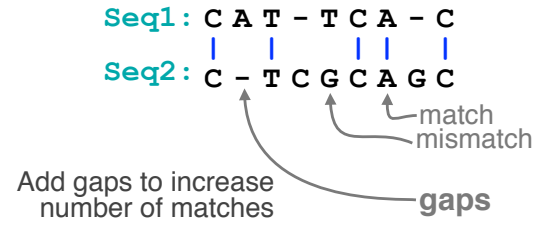
```
Seq1: C A T T C A C
Seq2: C T C G C A G C
```

[Screencast Material]

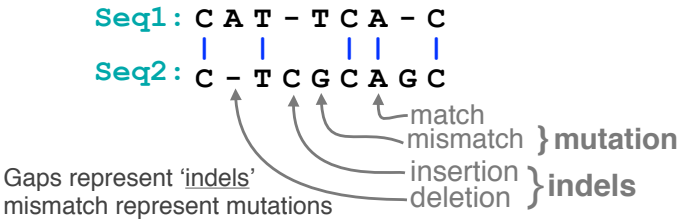
Basic Idea: Display one sequence above another with spaces (termed **gaps**) inserted in both to reveal **similarity** of nucleotides or amino acids.



Basic Idea: Display one sequence above another with spaces (termed **gaps**) inserted in both to reveal **similarity** of nucleotides or amino acids.



Basic Idea: Display one sequence above another with spaces (termed **gaps**) inserted in both to reveal **similarity** of nucleotides or amino acids.



Why compare biological sequences?

- To obtain **functional or mechanistic insight** about a sequence by inference from another potentially better characterized sequence
- To find whether two (or more) genes or proteins are **evolutionarily related**
- To find **structurally or functionally similar regions** within sequences (e.g. catalytic sites, binding sites for other molecules, etc.)
- Many practical bioinformatics applications…

Practical applications include...

- **Similarity searching of databases**
  - Protein structure prediction, annotation, etc...
- **Assembly of sequence reads** into a longer construct such as a genomic sequence
- **Mapping sequencing reads to a known genome**
  - "Resequencing", looking for differences from reference genome - SNPs, indels (insertions or deletions)
  - Mapping transcription factor binding sites via ChIP-Seq (chromatin immuno-precipitation sequencing)
  - Pretty much all next-gen sequencing data analysis

Practical applications include...

- **Similarity searching of databases**
  - Protein structure prediction
- **Assembly of sequen**
  construct such
- **Mappi** **to a known genome**
  - ing for differences from reference
  s, indels (insertions or deletions)
  - ing transcription factor binding sites via ChIP-Seq (chromatin immuno-precipitation sequencing)
  - Pretty much all next-gen sequencing data analysis

N.B. Pairwise sequence alignment is arguably the most fundamental operation of bioinformatics!

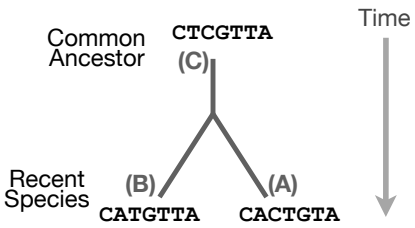
ALIGNMENT FOUNDATIONS

- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

Sequence changes during evolution

There are three major types of sequence change that can occur during evolution.

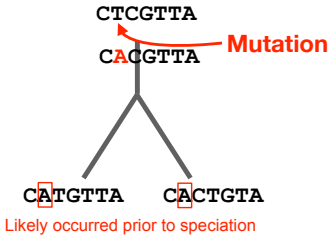
- Mutations/Substitutions
- Deletions
- Insertions



Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.

- **Mutations/Substitutions** CTCGTTA → CACGTTA
- Deletions
- Insertions

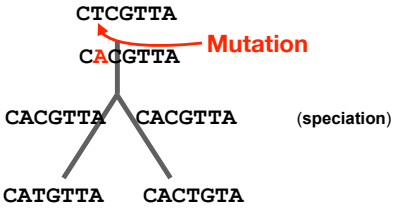


Likely occurred prior to speciation

Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.

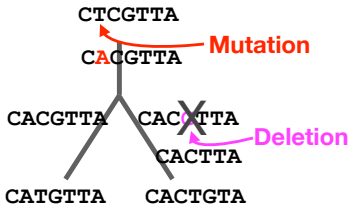
- Mutations/Substitutions CTCGTTA → CACGTTA
- Deletions
- Insertions



Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.

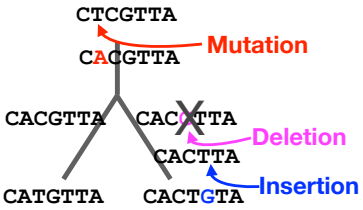
- Mutations/Substitutions CTCGTTA → CACGTTA
- **Deletions** CACGTTA → CACTTA
- Insertions



Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.

- Mutations/Substitutions CTCGTTA → CACGTTA
- Deletions CACGTTA → CACTTA
- **Insertions** CACTTA → CACTGTA

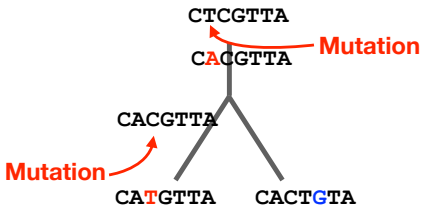


Mutations, deletions and insertions

There are three major types of sequence change that can occur during evolution.

- **Mutations/Substitutions**
- Deletions
- Insertions

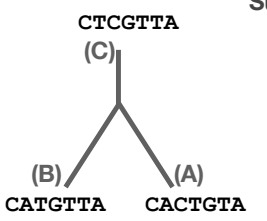
CTCGTTA → CACGTTA
CACGTTA → CATGTTA

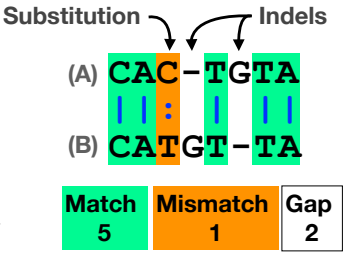


Alignment view

Alignments are great tools to visualize sequence similarity and evolutionary changes in homologous sequences.

- **Mismatches** represent mutations/substitutions
- **Gaps** represent insertions and deletions (indels)

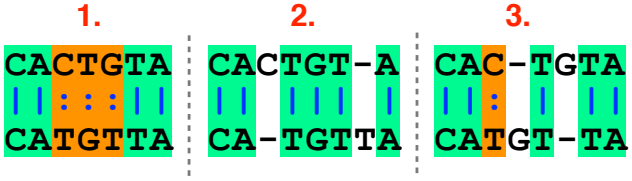




Alternative alignments

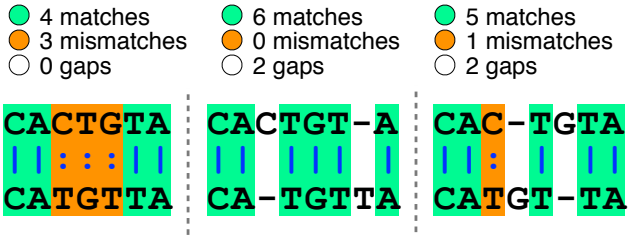
- Unfortunately, finding the correct alignment is difficult if we do not know the evolutionary history of the two sequences

Q. Which of these 3 possible alignments is best?



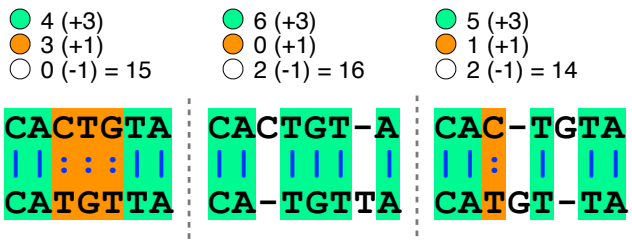
Alternative alignments

- One way to judge alignments is to compare their number of matches, insertions, deletions and mutations



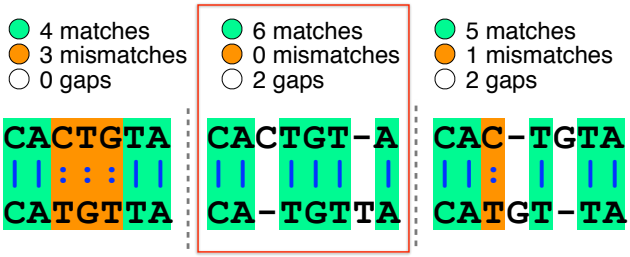
Scoring alignments

- We can assign a score for each match (+3), mismatch (+1) and indel (-1) to identify the **optimal alignment** *for this scoring scheme*



Optimal alignments

- Biologists often prefer **parsimonious alignments**, where the number of postulated sequence changes is minimized.



