Department of Biology

Penn
University of Pennsylvania

Introduction to Bioinformatics

ITMAT Bioinformatics

Lecture Five
Nucleic Acid, Sequencing Basics, Alignment

Fall 2023

Lecturer
Gregory R. Grant

Teaching Assistants
Chetan Vadali
Jianing Yang

Gregory R. Grant

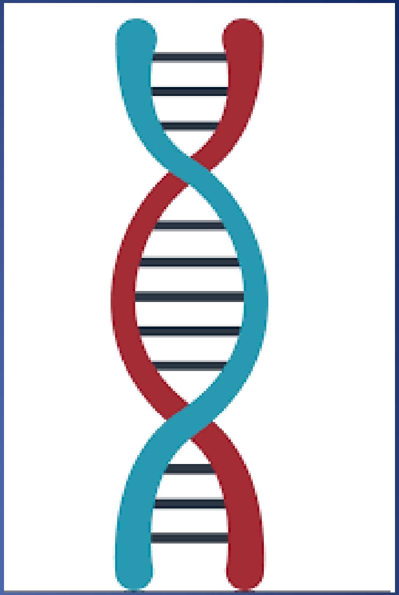
Genetics Department

ggrant@pennmedicine.upenn.edu

ITMAT Bioinformatics Laboratory

University of Pennsylvania

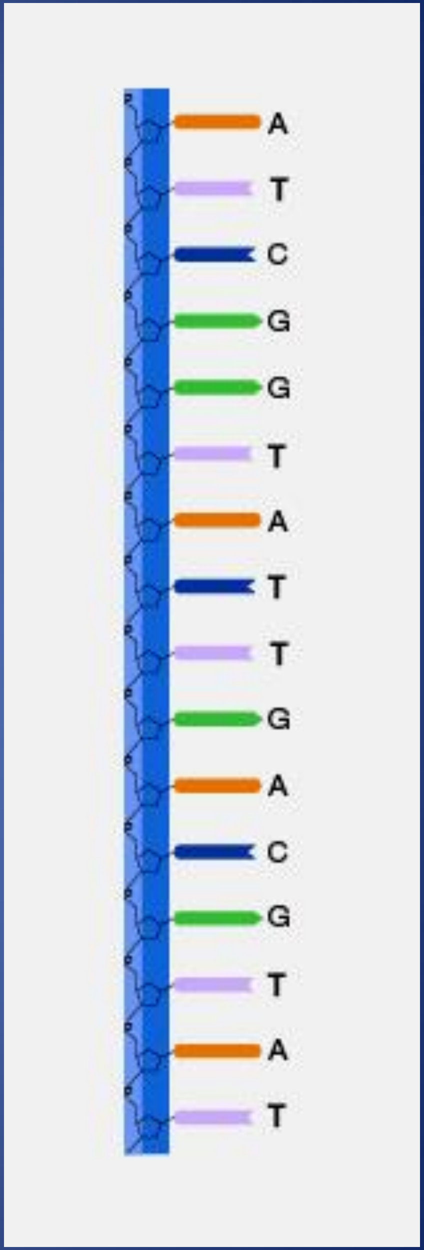
Nucleic Acid

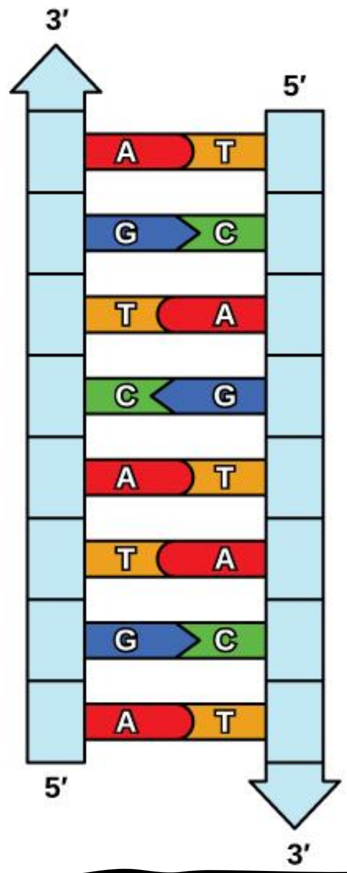


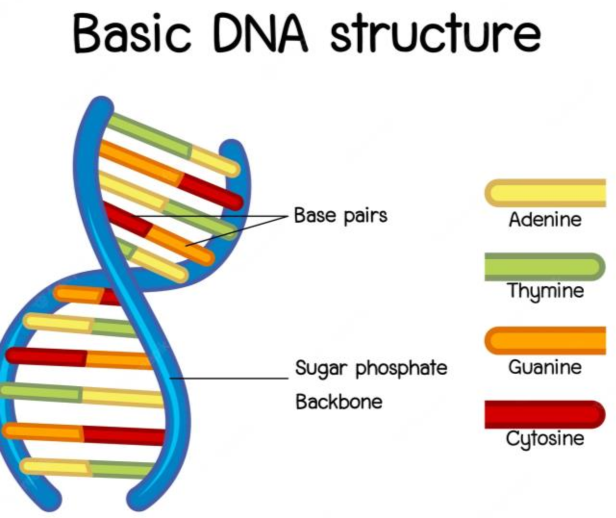
- Nucleic Acids form the primary information-carrying molecules in a cell.
- There are two major classes:
  - DNA
  - RNA
- DNA is typically double-stranded, RNA is typically single-stranded.
  - However, like everything in biology, there are exceptions.
- Here we care very little about the chemistry and physics of these molecules.
- What we really care about is its ‘sequence’

Sequence

- A single strand of DNA is a molecule that is constructed from four different ‘nucleotides’ connected *linearly*.
- We abbreviate the four nucleotides A, C, G and T
  - adenine, cytosine, guanine and thymine
- Therefore, as far as we’re concerned, a strand of DNA is just one long word written in four letters.
- These words encode most of the information it takes to make an organism.







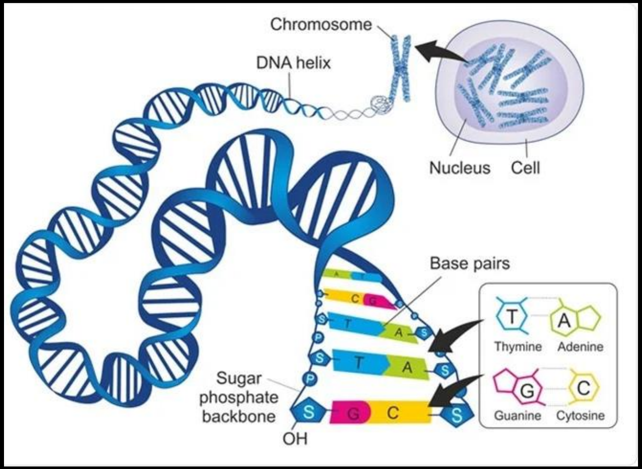
DNA Double Helix

- DNA typically has two strands.
- But one strand uniquely determines the other.
  - A only pairs with T
  - C only pairs with G.

Terminology

- A **“String”, or “Sequence”** is an ordered list of letters, or numbers.
- A **“Substring”** of a string is a *contiguous* sequence of characters from the string
- A **“Subsequence”** of a string is sequence derived from another sequence by deleting some elements without changing the order of the remaining elements

AGACTATCTACAATTGT	- string
CTATCTA	- substring
AGA TATC AATT	- subsequence

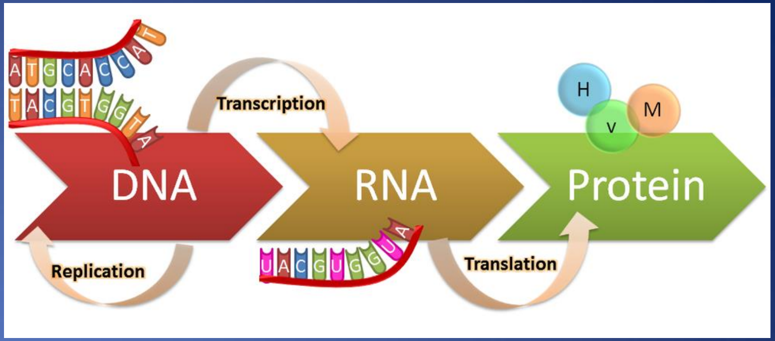


DNA

- DNA is somewhat passive, like a book.
  - Same in every cell.
  - Interacts with environment by being "read".
    - Also known as "transcription".
  - DNA consists of just a few very long sequences, called 'chromosomes', which are tens to hundreds of millions of bases each.

Genes

"Genes" are substrings of chromosomes that get copied into RNA, which then go off and do something.



Some RNA are translated into protein, in which case RNA is just a transfer of information from DNA to the protein assembly machinery.

The protein coding part is a subsequence of the gene.

- Consisting just of exons

The corresponding RNA are called mRNA, 'm' for 'messenger'

All other RNA are called 'non-coding'.

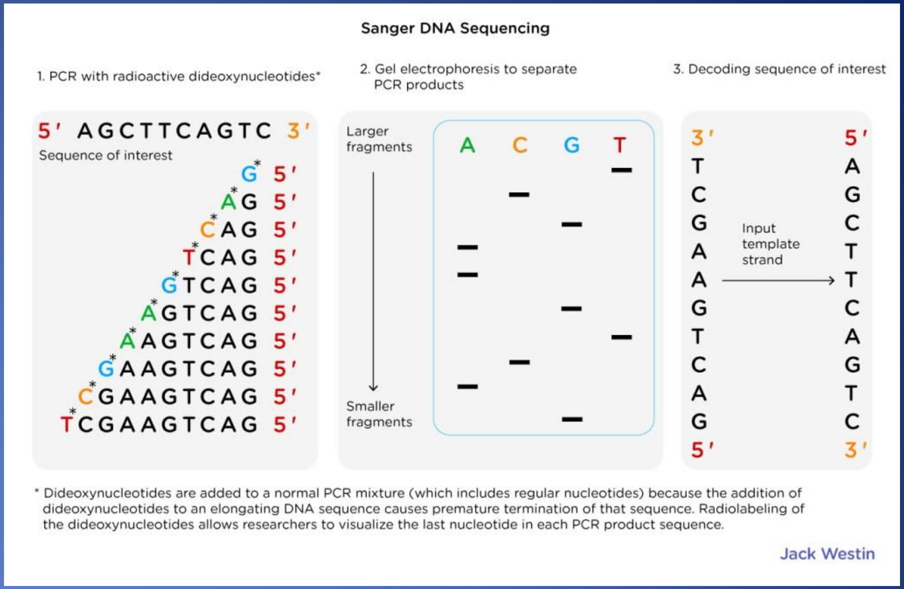
most non-coding RNA are 'regulatory'.

Some non-coding RNA are 'structural' and form components of complexes.

Non-coding RNA comes in all sizes, from microRNA of ~20 bases to long intergenic RNA (lincRNA) which can be thousands of bases

Sequencing

- The structure of DNA was discovered (exactly) 70 years ago (1953).


Sanger DNA Sequencing
1. PCR with radioactive dideoxynucleotides*
2. Gel electrophoresis to separate PCR products
3. Decoding sequence of interest
5′ AGCTTCAGTC 3′
Sequence of interest
G 5′
AG 5′
CAG 5′
TCAG 5′
GTCAG 5′
AGTCAG 5′
AAGTCAG 5′
GAAGTCAG 5′
CGAAGTCAG 5′
TCGAAGTCAG 5′
Larger fragments
Smaller fragments
A C G T
3′ T C G A A G T C A G 5′
5′ A G C T T C A G T C 3′
Input template strand
* Dideoxynucleotides are added to a normal PCR mixture (which includes regular nucleotides) because the addition of dideoxynucleotides to an elongating DNA sequence causes premature termination of that sequence. Radiolabeling of the dideoxynucleotides allows researchers to visualize the last nucleotide in each PCR product sequence.
Jack Westin

- Since then, our ability to read DNA sequences has progressed steadily.
- In the 1970's "Sanger Sequencing" was developed which made sequencing routine for the first time.
  - They gave Sanger his second Nobel Prize for this.
  - But it was still relatively low throughput and expensive.

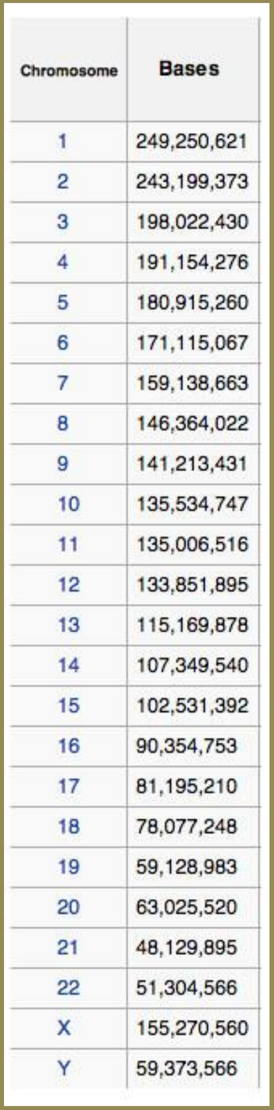
The Human Genome

- Using Sanger Sequencing it took more than 10 years and billions of dollars to sequence a first rough draft of the human genome.
  - 3,200,000,000 nucleotides.

- So-called "Next Generation Sequencing" (NGS) was developed in the early 2000's and by 2010 was already in widespread use.
  - They also call it "High Throughput Sequencing" (HTS).
- We can now sequence a human genome in a weekend for a couple thousand dollars.

Length Limitations of Sequencing Methods

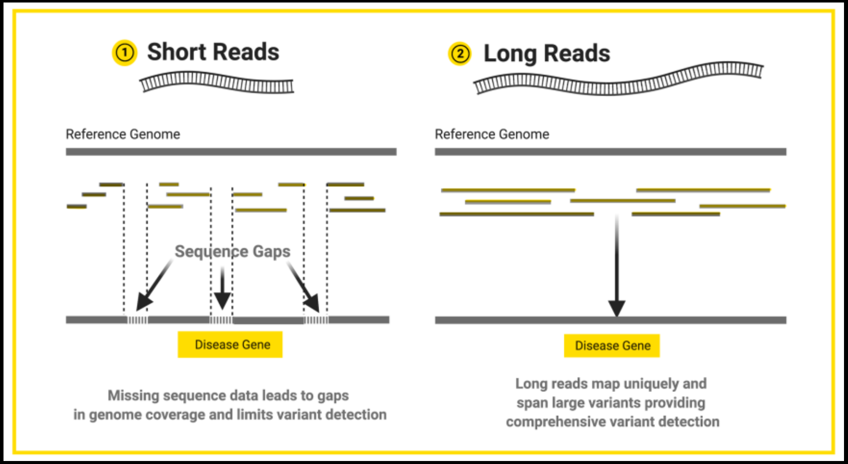
- A typical DNA molecule is 100 million nucleotides.
- We hope one day to be able to read such a long (single) molecule from end to end.
  - But right now, we're not even close.
- TERMINOLOGY: A contiguous segment of sequence output by a sequencing machine is called a "READ"
  - Sanger sequencing produces reads on the order of 500 bases.
  - NGS reads are closer to 150 bases.
    - 125-150 is routine, though it can be pushed to over 250 now.
  - Longer read technologies are maturing, but they still have issues limiting their practical application.



Chromosome	Bases
1	249,250,621
2	243,199,373
3	198,022,430
4	191,154,276
5	180,915,260
6	171,115,067
7	159,138,663
8	146,364,022
9	141,213,431
10	135,534,747
11	135,006,516
12	133,851,895
13	115,169,878
14	107,349,540
15	102,531,392
16	90,354,753
17	81,195,210
18	78,077,248
19	59,128,983
20	63,025,520
21	48,129,895
22	51,304,566
X	155,270,560
Y	59,373,566

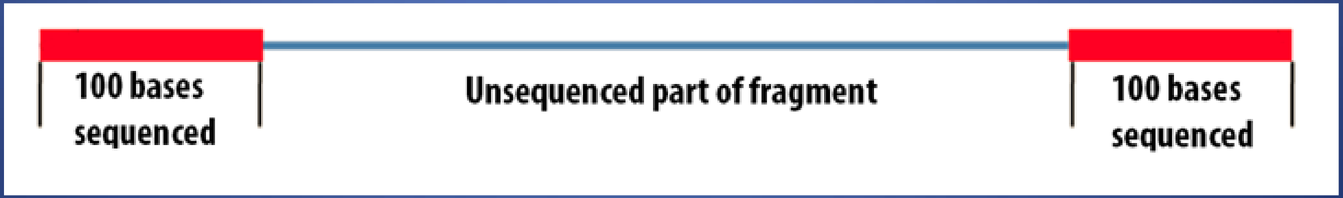
Short vs. Long Read Technology

- There are platforms that produce reads on the order of thousands of bases.
  - Two platforms currently offer this: Pacific Biosciences and Oxford Nanopore
- But they have very high error rates, on the order of 15%.
- One company, Illumina, has a near monopoly in the short-read market.
  - Therefore, we will focus on Illumina short-read data

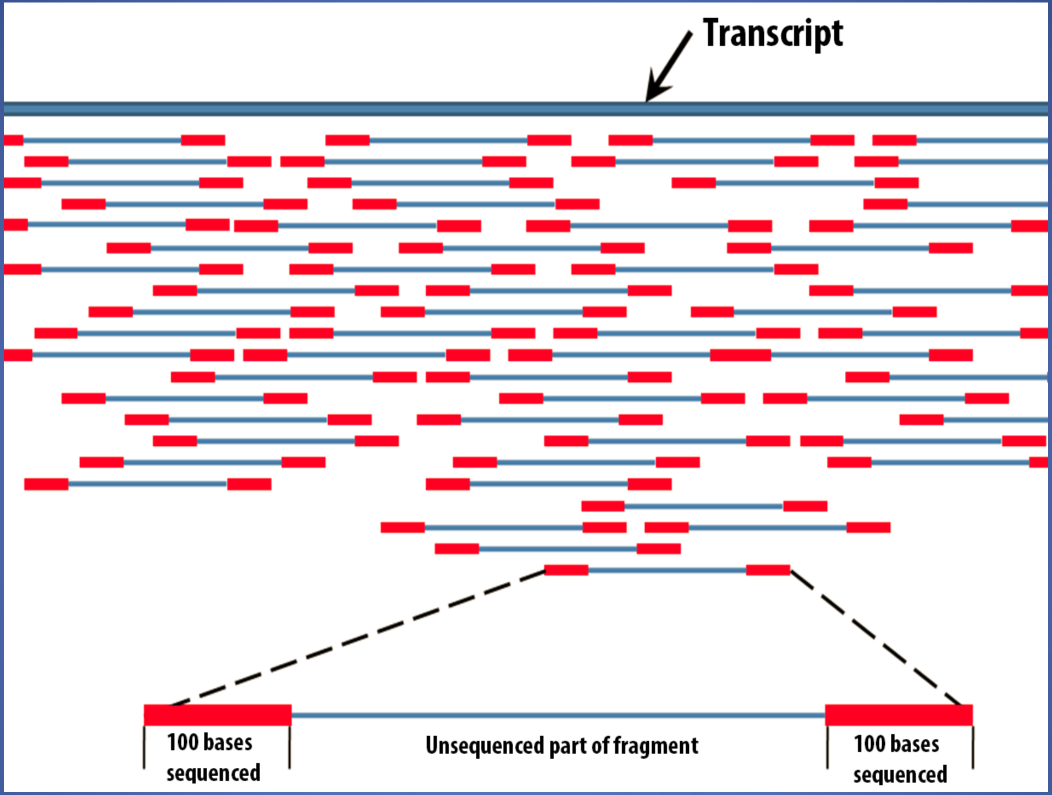

① Short Reads
Reference Genome
Sequence Gaps
Disease Gene
Missing sequence data leads to gaps in genome coverage and limits variant detection
② Long Reads
Reference Genome
Disease Gene
Long reads map uniquely and span large variants providing comprehensive variant detection

Shotgun Sequencing

- In order to sequence a long molecule, it is first fragmented into small pieces on the order of 200 – 500 bases.
- Then for each fragment, only about 100-150 bases are read, typically from both ends.



- When sequencing a transcript, the raw data consists of just the red parts in the figure.

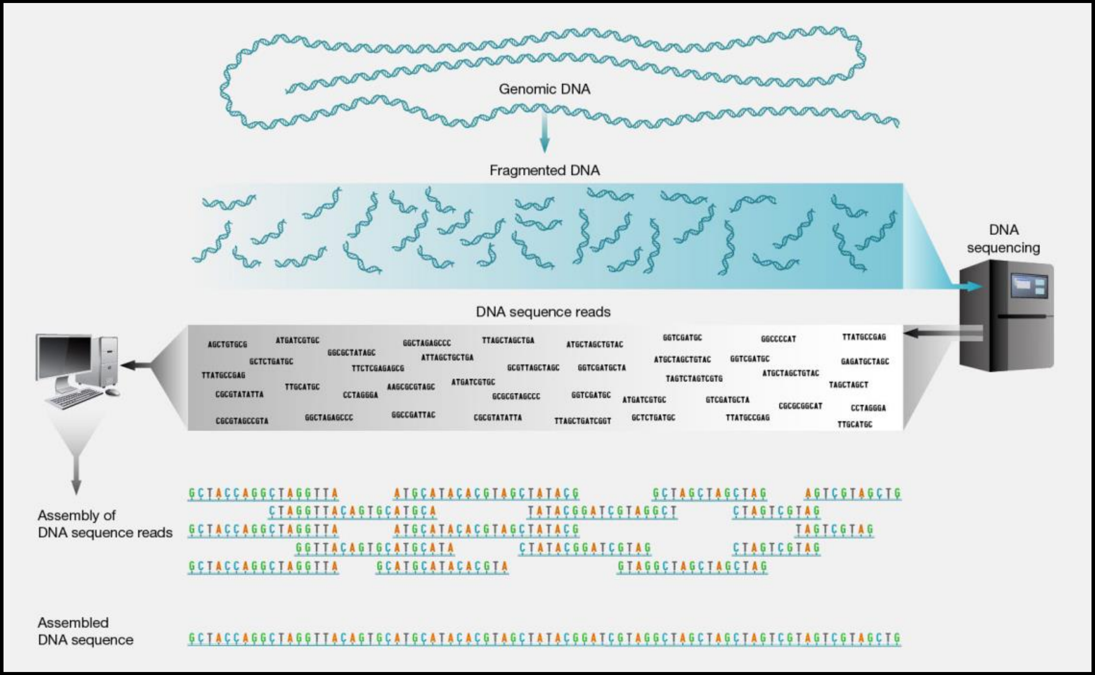


- For technical reasons, reads don't (usually) go all the way to the very end of the fragments.
  - But usually close to it.
  - We'll see why in a later lecture

Shotgun Sequencing

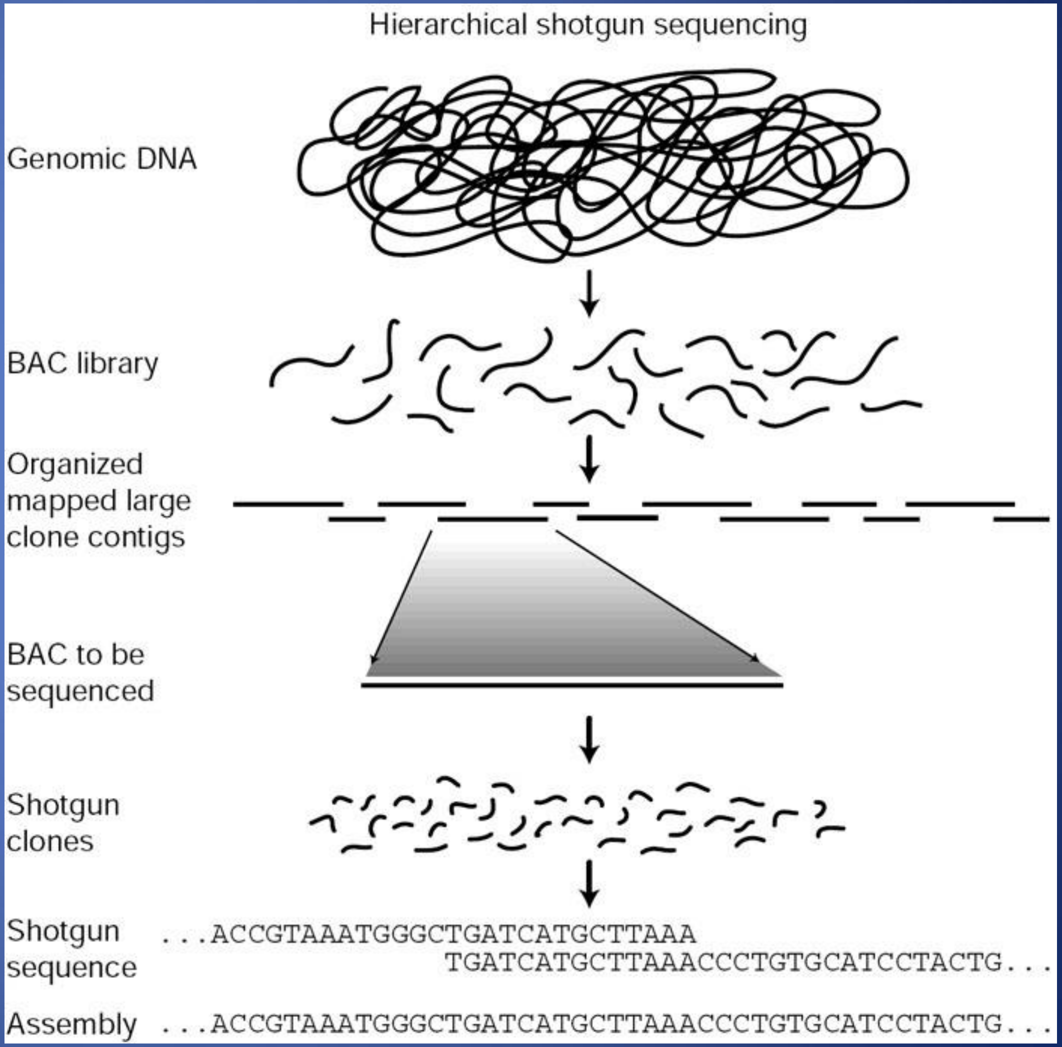
Assembly

- Once we have the (millions of) reads, then comes the hard part.
- Assembling them back into the full original sequence by identifying overlapping reads and glue them together into longer and longer "contigs".

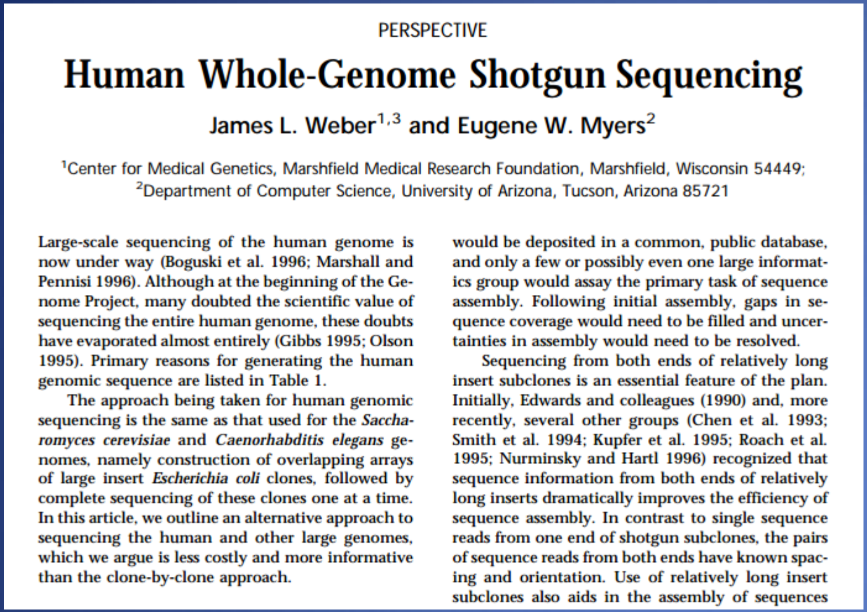


The Human Genome Project

- They started with a hierarchical strategy.
- First divide the genome into large pieces (on the order of hundreds of thousands of bases each)
  - Called BAC's (Bacterial Artificial Chromosomes)
  - We know how BAC's fit together to make a chromosome.
- Then use shotgun sequencing on each BAC to determine its sequence.


Hierarchical shotgun sequencing
Genomic DNA
BAC library
Organized mapped large clone contigs
BAC to be sequenced
Shotgun clones
Shotgun sequence
...ACCGTAAATGGGCTGATCATGCTTAAA
TGATCATGCTTAAACCCTGTGCATCCTACTG...
Assembly
...ACCGTAAATGGGCTGATCATGCTTAAACCCTGTGCATCCTACTG...

Whole Genome Shotgun



PERSPECTIVE

Human Whole-Genome Shotgun Sequencing

James L. Weber[1,3] and Eugene W. Myers[2]

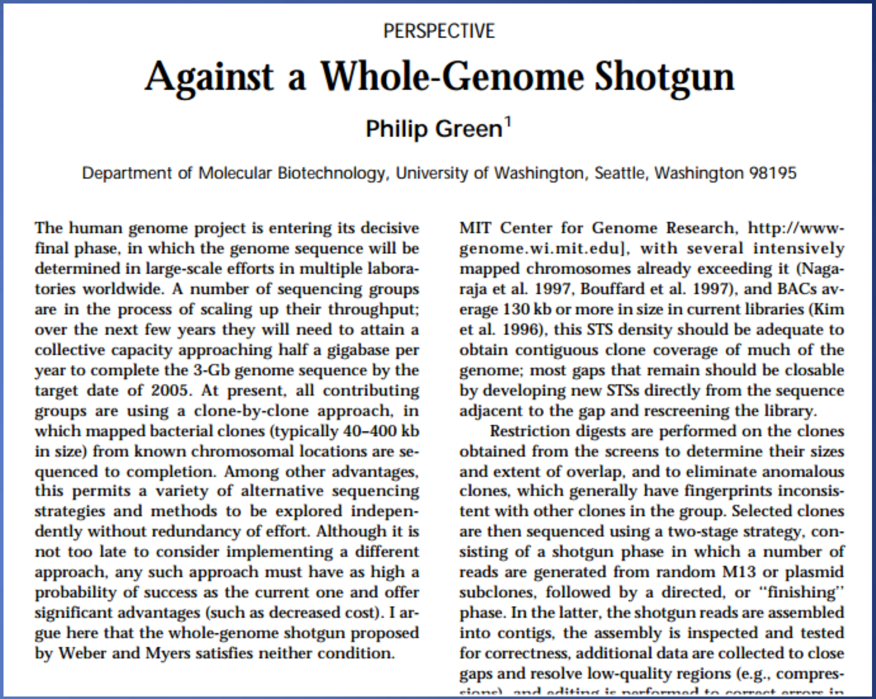
[1]Center for Medical Genetics, Marshfield Medical Research Foundation, Marshfield, Wisconsin 54449; [2]Department of Computer Science, University of Arizona, Tucson, Arizona 85721

Large-scale sequencing of the human genome is now under way (Boguski et al. 1996; Marshall and Pennisi 1996). Although at the beginning of the Genome Project, many doubted the scientific value of sequencing the entire human genome, these doubts have evaporated almost entirely (Gibbs 1995; Olson 1995). Primary reasons for generating the human genomic sequence are listed in Table 1.