Optimal alignments

- Biologists often prefer **parsimonious alignments**, where the number of postulated sequence changes is minimized.

4 matches
3 mismatches
0 gaps

```
CACTGTA
||:::||
CATGTTA
```

6 matches
0 mismatches
2 gaps

```
CACTGT-A
||  ||| |
CA-TGTTA
```

5 matches
1 mismatches
2 gaps

```
CAC-TGTA
||:  | ||
CATGT-TA
```

Optimal alignments

- Biologists often prefer **parsimonious alignments**, where the number of postulated sequence changes is minimized.

4 matches
3 mismatches
0 gaps

```
CACTGTA
||:::||
CATGTTA
```

6 matches
0 mismatches
2 gaps

```
CACTG-TA
||  | ||
CA-TGTTA
```

5 matches
1 mismatches
2 gaps

```
CAC-TGTA
||:  | ||
CATGT-TA
```

Optimal alignments

- Biologists often prefer **parsimonious alignments**, where the numb… sequence changes is …

Warning: There may be more than one optimal alignment and these may not reflect the true evolutionary history of our sequences!

4 matches
3 mis…

…matches
1 mismatches
2 gaps

```
CACTGT-A
||  ||| |
CA-TGTTA
```

```
CAC-TGTA
||:  | ||
CATGT-TA
```

ALIGNMENT FOUNDATIONS

- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

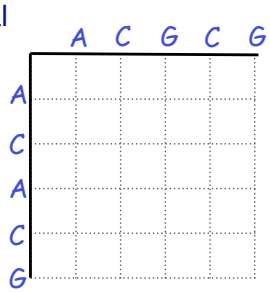
ALIGNMENT FOUNDATIONS

- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - D…
  - B…

How do we compute the optimal alignment between two sequences?

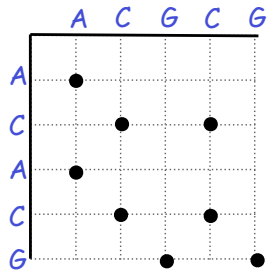
Dot plots: simple graphical approach

- Place one sequence on the vertical axis of a 2D grid (or matrix) and the other on the horizontal



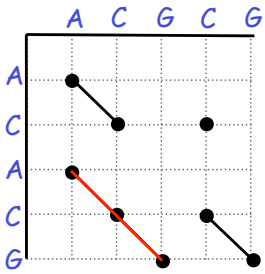
Dot plots: simple graphical approach

- Now simply put dots where the horizontal and vertical sequence values match



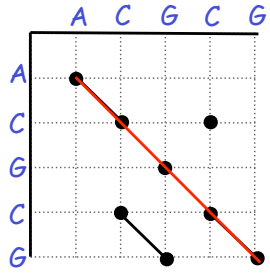
Dot plots: simple graphical approach

- Diagonal runs of dots indicate matched segments of sequence



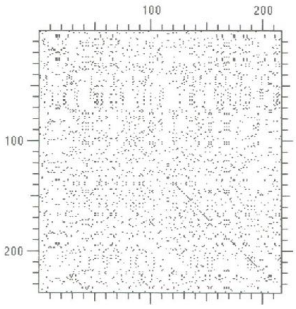
Dot plots: simple graphical approach

Q. What would the dot matrix of a two identical sequences look like?



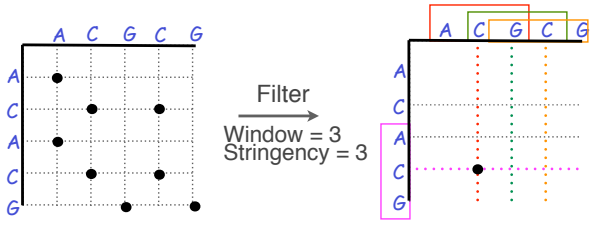
Dot plots: simple graphical approach

- Dot matrices for long sequences can be noisy



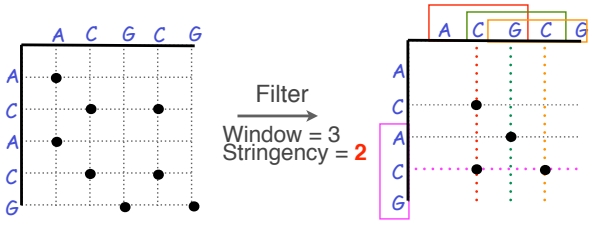
Dot plots: window size and match stringency

Solution: use a window and a threshold
– compare character by character within a window
– require certain fraction of matches within window in order to display it with a dot.
- You have to choose window size and stringency

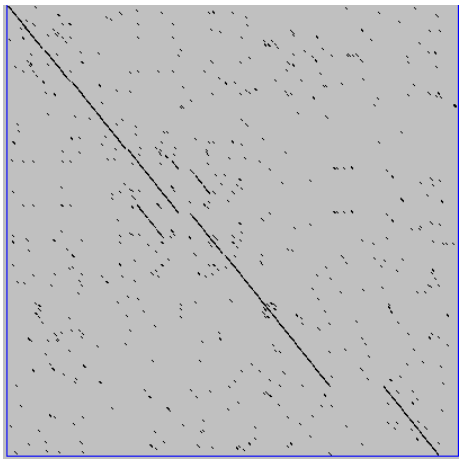


Dot plots: window size and match stringency

Solution: use a window and a threshold
– compare character by character within a window
– require certain fraction of matches within window in order to display it with a dot.
- You have to choose window size and stringency



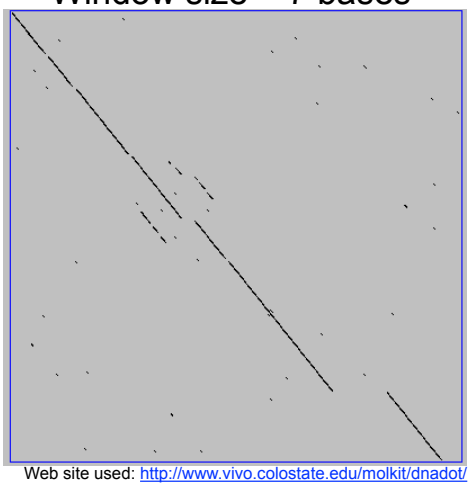
Window size = 5 bases



A dot plot simply puts a dot where two sequences match. In this example, dots are placed in the plot if 5 bases in a row match perfectly. Requiring a 5 base perfect match is a **heuristic** – only look at regions that have a certain degree of identity.

Do you expect evolutionarily related sequences to have more word matches (matches in a row over a certain length) than random or unrelated sequences?