The approach being taken for human genomic sequencing is the same as that used for the *Saccharomyces cerevisiae* and *Caenorhabditis elegans* genomes, namely construction of overlapping arrays of large insert *Escherichia coli* clones, followed by complete sequencing of these clones one at a time. In this article, we outline an alternative approach to sequencing the human and other large genomes, which we argue is less costly and more informative than the clone-by-clone approach.

would be deposited in a common, public database, and only a few or possibly even one large informatics group would assay the primary task of sequence assembly. Following initial assembly, gaps in sequence coverage would need to be filled and uncertainties in assembly would need to be resolved.

Sequencing from both ends of relatively long insert subclones is an essential feature of the plan. Initially, Edwards and colleagues (1990) and, more recently, several other groups (Chen et al. 1993; Smith et al. 1994; Kupfer et al. 1995; Roach et al. 1995; Nurminsky and Hartl 1996) recognized that sequence information from both ends of relatively long inserts dramatically improves the efficiency of sequence assembly. In contrast to single sequence reads from one end of shotgun subclones, the pairs of sequence reads from both ends have known spacing and orientation. Use of relatively long insert subclones also aids in the assembly of sequences



PERSPECTIVE

Against a Whole-Genome Shotgun

Philip Green[1]

Department of Molecular Biotechnology, University of Washington, Seattle, Washington 98195

The human genome project is entering its decisive final phase, in which the genome sequence will be determined in large-scale efforts in multiple laboratories worldwide. A number of sequencing groups are in the process of scaling up their throughput; over the next few years they will need to attain a collective capacity approaching half a gigabase per year to complete the 3-Gb genome sequence by the target date of 2005. At present, all contributing groups are using a clone-by-clone approach, in which mapped bacterial clones (typically 40–400 kb in size) from known chromosomal locations are sequenced to completion. Among other advantages, this permits a variety of alternative sequencing strategies and methods to be explored independently without redundancy of effort. Although it is not too late to consider implementing a different approach, any such approach must have as high a probability of success as the current one and offer significant advantages (such as decreased cost). I argue here that the whole-genome shotgun proposed by Weber and Myers satisfies neither condition.

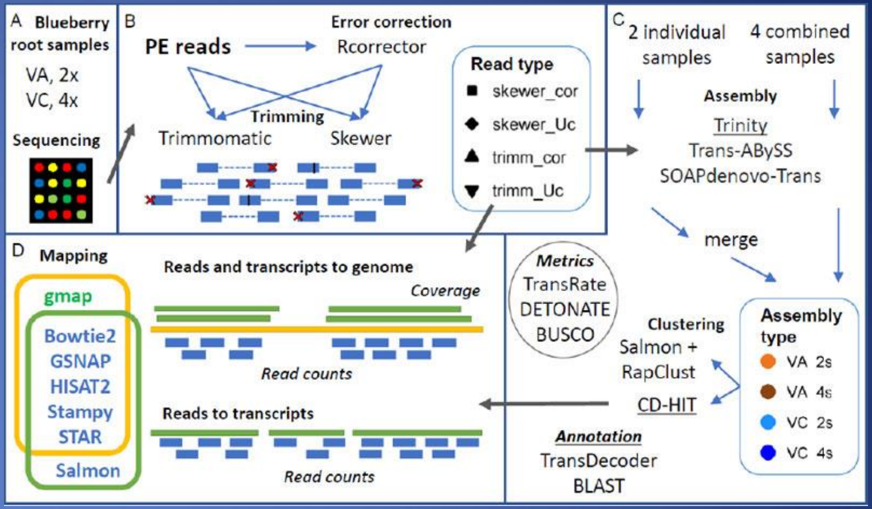
MIT Center for Genome Research, http://www-genome.wi.mit.edu], with several intensively mapped chromosomes already exceeding it (Nagaraja et al. 1997, Bouffard et al. 1997), and BACs average 130 kb or more in size in current libraries (Kim et al. 1996), this STS density should be adequate to obtain contiguous clone coverage of much of the genome; most gaps that remain should be closable by developing new STSs directly from the sequence adjacent to the gap and rescreening the library.

Restriction digests are performed on the clones obtained from the screens to determine their sizes and extent of overlap, and to eliminate anomalous clones, which generally have fingerprints inconsistent with other clones in the group. Selected clones are then sequenced using a two-stage strategy, consisting of a shotgun phase in which a number of reads are generated from random M13 or plasmid subclones, followed by a directed, or "finishing" phase. In the latter, the shotgun reads are assembled into contigs, the assembly is inspected and tested for correctness, additional data are collected to close gaps and resolve low-quality regions (e.g., compressions), and editing is performed to correct errors in

- Around 2000 a debate started.
  - Gene Myers versus Phil Green
- Some were saying skip the BAC step and go straight to whole genome shotgun.
- The whole genome advocates prevailed, and the government changed approach.

From Reads to Analyzable data

- Sequencing genomes is just one of *many* things you can do with NGS.
  - It's the bioinformatician's job to piece together the disconnected information (reads) to accomplish whatever is the goal at hand.



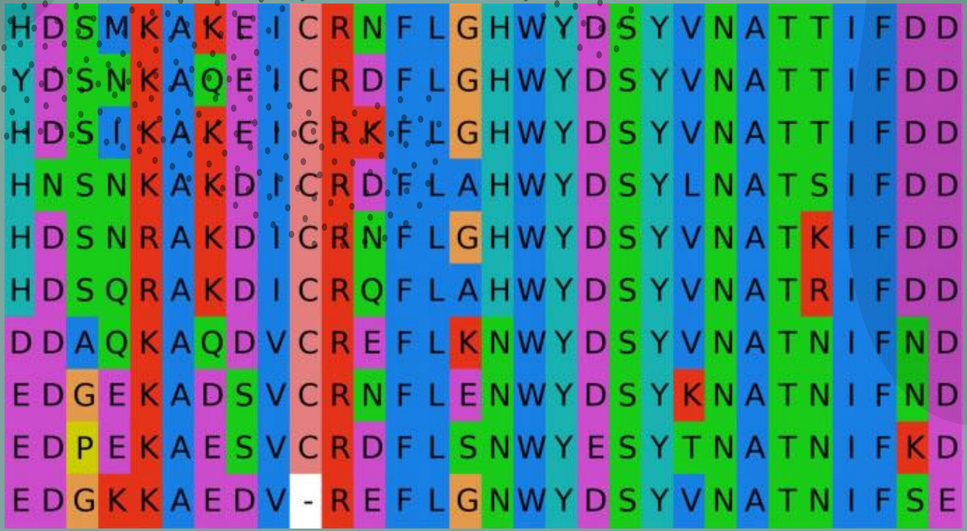
Example Analysis Pipeline

- Given the fast pace of technology these days, method developers are always up against a fast-moving target.
  - You may spend years developing a method, only to find it's obsolete before you even publish it.
  - For example, whole companies and careers were destroyed when sequencing replaced microarrays around 2010.

Example Applications of NGS

- To sequence the DNA of a new species, the job is to assemble reads (puzzle-like) into a full genome.
- If we're sequencing DNA of an already sequenced species (e.g., human) then the job might be to determine single nucleotide polymorphisms.
  - Single base differences with respect to the original reference genome.
- If we're sequencing RNA the job might be to identify which gene a read came from.
  - To assess gene structure, or the expression level of that gene.

Alignment


HDSMKAKEICRNFLGHWYDSYVNATTIFDD
YDSNKAQEICRDFLGHWYDSYVNATTIFDD
HDSIKAKEICRKFLGHWYDSYVNATTIFDD
HNSNKAKDICRDFLAHWYDSYLNATSIFDD
HDSNRAKDICRNFLGHWYDSYVNATKIFDD
HDSQRAKDICRQFLAHWYDSYVNATRIFDD
DDAQKAQDVCREFLKNWYDSYVNATNIFND
EDGEKADSVCRNFLENWYDSYKNATNIFND
EDPEKAESVCRDFLSNWYESYTNATNIFKD
EDGKKAEDV-REFLGNWYDSYVNATNIFSE

- In every case we must employ sequence alignment.
- “Sequence alignment” is a broad term which comes in many flavors.
- But at its most basic, we align two sequences, that we assume are related
  - Perhaps one is a subsequence of the other, or
  - Perhaps both are descended from a common ancestor sequence.
- We align to represent this relationship.

Evolution of Sequences

Consider the following sequence:
AGATTACAGAT

Depending on many factors, this sequence will mutate over time.

- If it’s not a particularly important part of the genome, then it may just drift randomly.
- If it is extremely important, then it may strongly resist mutations.

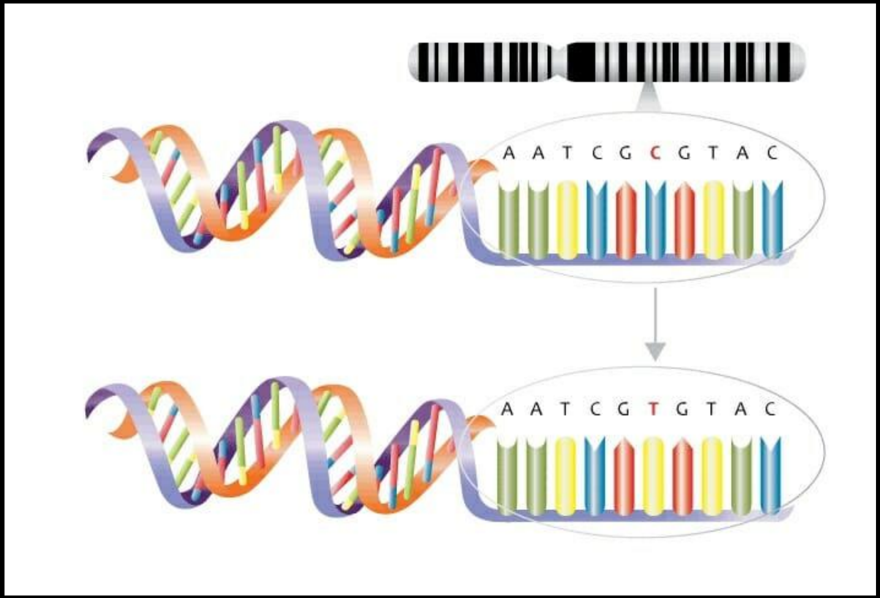
We identify three *basic* mutations that can modify the sequence.

Substitutions

- *Substitutions* happen when one nucleotide changes into another.

  AGATTACAGAT

  AGATAACAGAT
- If this happens in the protein coding part of a gene, it may (or may not) change the protein itself.
- Even if it doesn't alter the protein, it could alter how it is regulated.
- Or it may simply do nothing.



Insertions

- An *Insertion* happened when a new base, or bases, are inserted between two neighboring bases.
- For example, if this sequence

  AGATTACAGAT

  evolves into this one

  AGATTCACAGAT
- If this happens in a protein coding gene, it can have massive consequences since it potentially changes the codon for every amino acid to follow.

Deletions

- A *Deletion* happened when a base, or bases, are deleted.
- For example, when this sequence

  AGATTACAGAT

  mutates into this one

  AGATTCAGAT
- This again can have a massive impact on a protein coding gene.

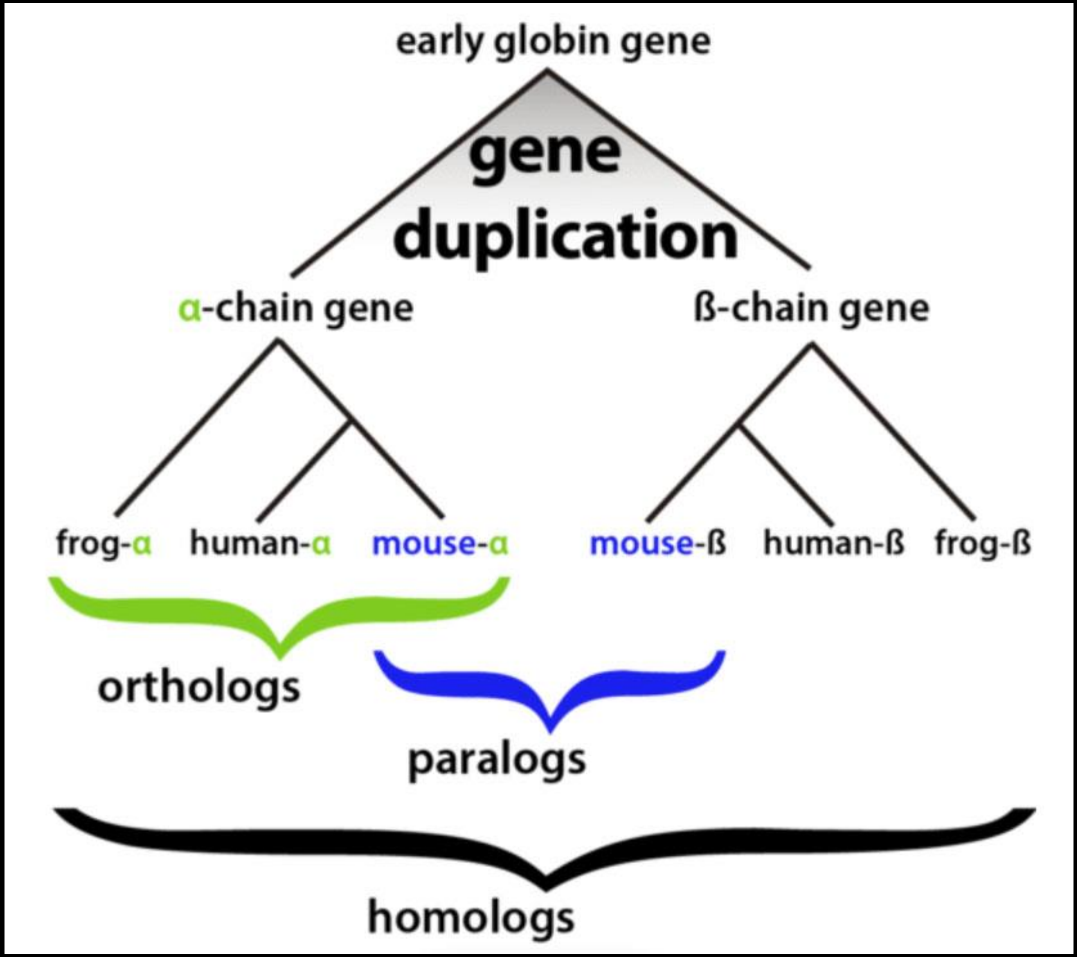
Insertions and Deletions of multiple bases

- Insertions and deletions can involve more than one base in a row.

- For example,
  AGATTACAGAT
  could evolve into
  AGATAGAT

- Since codons consist of three nucleotides, a deletion of a triple usually has a much less dramatic impact on a protein.

Terminology



Directionality

- Suppose we have two sequences in two different species that differ by one base.

AGATTACAGAT (human)

AGATTTCAGAT (mouse)

  - Assume we know these two sequences are related.

- What we don't know is which one was the ancestral sequence and which species experienced the mutation.

- Could be the common ancestor had an A that mutated into a T in the line to humans. Or it could be the other way around.
  - It could even be the result of two mutations.

Alignment

- We represent the evolutionary relationship with an “alignment”

```
AGATTACAGAT
|||| ||||||
AGATAACAGAT
```

- When there are only substitutions, like this, it’s not complicated.

Indels

- When there are insertions or deletions, it’s represented like this

```
AGATTACAGAT      Sequence #1
| | | |       | | | |
AGAT- - - AGAT      Sequence #2
```

- Since we do not know if it was an insertion in Sequence #1 or a deletion in sequence #2 that resulted in this, we use the term ‘***indel***’

Indel/Indel

- When we construct alignments, we specifically avoid aligning an indel with another indel because there's no information added by doing that.
- In other words, we do not make alignments like this:

```
AG-CTC
|   ||
AC-CTG
```

- The indel/indel might as well be removed:

```
AGCTC
|  ||
ACCTG
```

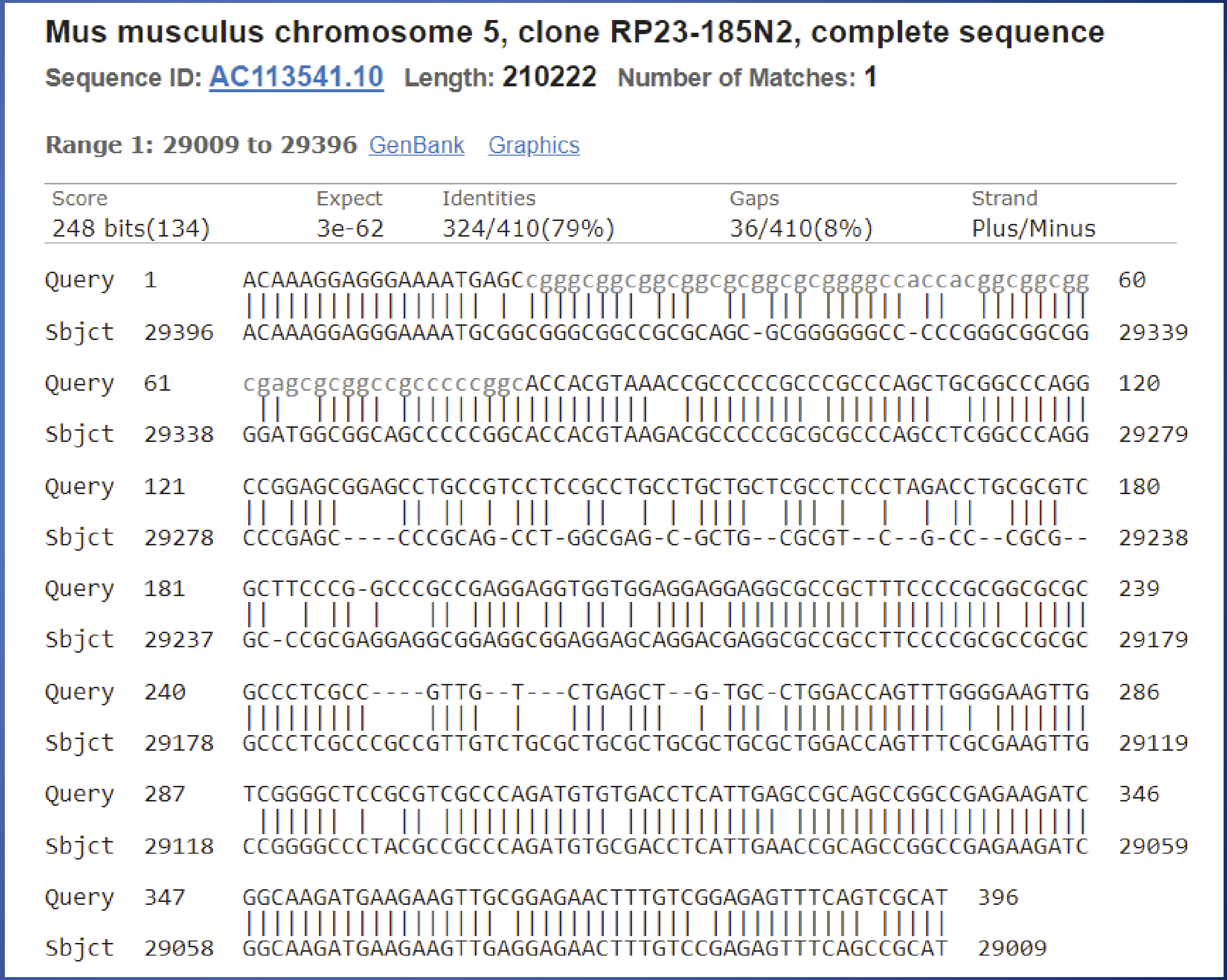
Point of Confusion

- We need a term for something that's not an indel.
  - And the agreed upon term is to call it a 'match'
- But this introduces a point of confusion.
  - Sometimes 'match' simply means not an indel.
  - And other times 'match' means the two things are equal.
- For example, consider the first position in the following, where A is aligned to C.
- In one context we call this a match because it's not an indel.
- In another context we call this a mismatch because they're not equal.
  - It should be clear from context.

```
AGAT-C
CGATTC
```

Example Alignment

- Alignment between a human and mouse gene.
  - Query = Mouse
  - Sbjct = Human
- Notice that the coordinates go up in the query, but down in the Subject.



Mus musculus chromosome 5, clone RP23-185N2, complete sequence

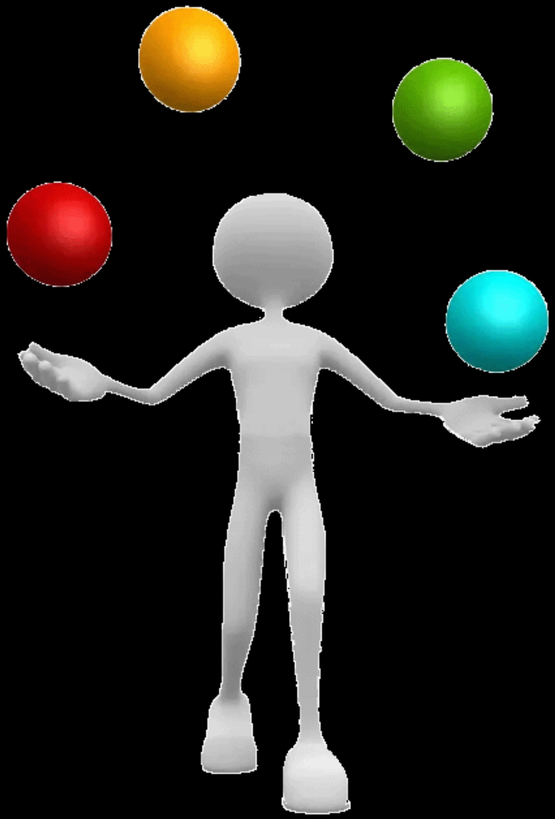
Sequence ID: AC113541.10 **Length:** 210222 **Number of Matches:** 1

Range 1: 29009 to 29396 GenBank Graphics

```
Score            Expect   Identities      Gaps          Strand
248 bits(134)    3e-62    324/410(79%)    36/410(8%)    Plus/Minus

Query  1      ACAAAGGAGGGAAAATGAGCcgggcggcggcggcgcggcgcggggccaccacggcggcgg  60
              ||||||||||||||||| | |||||||| |||  || ||| |||||| ||  ||||||||
Sbjct  29396  ACAAAGGAGGGAAAATGCGGCGGGCGGCCGCGCAGC-GCGGGGGGCC-CCCGGGCGGCGG  29339

Query  61     cgagcgcggccgcccccggcACCACGTAAACCGCCCCCGCCCGCCCAGCTGCGGCCCAGG  120
               ||  ||||| ||||||||||||||||||  ||||||||| ||||||||  |||||||||
Sbjct  29338  GGATGGCGGCAGCCCCCGGCACCACGTAAGACGCCCCCGCGCGCCCAGCCTCGGCCCAGG  29279

Query  121    CCGGAGCGGAGCCTGCCGTCCTCCGCCTGCCTGCTGCTCGCCTCCCTAGACCTGCGCGTC  180
              || ||||    || || | |||  ||  | | ||||  ||| |  |  | ||  ||||
Sbjct  29278  CCCGAGC----CCCGCAG-CCT-GGCGAG-C-GCTG--CGCGT--C--G-CC--CGCG--  29238

Query  181    GCTTCCCG-GCCCGCCGAGGAGGTGGTGGAGGAGGAGGCGCCGCTTTCCCCGCGGCGCGC  239
              ||  | || |   || |||| || || | |||| |||||||||| ||||||||| |||||
Sbjct  29237  GC-CCGCGAGGAGGCGGAGGCGGAGGAGCAGGACGAGGCGCCGCCTTCCCCGCGCCGCGC  29179

Query  240    GCCCTCGCC----GTTG--T---CTGAGCT--G-TGC-CTGGACCAGTTTGGGGAAGTTG  286
              |||||||||    ||||  |   ||| |||  | ||| |||||||||||| | |||||||
Sbjct  29178  GCCCTCGCCCGCCGTTGTCTGCGCTGCGCTGCGCTGCGCTGGACCAGTTTCGCGAAGTTG  29119

Query  287    TCGGGGCTCCGCGTCGCCCAGATGTGTGACCTCATTGAGCCGCAGCCGGCCGAGAAGATC  346
               |||||| |  || |||||||||||| ||||||||||| |||||||||||||||||||||
Sbjct  29118  CCGGGGCCCTACGCCGCCCAGATGTGCGACCTCATTGAACCGCAGCCGGCCGAGAAGATC  29059

Query  347    GGCAAGATGAAGAAGTTGCGGAGAACTTTGTCGGAGAGTTTCAGTCGCAT  396
              |||||||||||||||||| ||||||||||||| ||||||||||| |||||
Sbjct  29058  GGCAAGATGAAGAAGTTGAGGAGAACTTTGTCCGAGAGTTTCAGCCGCAT  29009
```

Optimal Alignment

- How do we know we have the "correct" alignment?
  - We don't and usually we cannot know for sure.
- Instead, we look for the alignment that's "best" in some mathematical sense.
  - On the assumption that a judiciously chosen mathematical criteria can reflect reality closely enough to serve practical purposes.
- In other words, we will find a way to 'score' an alignment so that better alignments have better scores.
  - Then we search for an algorithm that finds the alignment with the best possible score.



Edit Distance

- There are many ways to score an alignment. The simplest is the so-called ‘edit distance’.
- The **edit distance** adds one for each single nucleotide substitution and one for each single nucleotide indel.
- *Smaller edit distances are better.*

```
ACATTACAGAT      Sequence #1
|  ||      ||||
AGAT- - - AGAT     Sequence #2
```

- Edit Distance = 4

Scoring Schemes

A more flexible alternative to the edit distance is to involve negative numbers.

E.g., +1 for a match, -1 for a mismatch and -2 for an indels.

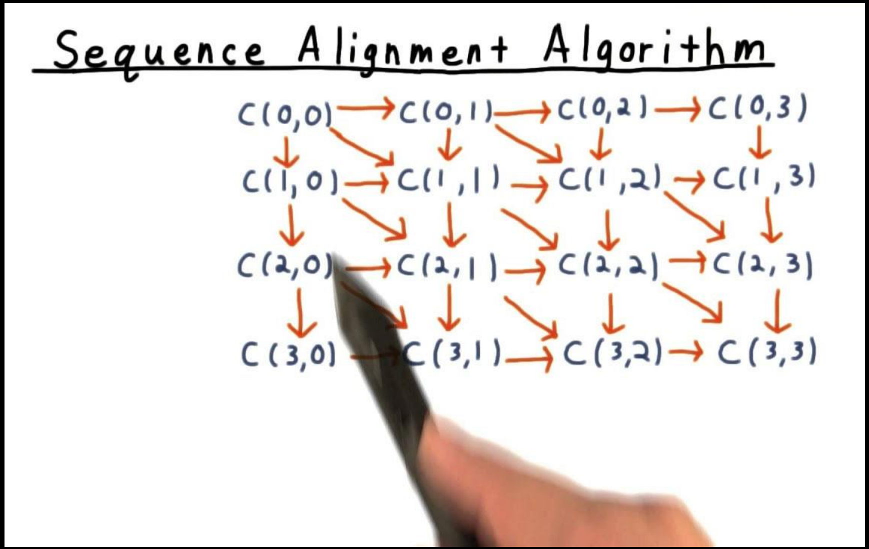
Larger scores are better

Indels are penalized greater than mismatches.

In what follows we will use scores, not edit distances.

Sequence Alignment Algorithms

- Given a scoring scheme, an *alignment algorithm* finds an alignment with the best possible score.
- We say “an” optimal alignment because there may be a tie for first place.
  - Could even be a multi-way tie.



$$\begin{tikzcd}
C(0,0) \arrow[r] \arrow[d] \arrow[dr] & C(0,1) \arrow[r] \arrow[d] \arrow[dr] & C(0,2) \arrow[r] \arrow[d] & C(0,3) \arrow[d] \\
C(1,0) \arrow[r] \arrow[d] \arrow[dr] & C(1,1) \arrow[r] \arrow[d] \arrow[dr] & C(1,2) \arrow[r] \arrow[d] \arrow[dr] & C(1,3) \arrow[d] \\
C(2,0) \arrow[r] \arrow[d] \arrow[dr] & C(2,1) \arrow[r] \arrow[d] \arrow[dr] & C(2,2) \arrow[r] \arrow[d] \arrow[dr] & C(2,3) \arrow[d] \\
C(3,0) \arrow[r] & C(3,1) \arrow[r] & C(3,2) \arrow[r] & C(3,3)
\end{tikzcd}$$

Exhaustive Search

- Without working too hard, we can find an algorithm that lists all possible alignments between two sequences.
  - It is a finite number.
- If we do that and calculate the alignment score for each one as we go, we can be sure to find all optimal alignments.
- This is called an “exhaustive search” algorithm, also known as solving the problem by “brute force”.

Very Large Search Space

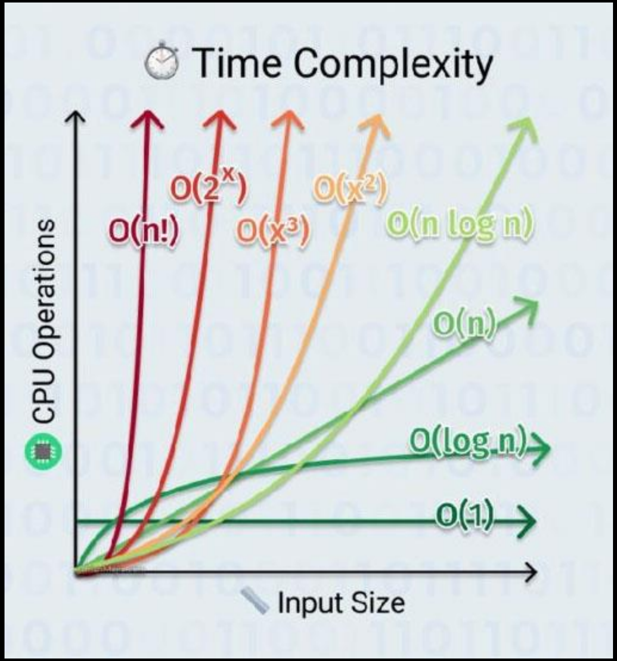
- The number of possible alignments grows quickly with sequence length.
  - For two sequences of length 11, there are 705,432 possible alignments.
- For sequences of length 10,000 the number is astronomical.
  - And biological sequences get a lot longer than that.