Window size = 7 bases



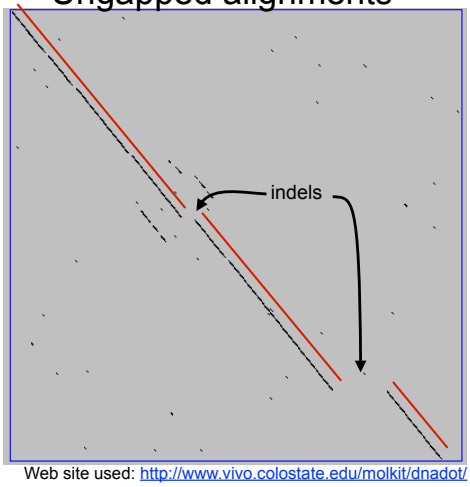
This is a dot plot of the same sequence pair. Now 7 bases in a row must match for a dot to be place. Noise is reduced.

Using windows of a certain length is very similar to using words (kmers) of N characters in the heuristic alignment search tools

Bigger window (kmer) fewer matches to consider

Web site used: http://www.vivo.colostate.edu/molkit/dnadot/

Ungapped alignments



Only **diagonals** can be followed.

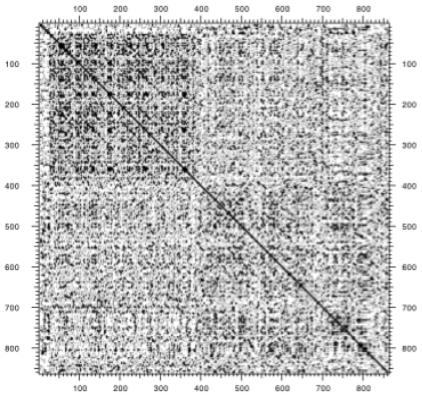
Downward or rightward paths represent **insertion** or **deletions** (gaps in one sequence or the other).

Web site used: http://www.vivo.colostate.edu/molkit/dnadot/

Uses for dot matrices

- Visually assessing the similarity of two protein or two nucleic acid sequences
- Finding local repeat sequences within a larger sequence by comparing a sequence to itself
  - Repeats appear as a set of diagonal runs stacked vertically and/or horizontally

Repeats

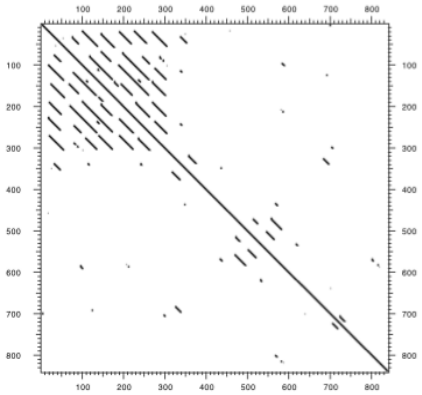


Human LDL receptor protein sequence (Genbank P01130)

W = 1
S = 1

(Figure from Mount, "Bioinformatics sequence and genome analysis")

Repeats



Human LDL receptor protein sequence (Genbank P01130)

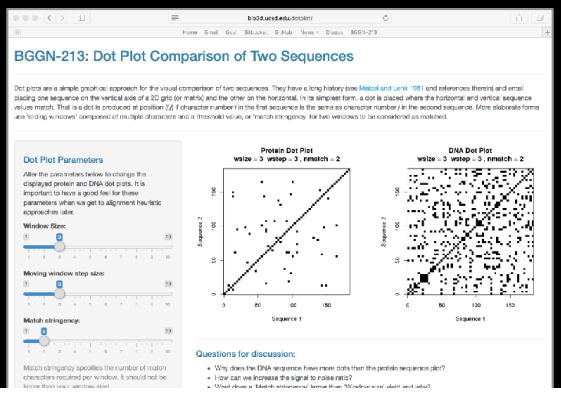
W = 23
S = 7

(Figure from Mount, "Bioinformatics sequence and genome analysis")

Your Turn!

Exploration of dot plot parameters (hands-on worksheet **Section 1**)

http://bio3d.ucsd.edu/dotplot/ https://bioboot.shinyapps.io/dotplot/



ALIGNMENT FOUNDATIONS

- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

The Dynamic Programming Algorithm

- The dynamic programming algorithm can be thought of an extension to the dot plot approach
  - One sequence is placed down the side of a grid and another across the top
  - Instead of placing a dot in the grid, we **compute a score** for each position
  - Finding the optimal alignment corresponds to finding the path through the grid with the **best possible score**

Needleman, S.B. & Wunsch, C.D. (1970) "A general method applicable to the search for similarities in the amino acid sequences of two proteins." J. Mol. Biol. 48:443-453.

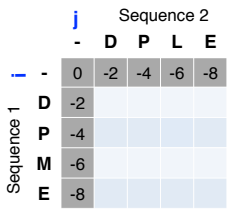
Algorithm of **Needleman and Wunsch**

- The Needleman–Wunsch approach to global sequence alignment has three basic steps:
  (1) setting up a 2D-grid (or **alignment matrix**),
  (2) **scoring the matrix**, and
  (3) identifying the **optimal path** through the matrix

Needleman, S.B. & Wunsch, C.D. (1970) "A general method applicable to the search for similarities in the amino acid sequences of two proteins." J. Mol. Biol. 48:443-453.

Scoring the alignment matrix

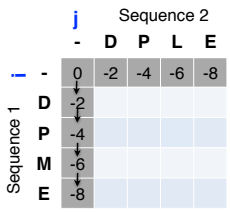
- Start by filling in the first row and column – these are all indels (gaps).
  - Each step you take you will add the **gap penalty** to the score ($S_{i,j}$) accumulated in the previous cell



Scores: match = +1, mismatch = -1, gap = -2

Scoring the alignment matrix

- Start by filling in the first row and column – these are all indels (gaps).
  - Each step you take you will add the **gap penalty** to the score ($S_{i,j}$) accumulated in the previous cell



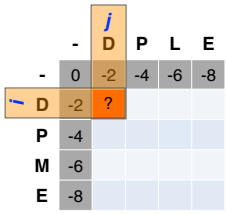
Scores: match = +1, mismatch = -1, gap = -2

$S_{i+4} = (-2) + (-2) + (-2) + (-2)$

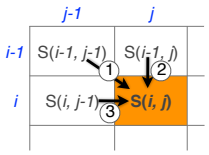
```
Seq1: DPME
Seq2: ----
```

Scoring the alignment matrix

- Then go to the empty corner cell (upper left). It has filled in values in up, left and diagonal directions
  - Now can ask which of the three directions gives the highest score?
  - keep track of this score and direction

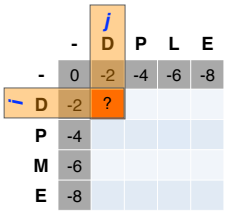


Scores: match = +1, mismatch = -1, gap = -2

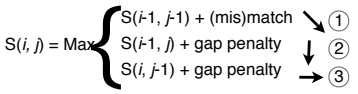


Scoring the alignment matrix

- Then go to the empty corner cell (upper left). It has filled in values in up, left and diagonal directions
  - Now can ask which of the three directions gives the highest score?
  - keep track of this score and direction

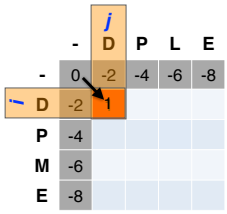


Scores: match = +1, mismatch = -1, gap = -2



Scoring the alignment matrix

- Then go to the empty corner cell (upper left). It has filled in values in up, left and diagonal directions
  - Now can ask which direction gives the highest score
  - keep track of direction and score



Scores: match = +1, mismatch = -1, gap = -2

① (0)+(+1) = +1 <= (D-D) match!

② (-2)+(-2) = -4

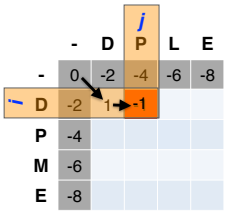
③ (-2)+(-2) = -4

Alignment

D
D

Scoring the alignment matrix

- At each step, the score in the current cell is determine by the scores in the neighboring cells
  - The maximal score and the direction that gave that score is stored (we will use these later to determine the optimal alignment)



Scores: match = +1, mismatch = -1, gap = -2

① (-2)+(-1) = -3 <= (D-P) mismatch!