Efficient Algorithms

- Since there are so many possible alignments, an exhaustive search for the optimal one(s) is not practical.
  - Instead, we need a clever algorithm to find them in finite time.

- The first algorithm to achieve this was published in 1970 and exploits a technique known as 'dynamic programming'.

- Dynamic programming has nothing to do with 'programming' it is more closely related to mathematical induction.



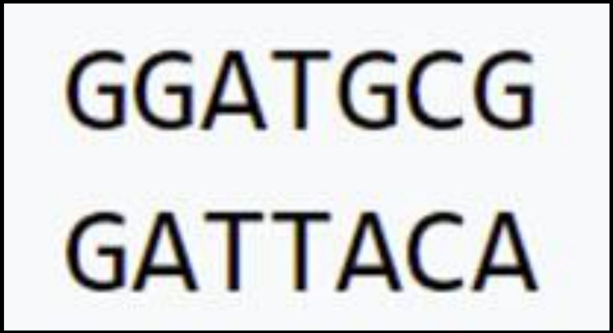
Needleman-Wunsch

- Suppose we seek the optimal alignment between the following two sequences.
- The score will count +1 for each match and -1 for each mismatch or indel.
- Remember: Optimal means an alignment with highest possible score.
  - And there may not be one unique such alignment, there may be a multi-way tie for first-place.

GGATGCG
GATTACA

Needleman-Wunsch

Put the two sequences in a table like this



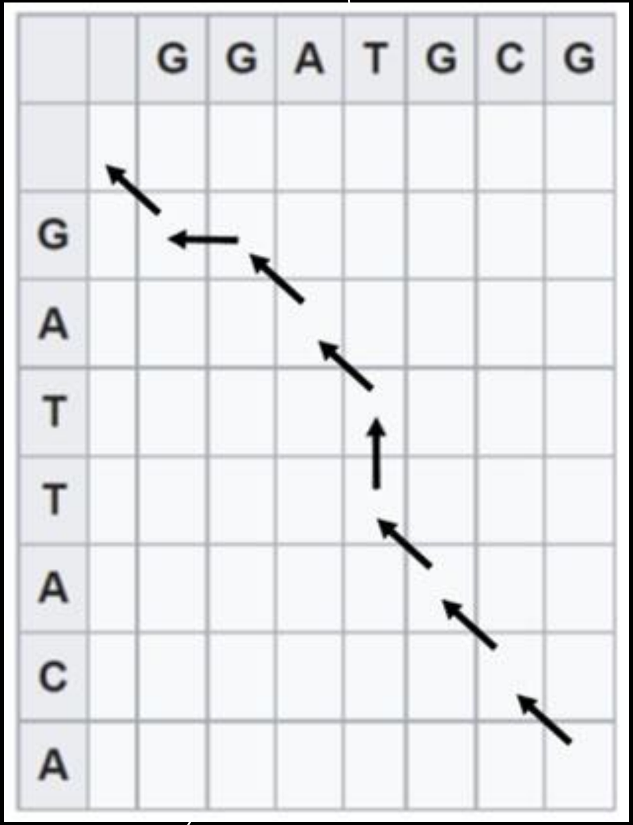
GGATGCG

GATTACA

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

Encoded Alignment

- A path from the bottom right cell to the top left encodes a particular alignment.



- **Diagonal arrow:** match
- **Vertical arrow:** indel in the first (horizontal) sequence
- **Horizontal arrow:** indel in the second (vertical) sequence.

```
GGAT-GCG
G-ATTACA
```

Encoded Alignments

- There’s a one-to-one correspondence between paths through the table and alignments.
  - Subject to one restriction: an indel is not allowed to align with an indel.

- How do we find the path(s) that represents alignment(s) with the highest possible score?
  - Note that it might not be unique, there could be more than one optimal alignment.

Filling in the Table

- We fill in the table with the best possible score of an alignment up to that point in both sequences.

- So, for example, the cell labeled X has the optimal score of an alignment between GG and GATTA

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

Filling in the Table

- The entries in the first row and first column are obvious.

- We proceed from here by filling in new cells for which three adjacent cells are already filled in:
  - The one above
  - The one to the left
  - The one at the upper left diagonal

		G	G	A	T	C	C	G
	0	-1	-2	-3	-4	-5	-6	-7
G	-1							
A	-2							
T	-3							
T	-4							
A	-5							
C	-6							
A	-7							

Filling in the Table

- There's only one such entry in the table so far, the one marked with an X.
- There are three ways to get to this cell.
  - From the left
  - From above
  - From the diagonal
- In each case we compute the running score.

		G	G	A	T	C	C	G
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	X						
A	-2							
T	-3							
T	-4							
A	-5							
C	-6							
A	-7							

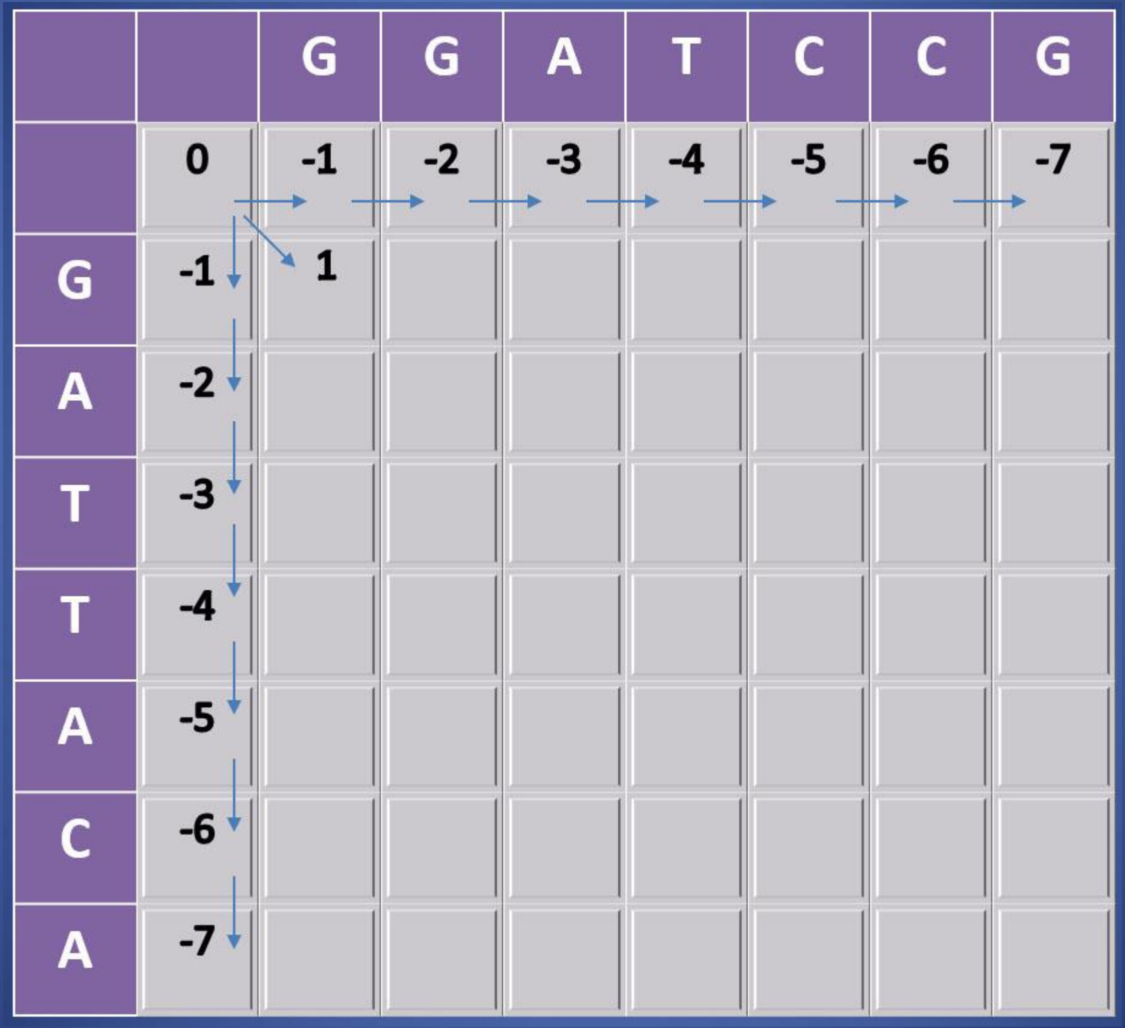
Filling in the Table

- If we come from the diagonal, the contribution to the score is +1 or -1 depending on whether the letters match.
- If we come from above or from the left, the only contributor to the score is one indel. So, we add -1 to the running score.
- The three possibilities for X then are:
  - Diagonal: 0 + 1 = 1
  - From the Left: -1 -1 = -2
  - From above: -1 -1 = -2

		G	G	A	T	C	C	G
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	X						
A	-2							
T	-3							
T	-4							
A	-5							
C	-6							
A	-7							

Filling in the Table

- The winner is the Diagonal with +1.

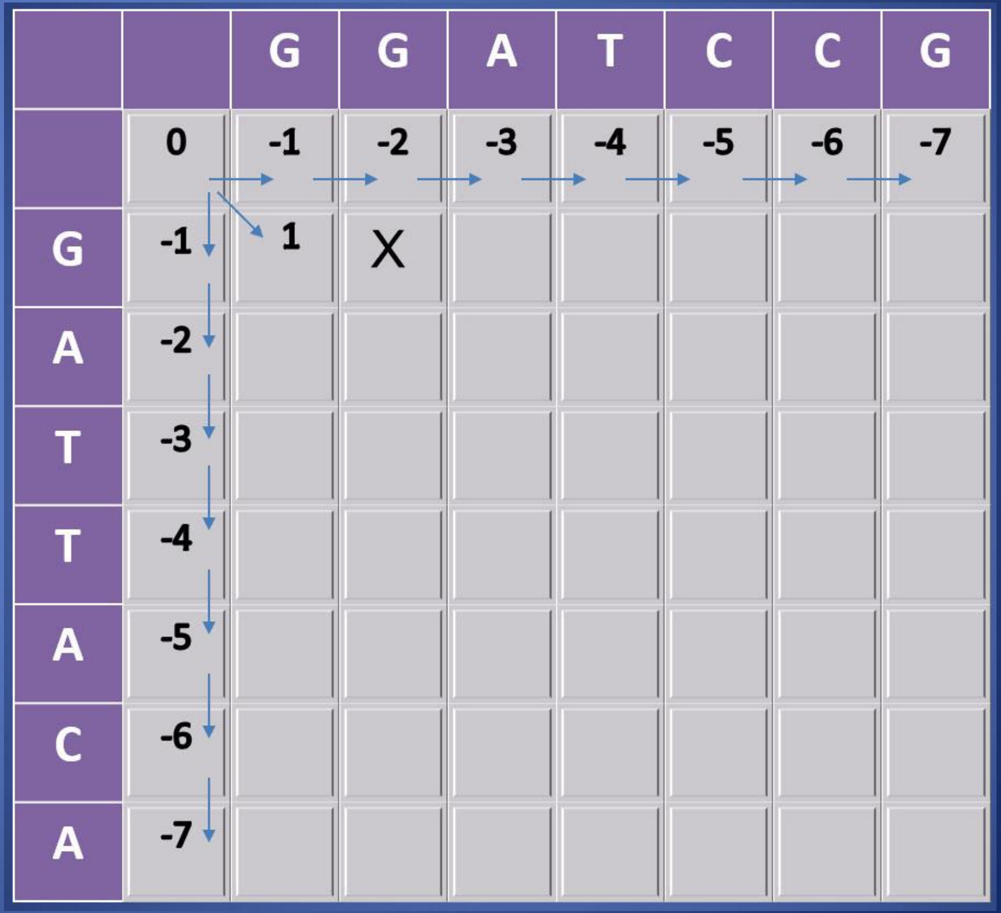


		G	G	A	T	C	C	G
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	1						
A	-2							
T	-3							
T	-4							
A	-5							
C	-6							
A	-7							

Filling in the Table

- Let’s do one more, the one labeled *X*.

- This is another match, G against G.
- So, the three possibilities are:
  - Diagonal: -1 +1 = 0
  - From the Left: 1 - 1 = 0
  - From Above: -2 - 1 = -3



		G	G	A	T	C	C	G
	0	-1	-2	-3	-4	-5	-6	-7
G	-1	1	X					
A	-2							
T	-3							
T	-4							
A	-5							
C	-6							
A	-7							

Filling in the Table

- It’s a tie.
- In this case we draw both arrows.
- If the traceback passes through this cell, then there will be more than one optimal alignment.

		G	G	A	T	C	C	G
	0	→ -1	→ -2	→ -3	→ -4	→ -5	→ -6	→ -7
G	↓ -1	↘ 1	↘ → 0					
A	↓ -2							
T	↓ -3							
T	↓ -4							
A	↓ -5							
C	↓ -6							
A	↓ -7							

Smaller Example

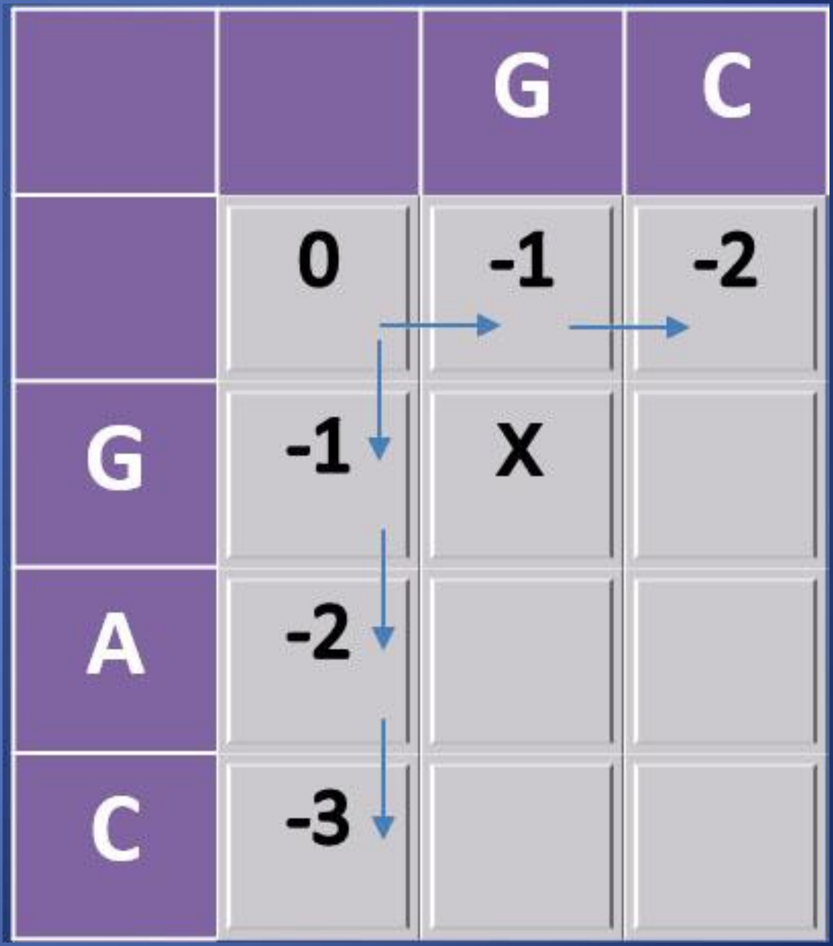
- It would take an hour to fill in that whole table by hand.
- So, let’s work a very simple example.
- GC aligned to GAC