② (-4)+(-2) = -6

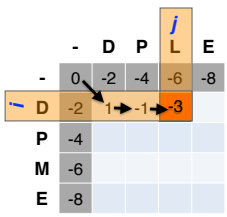
③ (1)+(-2) = -1

Alignment

D-
DP

Scoring the alignment matrix

- We will continue to store the alignment score ($S_{i,j}$) for all possible alignments in the alignment matrix.



Scores: match = +1, mismatch = -1, gap = -2

① (-4)+(-1) = -5 <= (D-L) mismatch

② (-6)+(-2) = -8

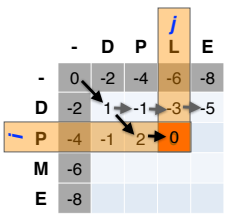
③ (-1)+(-2) = -3

Alignment

D--
DPL

Scoring the alignment matrix

- For the highlighted cell, the corresponding score ($S_{i,j}$) refers to the score of the optimal alignment of the first *i* characters from sequence1, and the first *j* characters from sequence2.



Scores: match = +1, mismatch = -1, indel = -2

① (-1)+(-1) = -2

② (-3)+(-2) = -5

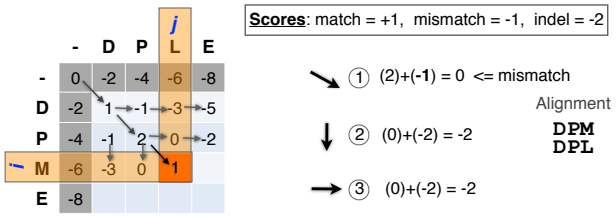
③ (2)+(-2) = 0

Alignment

DP-
DPL

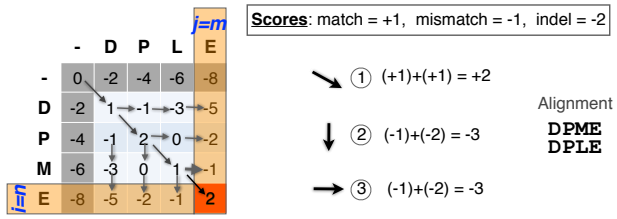
Scoring the alignment matrix

- At each step, the score in the current cell is determine by the scores in the neighboring cells
  - The maximal score and the direction that gave that score is stored



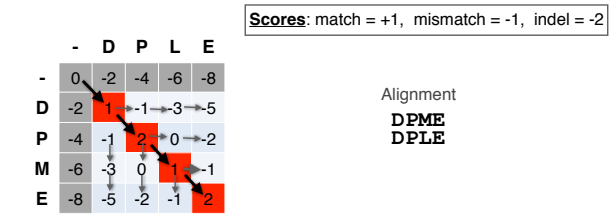
Scoring the alignment matrix

- The score of the best alignment of the entire sequences corresponds to $S_{n,m}$
  - (where *n* and *m* are the length of the sequences)



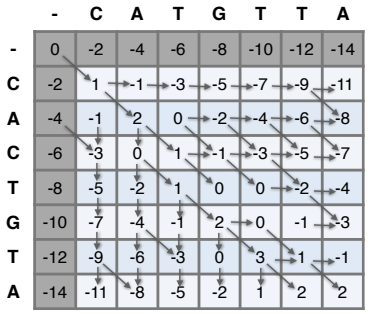
Scoring the alignment matrix

- To find the best alignment, we retrace the arrows starting from the bottom right cell
  - N.B. The optimal alignment score and alignment are dependent on the chosen scoring system



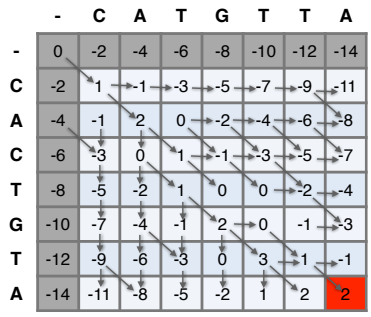
Questions:

- What is the optimal score for the alignment of these sequences and how do we find the optimal alignment?



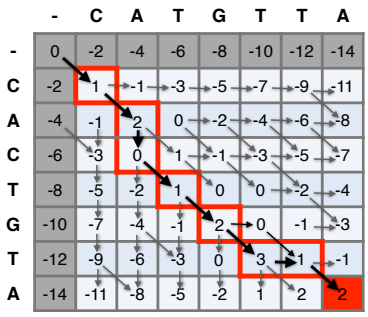
Questions:

- What is the optimal score for the alignment of these sequences and how do we find the optimal alignment?



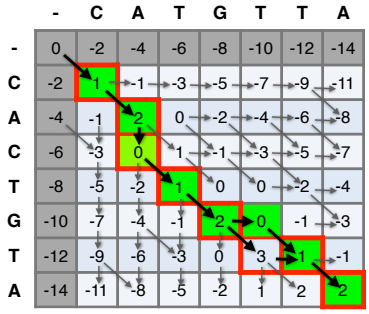
Questions:

- To find the best alignment we retrace the arrows starting from the bottom right cell



More than one alignment possible

- Sometimes more than one alignment can result in the same optimal score



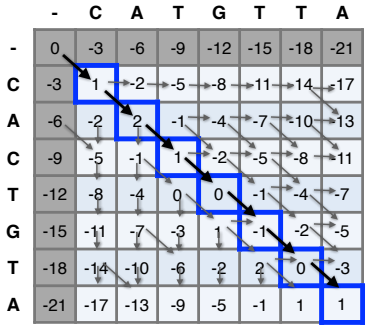
Alignment

```
CACTGT-A
CA-TGTTA

CACTG-TA
CA-TGTTA
```

The alignment and score are dependent on the scoring system

- Here we increase the gap penalty from -2 to -3



Alignment

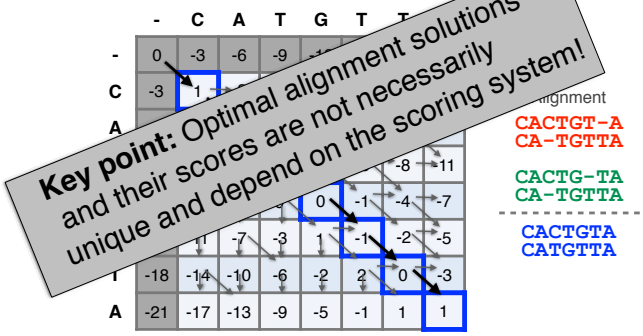
```
CACTGT-A
CA-TGTTA

CACTG-TA
CA-TGTTA

CACTGTA
CATGTTA
```

The alignment and score are dependent on the scoring system

- Here we increase the gap penalty from -2 to -3



Key point: Optimal alignment solutions and their scores are not necessarily unique and depend on the scoring system!

Alignment

```
CACTGT-A
CA-TGTTA

CACTG-TA
CA-TGTTA

CACTGTA
CATGTTA
```

Your Turn!