		G	C
G			
A			
C			

		G	C
	0	-1	-2
G	-1		
A	-2		
C	-3		

Filling in the Table

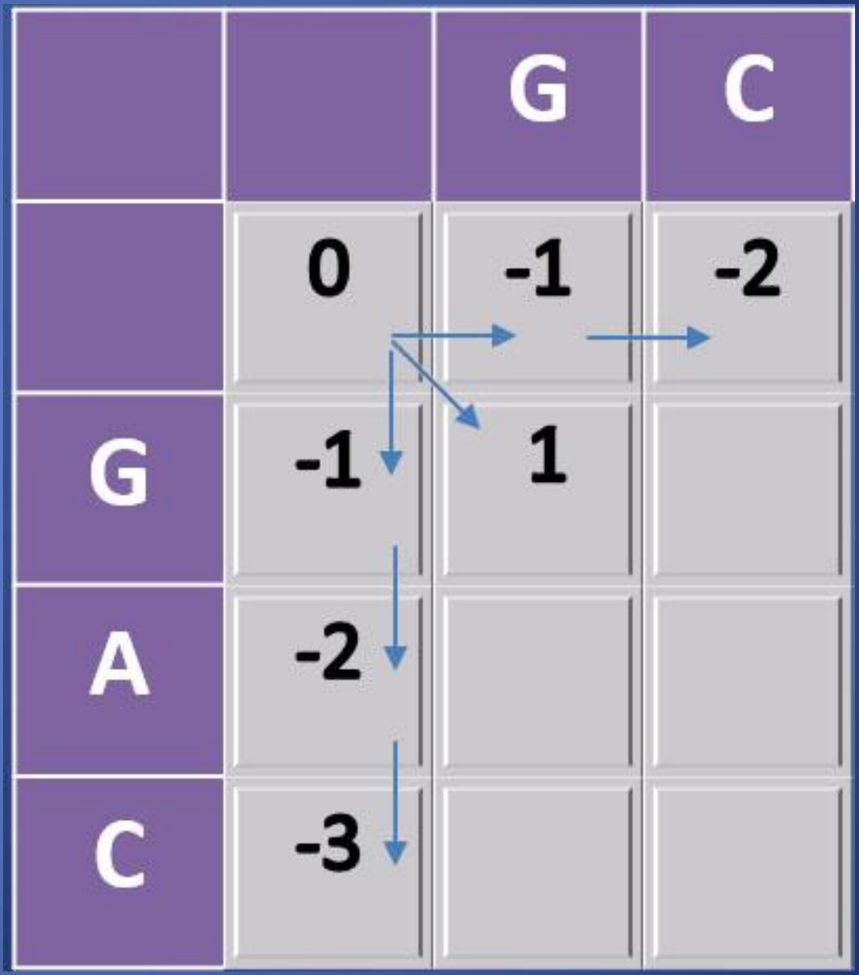
- The cell in marked with an *X* represents a match:
  - G in the row and G in the column.
- So, this adds +1 to the score.
  - Diagonal: 0 + 1 = 1
  - From the Left: -1 -1 = -2
  - From Above: -1 -1 = -2
- The maximum is +1 (coming from the diagonal direction).



		G	C
	0	-1	-2
G	-1	X	
A	-2		
C	-3		

Filling in the Table

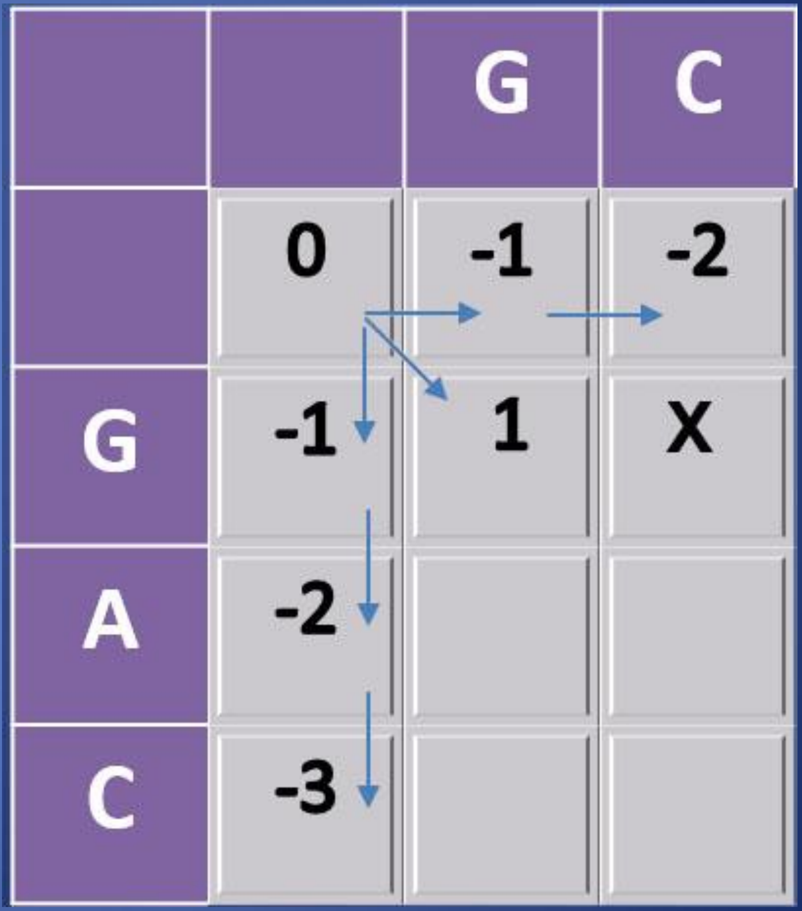
- We annotate the table like this, keeping track of the direction that achieved the maximum.



		G	C
	0	→ -1	→ -2
G	↓ -1	↘ 1	
A	↓ -2		
C	↓ -3		

Filling in the Table

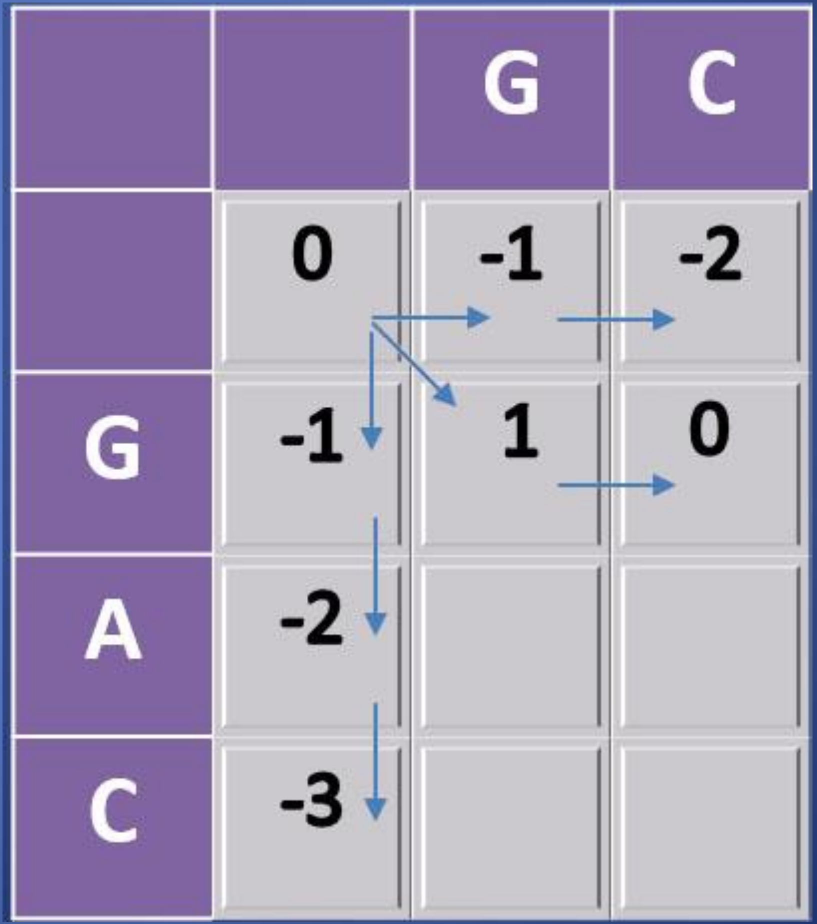
- Next fill in the cell labeled *X*.
- Here the cell represents a mismatch G in the row to C in the column.
- So, this adds -1 to the score.
- The three directions give:
  - Diagonal: -1-1=-2
  - Left: 1-1=0
  - Above: -2-1=-3
- The maximum = 0 and comes from the left direction.



		G	C
	0	-1	-2
G	-1	1	X
A	-2		
C	-3		

Filling in the Table

- We annotate the table like this.

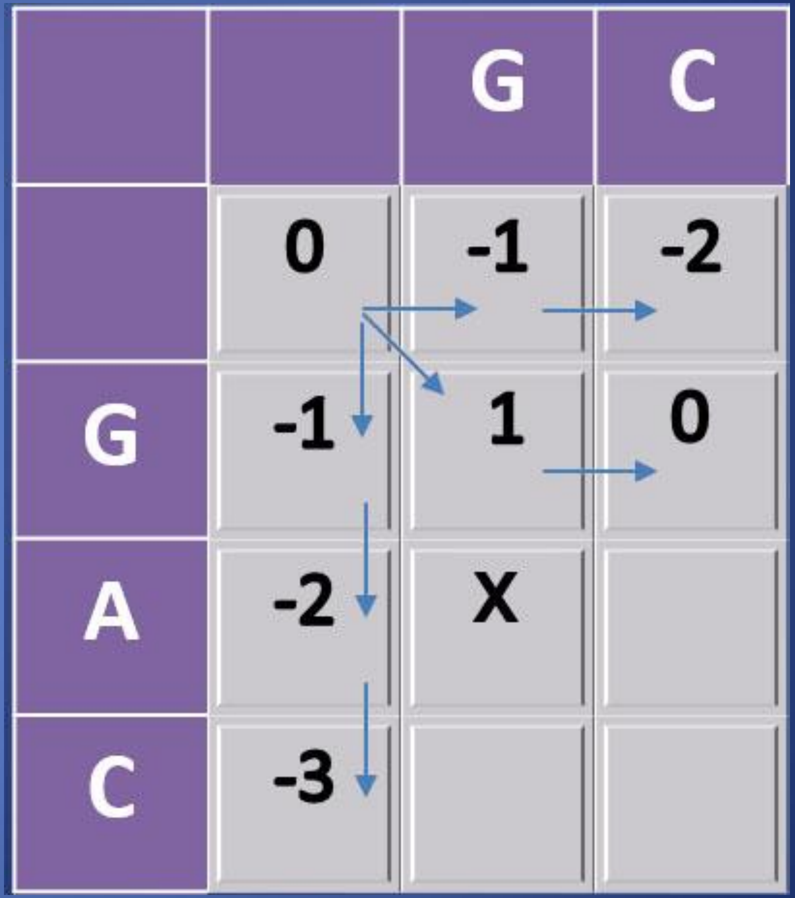


		G	C
	0	-1	-2
G	-1	1	0
A	-2		
C	-3		

Filling in the Table

- The cell marked with an *x* is the only one that can be filled in next.

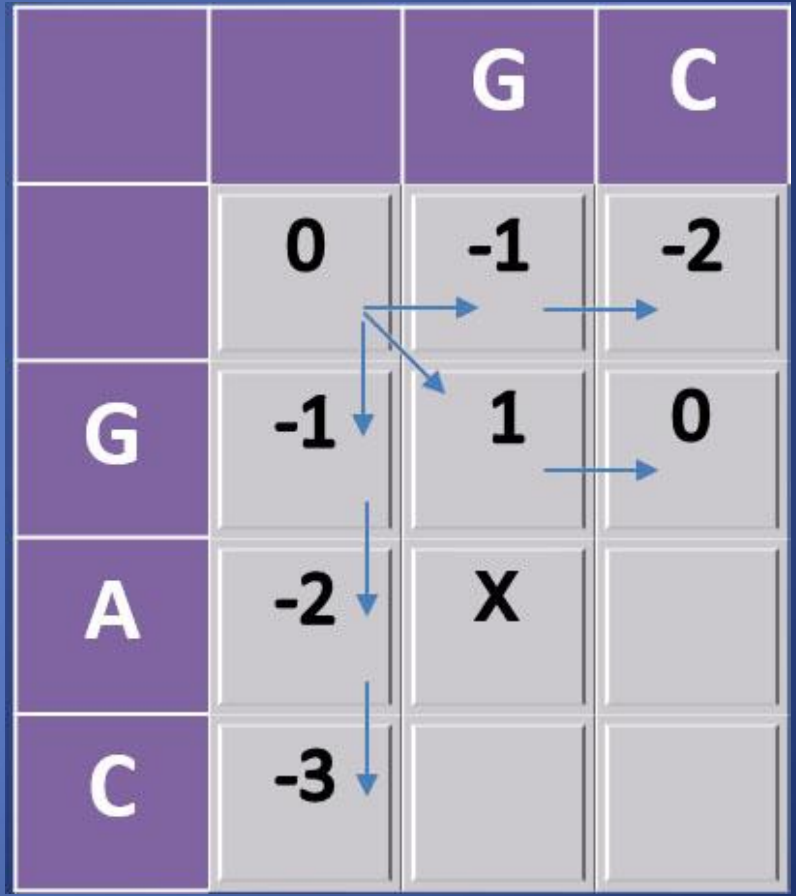
- This cell represents a mismatch G in the column and A in the row.



		G	C
	0	-1	-2
G	-1	1	0
A	-2	X	
C	-3		

Filling in the Table

- Since it’s a mismatch, it adds -1 to the score.
- The three directions give:
  - Diagonal: -1-1=-2
  - Left: -2-1=-3
  - Above: 1-1=0
- The smallest is 1-1=0 coming from the cell above.