Hands-on worksheet **Sections 2** & **3**

Match: +2
Mismatch: -1
Gap: -2

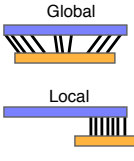
		A	G	T	T	C
	0					
A						
T						
T						
G						
C						

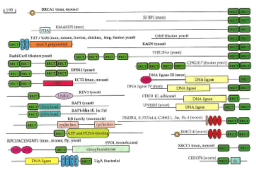
ALIGNMENT FOUNDATIONS

- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

Global *vs* local alignments

- Needleman-Wunsch is a **global alignment** algorithm
  - Resulting alignment spans the complete sequences end to end
  - This is appropriate for closely related sequences that are similar in length
- For many practical applications we require **local alignments**
  - Local alignments highlight sub-regions (*e.g.* protein domains) in the two sequences that align well





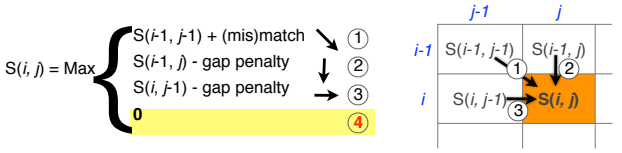
Local alignment: Definition

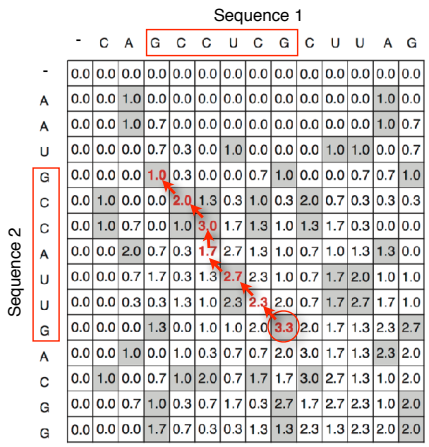
- Smith & Waterman proposed simply that a local alignment of two sequences allow arbitrary-length segments of each sequence to be aligned, with no penalty for the unaligned portions of the sequences. Otherwise, the score for a local alignment is calculated the same way as that for a global alignment

Smith, T.F. & Waterman, M.S. (1981) "Identification of common molecular subsequences." J. Mol. Biol. 147:195-197.

The Smith-Waterman algorithm

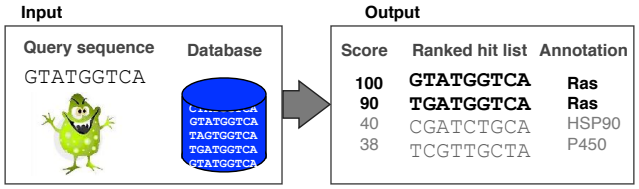
- Three main modifications to Needleman-Wunsch:
  - Allow a node to start at 0
  - The score for a particular cell cannot be negative
    - if all other score options produce a negative value, then a zero must be inserted in the cell
  - Record the highest- scoring node, and trace back from there





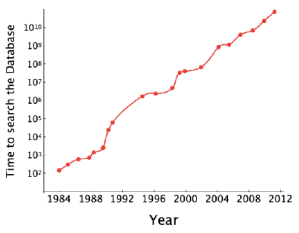
Local alignments can be used for database searching

- **Goal**: Given a query sequence (Q) and a sequence database (D), find a list of sequences from D that are most similar to Q
  - **Input**: Q, D and scoring scheme
  - **Output**: Ranked list of hits



The database search problem

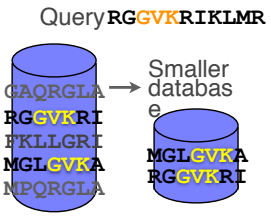
- Due to the rapid growth of sequence databases, search algorithms have to be both efficient and sensitive
  - Time to search with SW is proportional to *m* x *n* (*m* is length of query, n is length of database), **too slow for large databases**!



To reduce search time **heuristic algorithms**, such as BLAST, first remove database sequences without a strong local similarity to the query sequence in a quick initial scan.

The database search problem

- Due to the rapid growth of sequence databases, search algorithms have to be both efficient and sensitive
  - Time to search with SW is proportional to *m* x *n* (*m* is length of query, n is length of database), **too slow for large databases**!



To reduce search time **heuristic algorithms**, such as BLAST, first remove database sequences without a strong local similarity to the query sequence in a quick initial scan.

ALIGNMENT FOUNDATIONS

- **Why…**
  - Why compare biological sequences?
- **What…**
  - Alignment view of sequence changes during evolution (matches, mismatches and gaps)
- **How…**
  - Dot matrices
  - Dynamic programing
    - Global alignment
    - Local alignment
  - BLAST heuristic approach

Rapid, heuristic versions of Smith–Waterman: **BLAST**

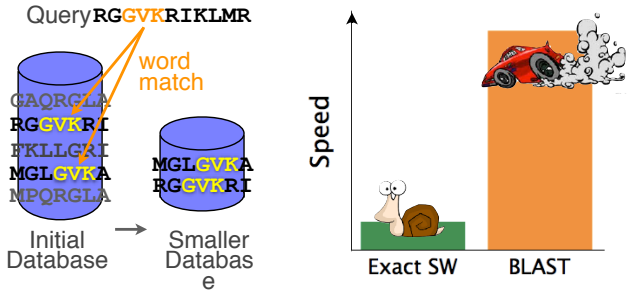
- BLAST (Basic Local Alignment Search Tool) is a simplified form of Smith-Waterman (SW) alignment that is popular because it is **fast** and **easily accessible**
  - BLAST is a heuristic approximation to SW - It examines only part of the search space
  - BLAST saves time by restricting the search by scanning database sequences for likely matches before performing more rigorous alignments
  - Sacrifices some sensitivity in exchange for speed
  - In contrast to SW, BLAST is not guaranteed to find optimal alignments

Rapid, heuristic versions of Smith–Waterman: **BLAST**

- BLAST (Basic Local Alignment Search Tool) is a simplified form of Smith-Waterman (SW) alignment that is popular because it is **fast**
  - BLAST finds regions of sequences
  - BLAST search by scanning matches before performing
  - sensitivity in exchange for speed
  - rast to SW, BLAST is not guaranteed to find optimal alignments

"The central idea of the BLAST algorithm is to confine attention to sequence pairs that contain an initial **word pair** match"
Altschul et al. (1990)

- BLAST uses this pre-screening heuristic approximation resulting in an an approach that is about 50 times faster than the Smith-Waterman



How BLAST works

- Four basic phases
  - **Phase 1**: compile a list of query word pairs (w=3)

RGGVKRI Query sequence

generate list of **w=3 words** for query

RGG
GGV
GVK
VKR
KRI

Blast

- **Phase 2**: expand word pairs to include those similar to query (defined as those above a similarity threshold to original word, i.e. match scores in substitution matrix)

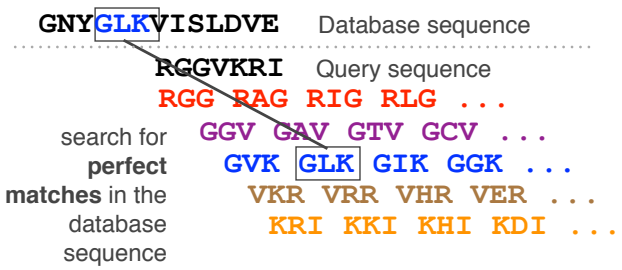
RGGVKRI Query sequence

extend list of words similar to query

RGG RAG RIG RLG ...
GGV GAV GTV GCV ...
GVK GAK GIK GGK ...
VKR VRR VHR VER ...
KRI KKI KHI KDI ...

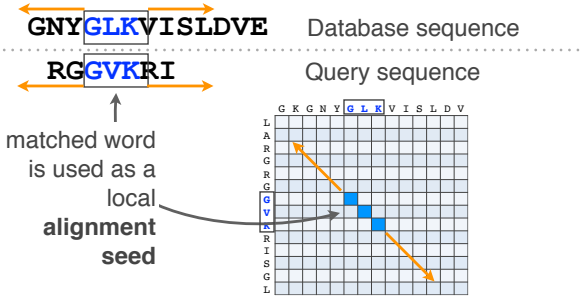
Blast

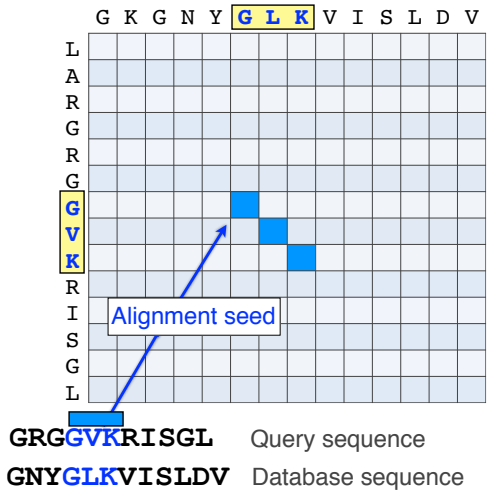
– **Phase 3**: a database is scanned to find sequence entries that match the compiled word list

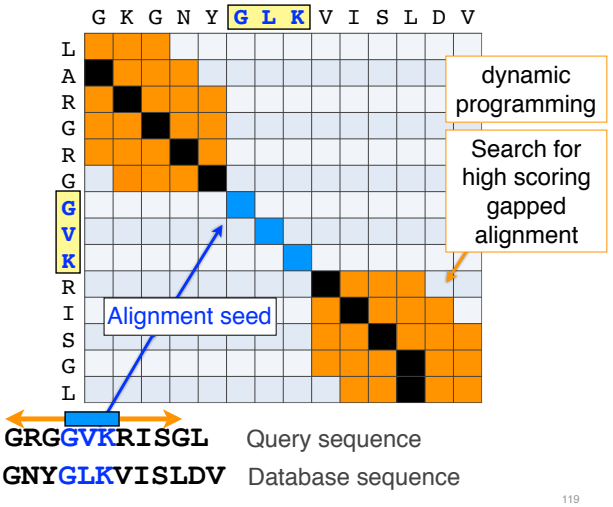


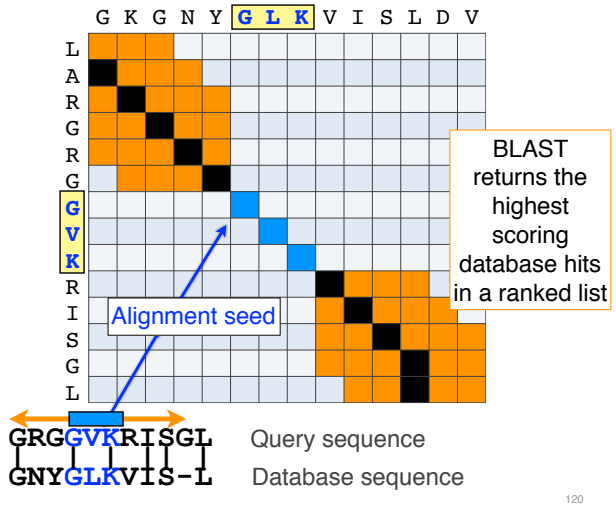
Blast

– **Phase 4**: the initial database hits are extended in both directions using dynamic programing









BLAST output

- BLAST returns the highest scoring database hits in a ranked list along with details about the target sequence and alignment statistics

Description	Max score	Query cover	E value	Max ident	Accession
kinesin-1 heavy chain [Homo sapiens]	677	100%	0	100%	NP_004512.1
Kif5b protein [Mus musculus]	676	100%	0	98%	AAA20133.1
Kinesin-14 heavy chain [Danio rerio]	595	88%	0	78%	XP_00320703
hypothetical protein EGK_18589	48.2	40%	0.03	32%	ELK35081.1
mKIAA4102 protein [Mus musculus]	42.7	38%	3.02	24%	EHH28205.1

Statistical significance of results

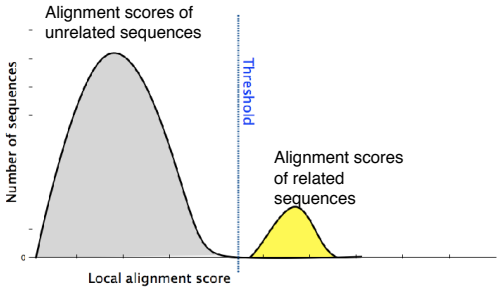
- An important feature of BLAST is the computation of statistical significance for each hit. This is described by the **E value** (expect value)

Description	Max score	Query cover	E value	Max ident	Accession
kinesin-1 heavy chain [Homo sapiens]	677	100%	0	100%	NP_004512.1
Kif5b protein [Mus musculus]	676	100%	0	98%	AAA20133.1
Kinesin-14 heavy chain [Danio rerio]	595	88%	0	78%	XP_00320703
hypothetical protein EGK_18589	48.2	40%	0.03	32%	ELK35081.1
mKIAA4102 protein [Mus musculus]	42.7	38%	3.02	24%	EHH28205.1

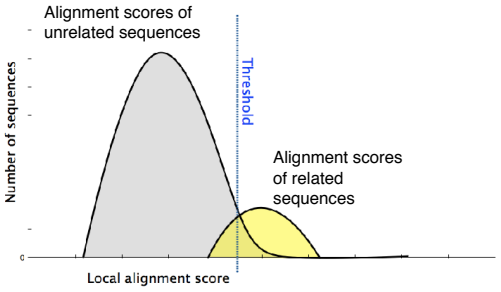
BLAST scores and E-values

- The **E value** is the **expected** number of hits that are as good or better than the observed local alignment score (with this score or better) if the query and database are **random** with respect to each other
  - *i.e.* the number of alignments expected to occur by chance with equivalent or better scores
- Typically, only hits with E value **below** a significance threshold are reported
  - This is equivalent to selecting alignments with score above a certain score threshold

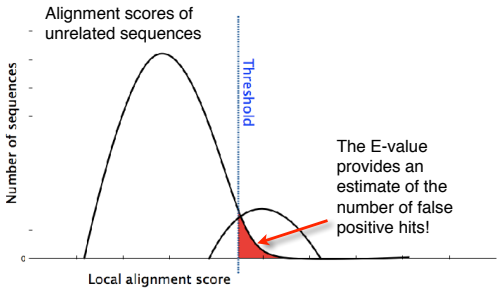
- Ideally, a threshold separates all query related sequences (yellow) from all unrelated sequences (gray)



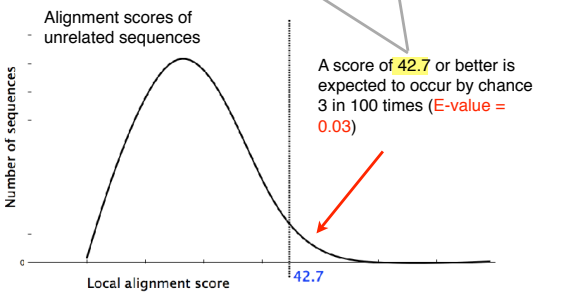
- Unfortunately, often both score distributions overlap
  - The E value describes the expected number of hits with a score above the threshold if the query and database are unrelated



- Unfortunately, often both score distributions overlap
  - The E value describes the expected number of hits with a score above the threshold if the query and database are unrelated



Description	Max score	Query cover	E value	Max ident	Accession
kinesin-1 heavy chain [Homo sapiens]	677	100%	0	100%	NP_004512.1
Kif5b protein [Mus musculus]	676	100%	0	98%	AAA20133.1
Kinesin-14 heavy chain [Danio rerio]	595	88%	0	78%	XP_00320703
hypothetical protein EGK_18589	42.7	40%	0.03	32%	ELK35081.1

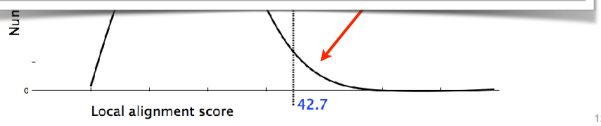


Description	Max score	Total score	Query cover	E value	Max ident	Accession
kinesin-1 heavy chain [Homo	677	677	100%	0	100%	NP_004512.1
Kif5b protein [Mus musculus]	676	676	100%	0	98%	AAA20133.1

In general *E* values < 0.005 are usually significant.