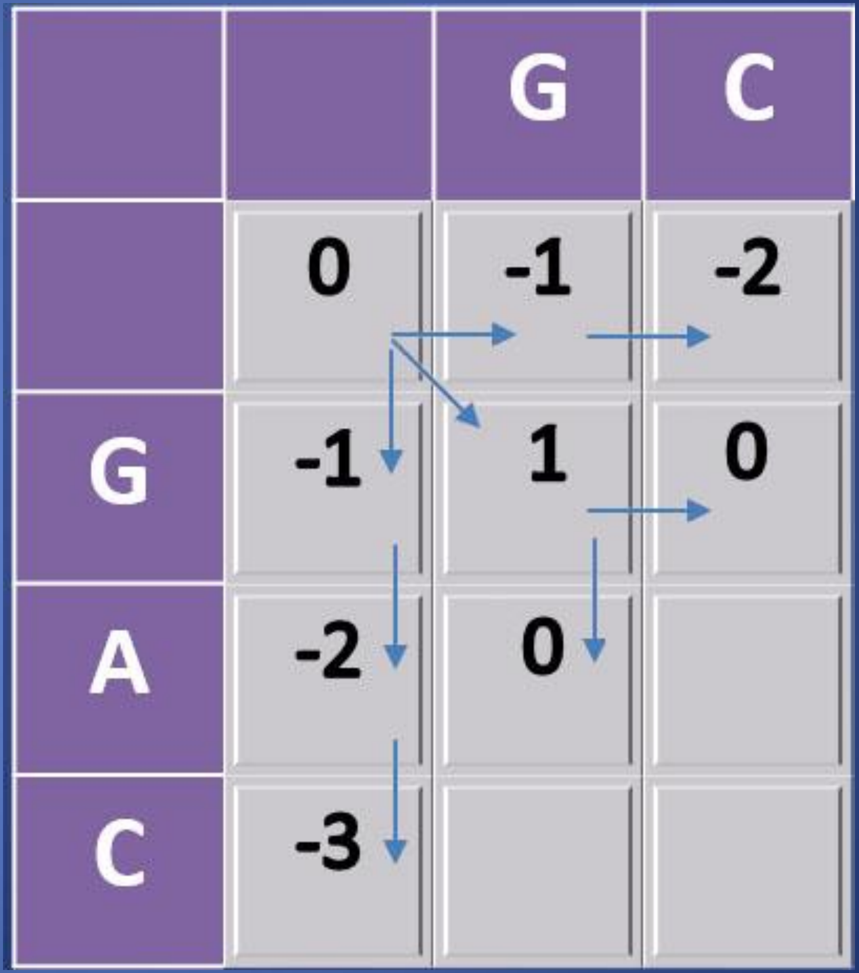


		G	C
	0	-1	-2
G	-1	1	0
A	-2	X	
C	-3		

Filling in the Table

- That gets us here.

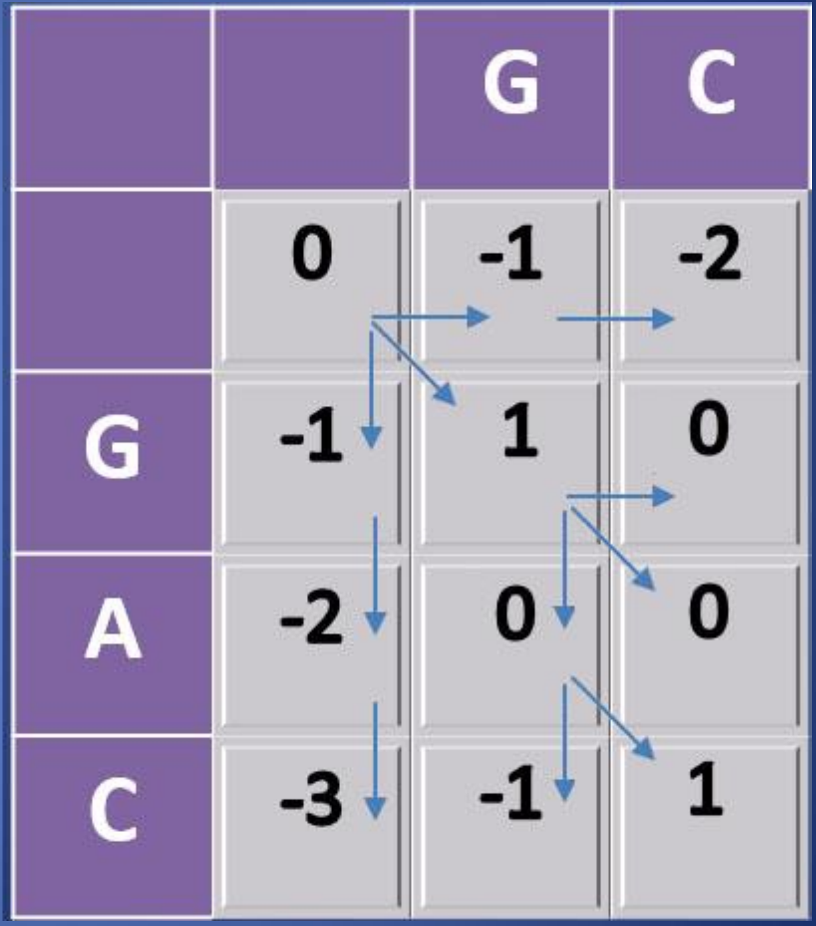
- Recall each cell contains the maximum of three numbers.



		G	C
	0	-1	-2
G	-1	1	0
A	-2	0	
C	-3		

Filling in the Table

- Filling in the remaining cells gives this.
- The value in the bottom right cell gives the optimal score of 1.
- Tracing back will give us the actual alignment.



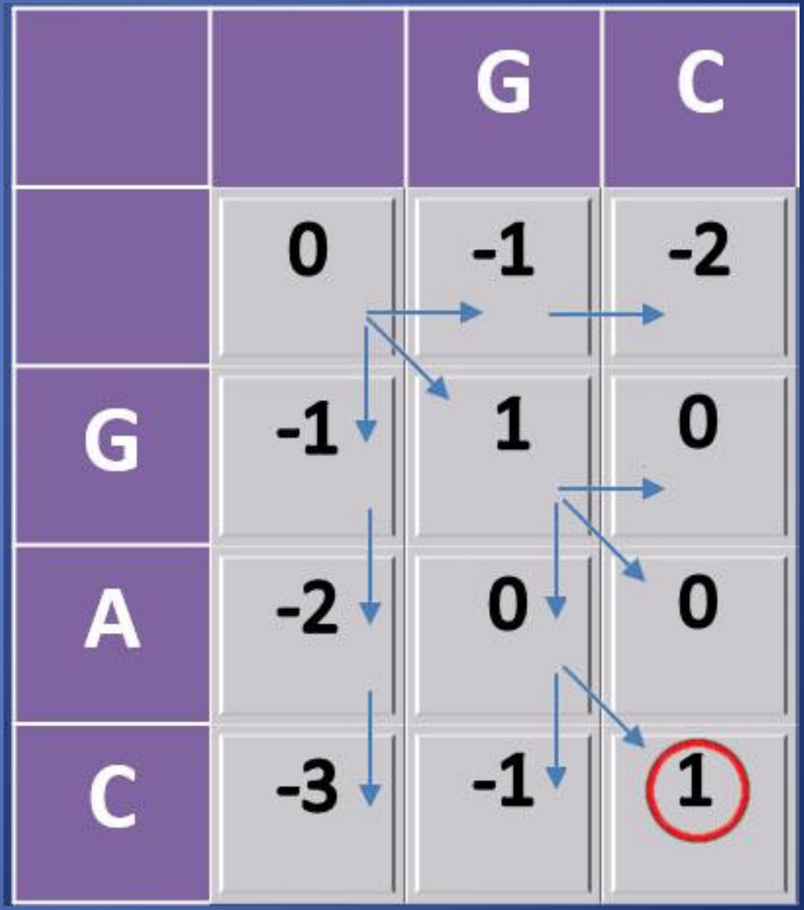
		G	C
	0	-1	-2
G	-1	1	0
A	-2	0	0
C	-3	-1	1

Tracing Back

- Start at the bottom right cell.
- This cell was reached by a diagonal arrow, so that means match/mismatch (as opposed to indel).
- This gets us here:

C

C

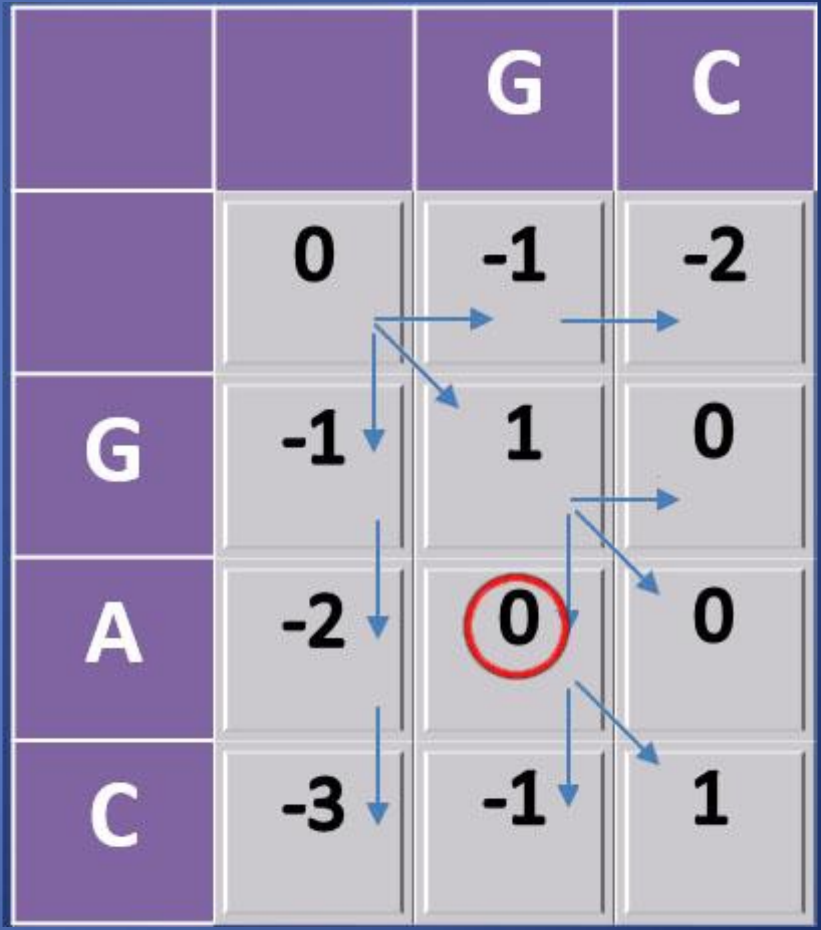


		G	C
	0	-1	-2
G	-1	1	0
A	-2	0	0
C	-3	-1	1

Tracing Back

- Follow the arrow to the cell circled in red.
- It was a vertical arrow that got us to this cell, so that's an indel.
- That gets us to here:

```
- C
A C
```



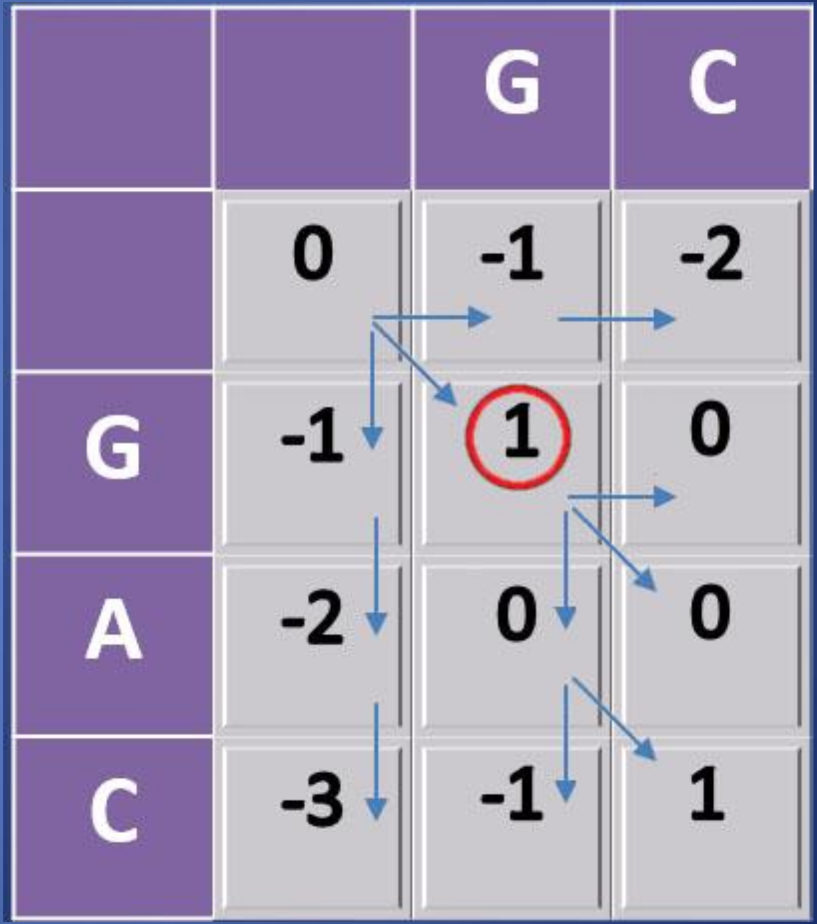
		G	C
	0	-1	-2
G	-1	1	0
A	-2	0	0
C	-3	-1	1

Tracing Back

- Follow the arrow to the cell circled in red.
- It was a diagonal arrow that got us to this cell, so that's a match/mismatch.
- That gets us to here:

G - C

G A C



		G	C
	0	-1	-2
G	-1	1	0
A	-2	0	0
C	-3	-1	1

Other Scoring Schemes

- In DNA, an A is more likely to mutate into a G than to a T.
- And any substitution is more likely than an indel.
- An indel of length 2 is not twice as unlikely as one of length 1.
- These facts argue for a more complicated scoring scheme.

Example where Score Matters

- Suppose we are to align ACGT with AGCT
- Consider two possible alignments:

Alignment #1	Alignment #2
ACGT	A - CGT
AGCT	AGC - T

- Scoring Scheme 1:
  - Match: +3, Mismatch: -3, Indel: -2
  - Alignment #1 score = 0, Alignment #2 score = 5
- Scoring Scheme 2:
  - Match: +3, Mismatch: -3, Indel: -5
  - Alignment #1 score = 0, Alignment #2 score = -1

- For scoring scheme #1, Alignment #2 is better.
- For scoring scheme #2, Alignment #1 is better.

Linear Gap Penalties

Think of the indel part of the score as -2K for a gap of length K.

This is called an ‘linear gap penalty’ since it grows proportionally with the length.

In this scoring scheme, larger numbers are better.

Using this scheme does not complicate Needleman-Wunsch.

Affine Gap Penalties

As we noted, it's less likely to start a gap than to extend one.

A linear gap penalty does not capture this.

A more general system that does capture this is to score -2 - K for a gap of length K+1.

Again, larger numbers are better.

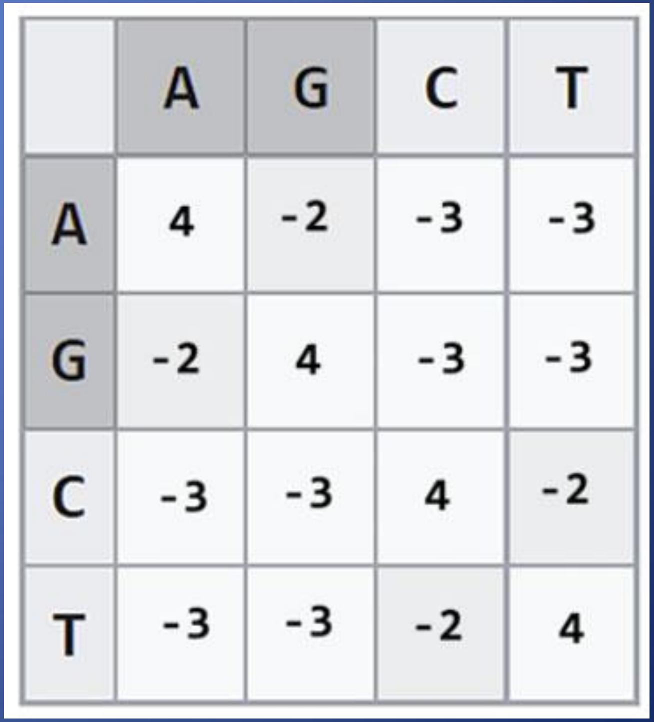
This is called an 'affine gap penalty'.

The penalty to initiate a gap (-2) is less than the penalty to extend it by a base (-1).

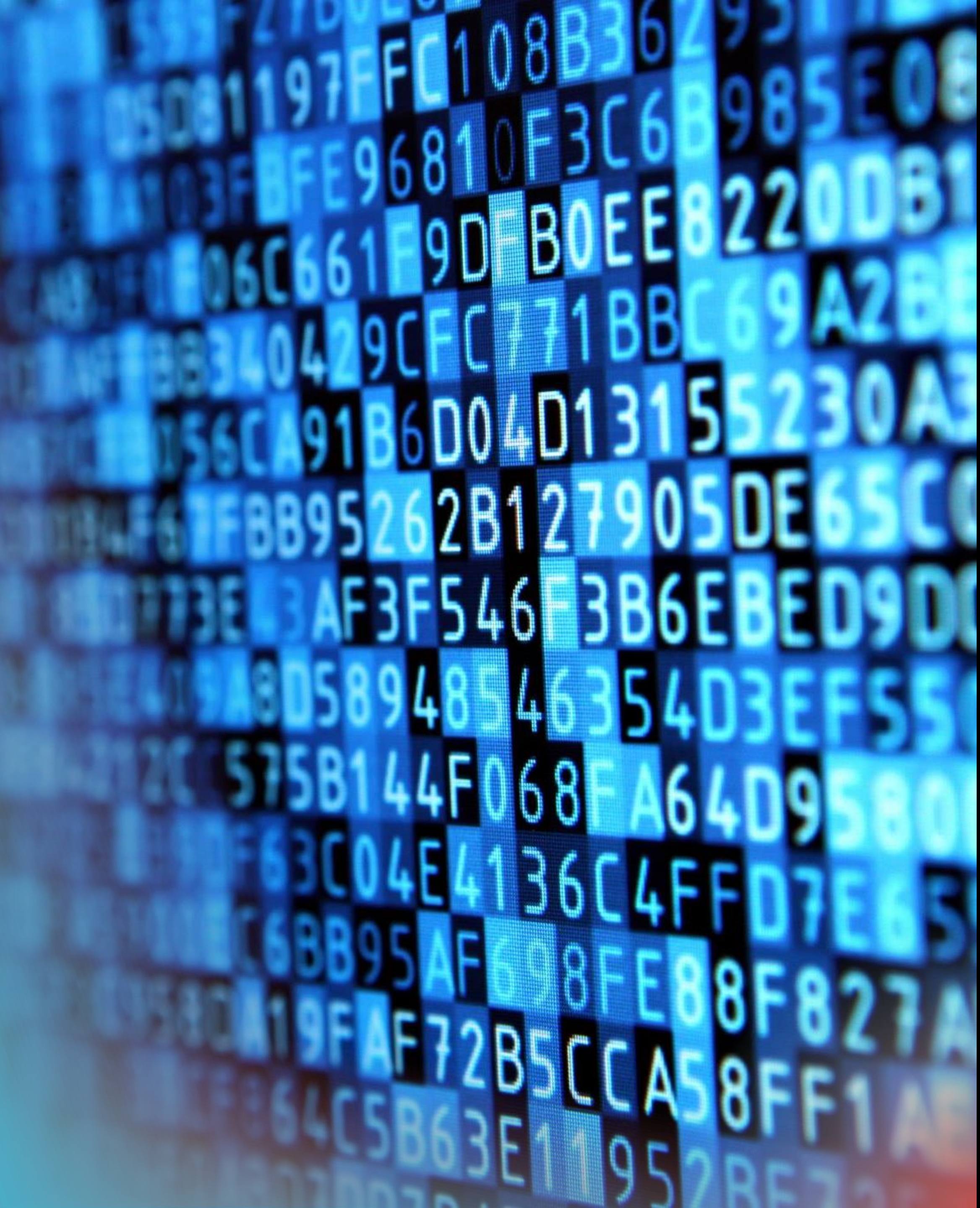
Using this scheme *does* complicate Needleman-Wunsch, because the value in a cell no longer depends on just the three neighbors.

Substitution Matrices

- In general, the substitution of one base for another can be different for each pair of nucleotides. In this case we can specify them in a so-called "substitution matrix"
- In this example substitutions of purines and pyrimidines are penalized greater than purines with purines or pyrimidines with pyrimidines.

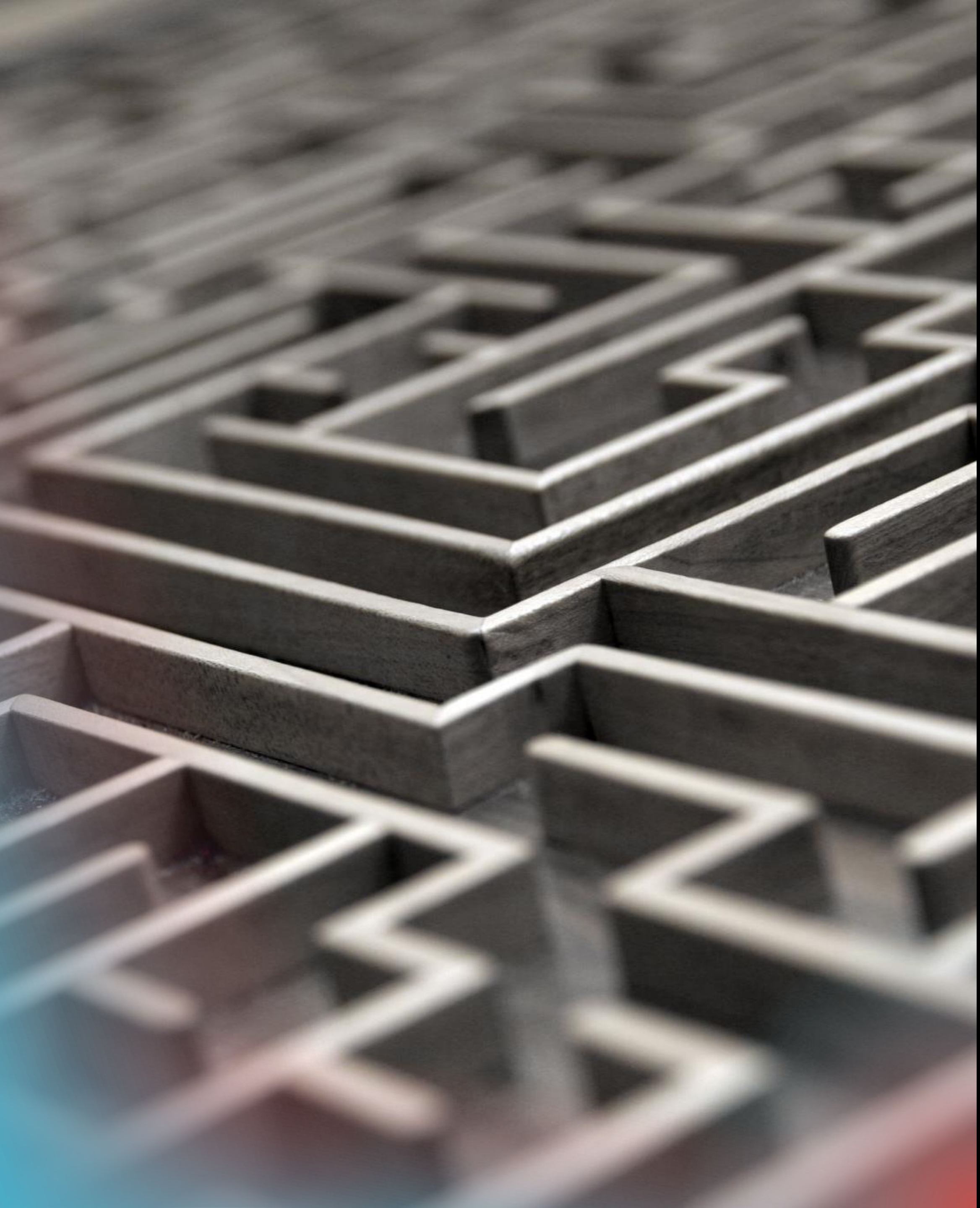


	A	G	C	T
A	4	-2	-3	-3
G	-2	4	-3	-3
C	-3	-3	4	-2
T	-3	-3	-2	4



Algorithmic Complexity

- Computers are fast, so why do we need to be clever anymore?
- Why not just write a program to do an exhaustive search through all alignments, to find the optimal ones?
- Unfortunately, computers are not that fast.
- An exhaustive search for sequences of length 1000 would take to the end of the universe.



Complexity

- Even if we're just aligning short sequences, we often must do it millions of times in a row.
  - For example, a high-throughput sequencing experiment can produce 100,000,000 sequence reads, which all need to be aligned to the (very large) genome.
- We need a way of characterizing the efficiency of an algorithm, so we can determine which of two proposed algorithms is faster.

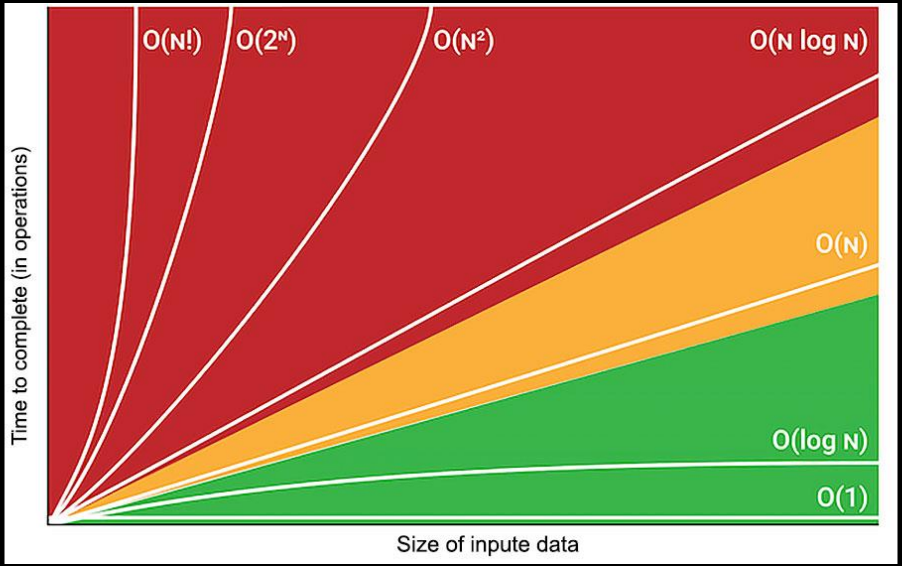
Big O

- Let *f(t)* be a function of time, where $t \geq 0$.
- Let *g(t)* be another function of time, $t \geq 0$.
- We say "*f* is big O of *g*" if the limit

$$\lim_{t \to \infty} \frac{f(t)}{g(t)}$$

is finite.

  – We write *f* = O(*g*).



- More generally, we require that there exists positive constants *M* and *N* such that

$$\left| {}^{f(t)}/_{g(t)} \right| < \mathrm{M} \text{ for all } t > N$$

  – For example, *f(t)* = cos(*t*), *g*(*t)* = 1. The limit does not exist, but cos*(t)* is still big O of *1*.

Example

Let $f(t) = t^3+4t+6$

Then f is big O of t^3

In fact, f is big O of t^n for any $n \geq 3$.

Meanwhile f is *not* big O of t^2.

$f(t)/t^2 = t + 4/t + 6/t^2 \rightarrow \infty$

It's not really about time

- The time variable *t* can really be any parameter that takes values in $[a, \infty)$ for some a.
  - For our purposes we'll replace *t* with *n*, the length of the larger of the two sequences to be aligned.

- And let *f(n)* be the number of operations required to perform an alignment.

- If *f* is big O of *n,* then doubling the length of the sequences being aligned only doubles the number of operations required.

- To date nobody has found an algorithm that is that fast and it's probably been proven impossible.

Complexity of Alignment

- If an algorithm is big O of n^2 then doubling the length of the sequences increases the number of operations by four-fold.

- This may sound problematic, but quadratic growth is usually considered good.

- What you really don't want is the algorithm to be no better than big O of 2^n.

- That's exponential growth that gets out of hand very quickly.

```
GPIEFDCPDDVYILDQAEEAGHDLPYSCRAGSCSSCAGKIAGGAVDQTD
GPIEFDCPDNVYILDQAEEAGHDLPYSCRAGSCSSCAGKIAGGAVDQTD
GPIEFECPDDVYILDQAEEEGHDLPYSCRAGSCSSCAGKVTAGSVDQSD
GPIEFECPDDVYILDQAEEEGHDLPYSCRAGSCSSCAGKVTAGTVDQSD
GTQEFECPSDVYILDHAEEVGIDLPYSCRAGSCSSCAGKVVGGEVDQSD
GPQEFDCPDDVYILDHAEEVGIELPYSCRAGSCSSCAGKVVNGNVNQED
GKQELECPDDVYILDAAEEAGIDLPYSCRAGSCSSCAGKVTSGSVNQDD
GNVEFQCPDDVYILDAAEEEGIDLPYSCRAGSCSSCAGKLKTGSLNQDD
GEQEVECDDDVYVLDAAEEAGIDLPYSCRAGSCSSCAGKVVSGSVDQSD
GELEVECDDDVYVLDAAEEAGIDLPYSCRAGSCSSCAGKVVSGSVDQSD
GEQEVECDDDVYVLDAAEEAGIDLPYSCRAGSCSSCAGKVVSGFVDQSD
GEQEVECEEDVYVLDAAEEAGLDLPYSCRAGSCSSCAGKVVSGSIDQSD
GEQEVECEEDVYVLDAAEEAGLDLPYSCRAGSCSSCAGKVVSGSIDQSD
```

High Complexity Problems

- If a problem has exponential complexity, or does not have polynomial complexity of a reasonably low degree
  - then it may simply be impractical to use an algorithm that guarantees an optimal solution.

- Consider for example the problem of aligning not just two sequences but *N* sequences in a so-called multiple sequence alignment.
  - This problem gets out of hand very fast.

Heuristics

- Therefore, often algorithms are developed that do not guarantee to find an optimal solution but at least find one that is reasonably close.
- Sometimes we can quantify how close, and sometimes we just proceed on faith.
- We will encounter numerous heuristic algorithms in this class.

Complexity of Needleman-Wunsch

- Without going into too much detail, notice how the Needleman-Wunsch algorithm requires filling in a table.
- If the sequences both have length *n,* then the table has on the order of n^2 cells.
  - $(n+1)^2$ is "on the order of" n^2, that just means they're both big O of each other.
- It takes at least four operations to populate one cell:
  - Compute three numbers and take the maximum.
- But notice how that number of operations is basically constant, it's the same for all cells.

Complexity of Needleman-Wunsch

- Suppose it takes *f*(*n*) operations to align two sequences of length *n* by Needleman-Wunsch.
- Suppose there are *k* operations involved in populating one cell, and that's the same *k* for all cells.
- You should be able to convince yourself, at least intuitively, that *f* is big O of n^2 and in fact

$$\lim_{n\to\infty} \frac{f(n)}{n^2} \to k$$

Complexity of Needleman-Wunsch with a Linear Gap Penalty

- If we use a linear gap penalty, then it requires more operations per cell.
- But that does not necessarily mean the complexity increases beyond $O(n^2)$.
- For each cell one may need to maintain the length of the current gap.
  - But the number of operations per cell is still constant.
  - So, it's still $O(n^2)$ just with a higher limiting constant.