To find out more about *E* values see: "*The Statistics of Sequence Similarity Scores*" available in the help section of the NCBI BLAST site:

http://www.ncbi.nlm.nih.gov/blast/tutorial/Altschul-1.html

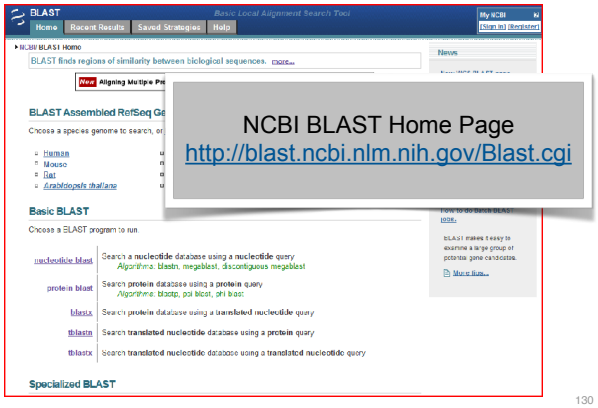


Your Turn!

Hands-on worksheet **Sections 4** & **5**

- Please do answer the last lab review question (**Q19**).
- We encourage discussion and exploration!

Practical database searching with BLAST



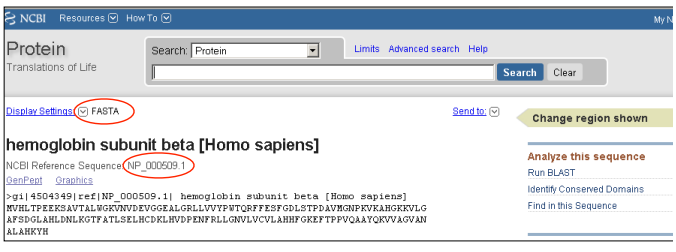
NCBI BLAST Home Page
http://blast.ncbi.nlm.nih.gov/Blast.cgi

Practical database searching with BLAST

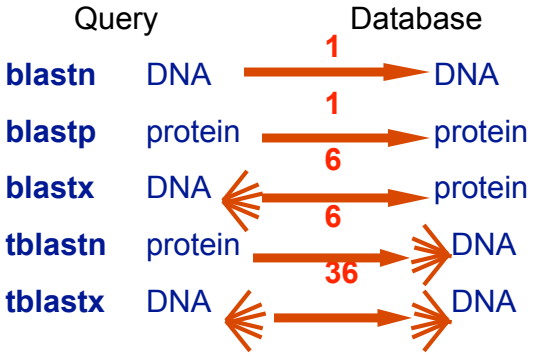
- There are four basic components to a traditional BLAST search
  - (1) Choose the sequence (query)
  - (2) Select the BLAST program
  - (3) Choose the database to search
  - (4) Choose optional parameters
- Then click "BLAST"

Step 1: Choose your sequence

- Sequence can be input in FASTA format or as accession number

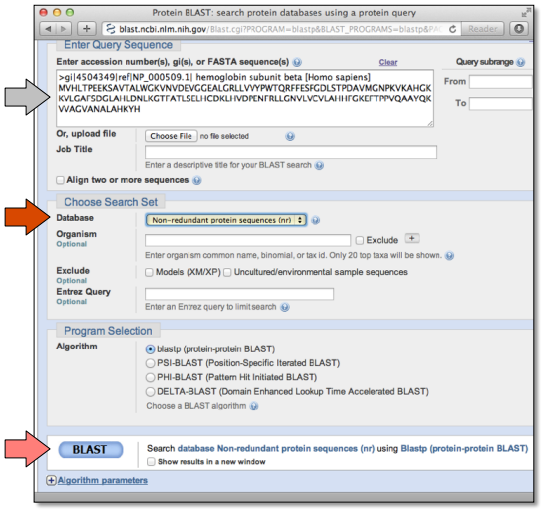


Step 2: Choose the BLAST program



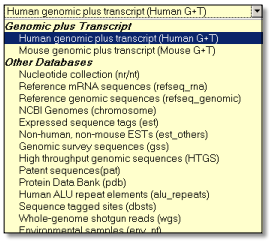
DNA potentially encodes six proteins

```
      5′ CAT CAA
        5′ ATC AAC
          5′ TCA ACT
5′ CATCAACTACAACTCCAAAGACACCCTTACACATCAACAAACCTACCCAC 3′
3′ GTAGTTGATGTTGAGGTTTCTGTGGGAATGTGTAGTTGTTTGGATGGGTG 5′
                                        5′ GTG GGT
                                          5′ TGG GTA
                                            5′ GGG TAG
```

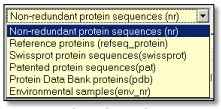



Step 3: Choose the database

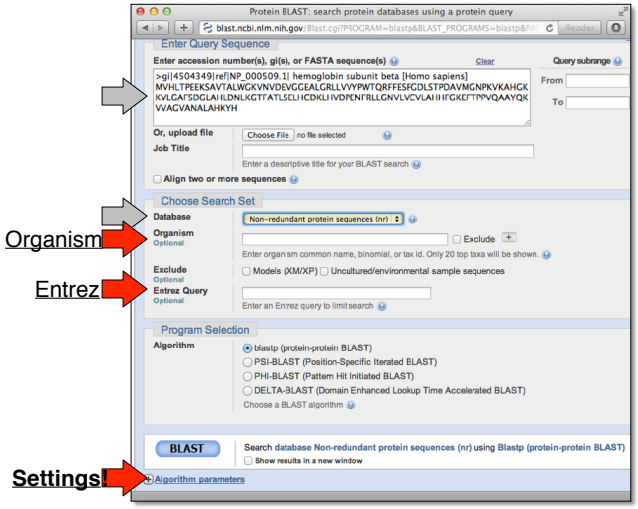
nr = non-redundant (most general database)
dbest = database of expressed sequence tags
dbsts = database of sequence tag sites
gss = genomic survey sequences



nucleotide databases



protein databases

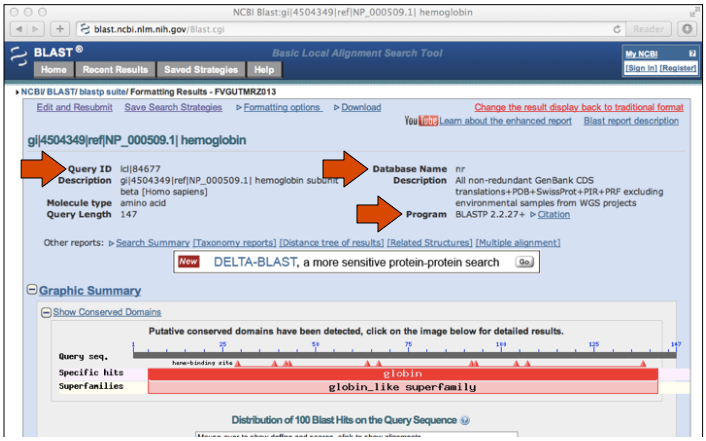


Step 4a: Select optional search parameters

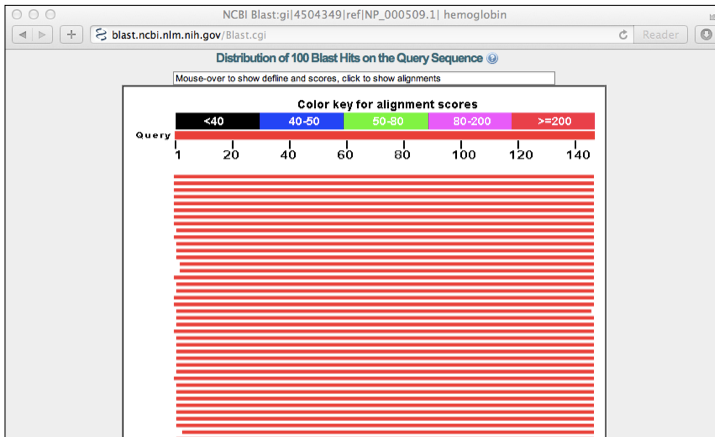
Step 4: Optional parameters

- You can...
  - choose the organism to search
  - change the substitution matrix
  - change the expect (E) value
  - change the word size
  - change the output format

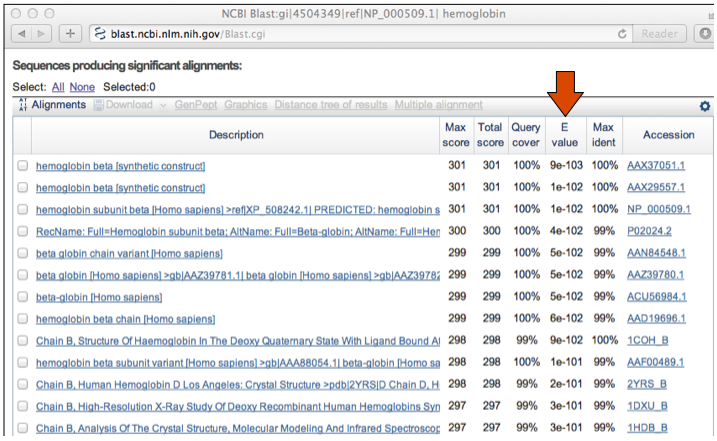
Results page



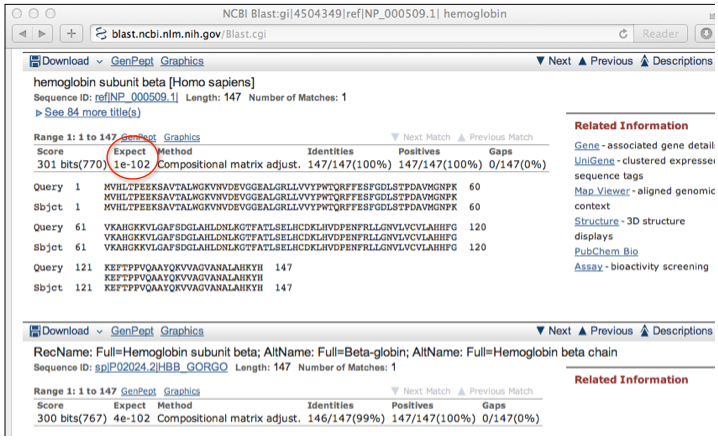
Further down the results page...



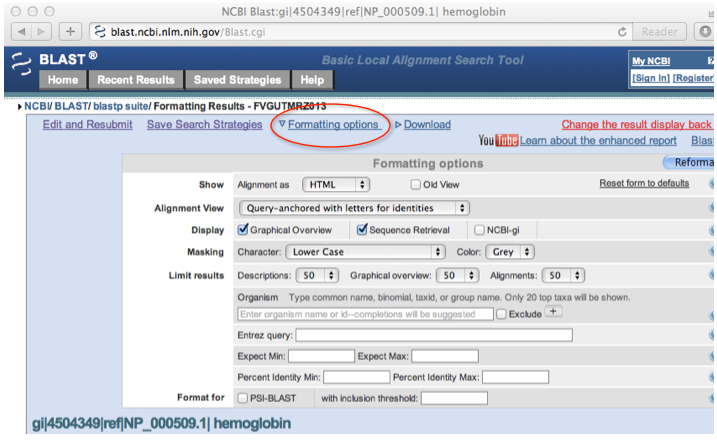
Further down the results page...



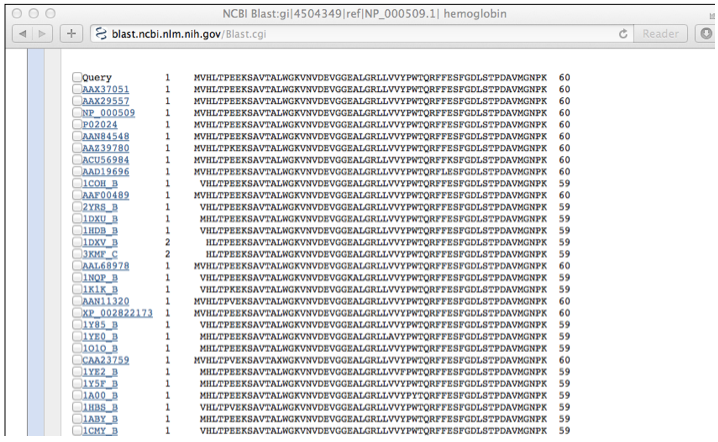
Further down the results page...



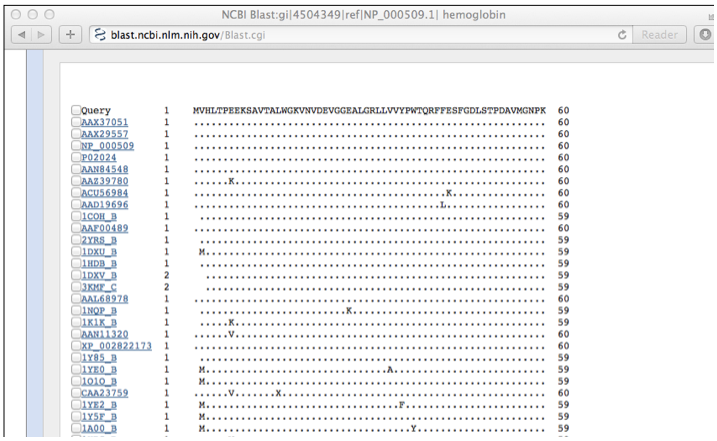
Different output formats are available



E.g. Query anchored alignments



... and alignments with dots for identities



Common problems

- Selecting the wrong version of BLAST
- Selecting the wrong database
- Too many hits returned
- Too few hits returned
- Unclear about the significance of a particular result - are these sequences homologous?

How to handle too many results

- Focus on the question you are trying to answer
  - select “refseq” database to eliminate redundant matches from “nr”
  - Limit hits by organism
  - Use just a portion of the query sequence, when appropriate
  - Adjust the expect value; lowering *E* will reduce the number of matches returned

How to handle too few results

- Many genes and proteins have no significant database matches
  - remove Entrez limits
  - raise E-value threshold
  - search different databases
  - try scoring matrices with lower BLOSUM values (or higher PAM values)
  - use a search algorithm that is more sensitive than BLAST (*e.g.* PSI-BLAST or HMMer)

Summary of key points

- Sequence alignment is a fundamental operation underlying much of bioinformatics.
- Even when optimal solutions can be obtained they are not necessarily unique or reflective of the biologically correct alignment.
- Dynamic programming is a classic approach for solving the pairwise alignment problem.
- Global and local alignment, and their major application areas.
- Heuristic approaches are necessary for large database searches and many genomic applications.

FOR NEXT CLASS…

Check out the online:

- ☑ **Reading**: Sean Eddy’s “What is dynamic programming?”
- ☑ **Homework**: (1) **Quiz**, (2) **Alignment Exercise**.

To Update!

Homework Grading

Both (1) quiz questions and (2) alignment exercise carry equal weights (*i.e.* 50% each).

(Homework 2) Assessment Criteria	Points	
Setup labeled alignment matrix	1	
Include initial column and row for GAPs	1	
All alignment matrix elements scored (i.e. filled in)	1	
Evidence for correct use of scoring scheme	1	
Direction arrows drawn between all cells	1	
Evidence of multiple arrows to a given cell if appropriate	1	D
Correct optimal score position in matrix used	1	C
Correct optimal score obtained for given scoring scheme	1	B
Traceback path(s) clearly highlighted	1	A
Correct *alignment(s)* yielding optimal score listed	1	A+