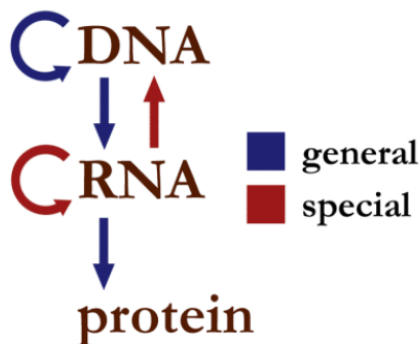


Sequence and Dynamic Programming (L3)

A) Sequence Data



DNA store genetic information

RNA helps express information and turns to protein.

Phenotype = Genotype + Environment

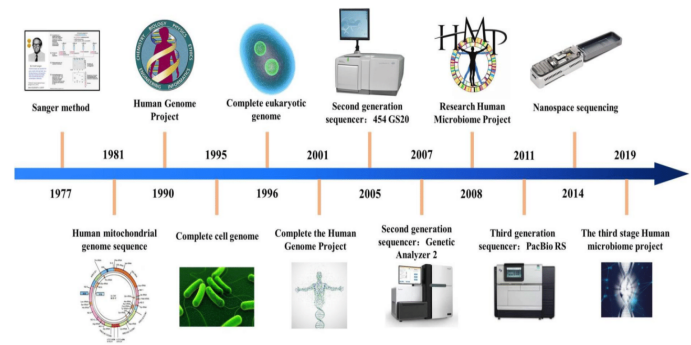
Phenotype : How we look, for example skin colours.

Genotype : DNA sequence / gene sequence. 99% dna sequence of each individuals is the same

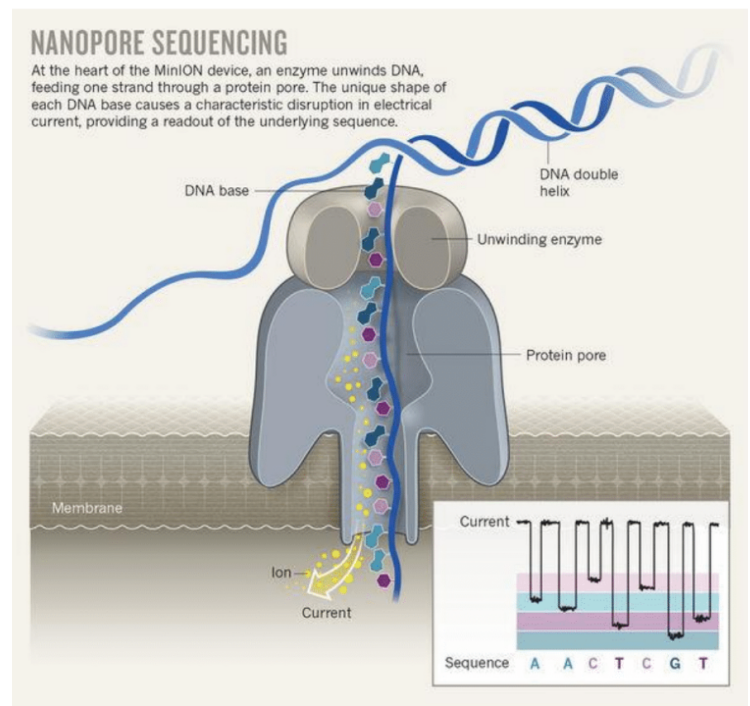
Environment : That affect the gene expression

1) Sequence Type

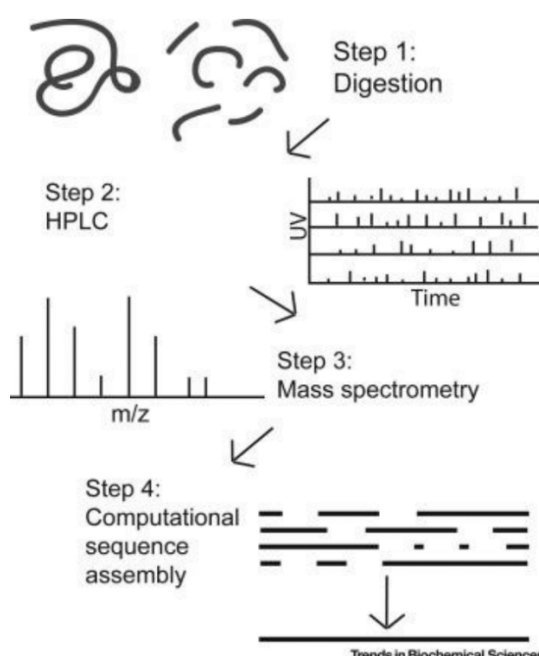
No.	Sequence		How to get the sequence
	DNA Sequence	• ATCG • Double Strand • There are 3 billions of this pair	As years go, there are more advanced ways to achieve sequence and it still under development From short read to long reads
	RNA Sequence	• AUCG • Single Strand	



- Nanopore Sequencing

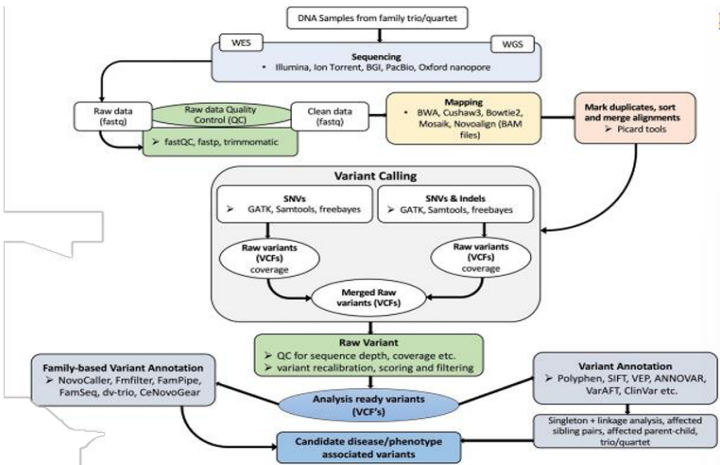


- ☐ DNA goes through **chemical pore** (like when we go through a security gate) with each **base generating electric current change**.
- ☐ Sequencing by detecting electric change from different bases (ATCG).
- ☐ Manage to sequence **long sequences up to 4 Mb** compared to the older generation that can only up to 1000 bp.
- ☐ But have a high **error rate at 5%**.
- ☐ Still under active development.

	Protein Sequence	• 20 amino acids • Multiple sequence alignment	Protein Sequencing  Step 1: Digestion Step 2: HPLC Step 3: Mass spectrometry Step 4: Computational sequence assembly Trends in Biochemical Sciences ☐ Based on Mass Spectrometry (MS) ☐ The process starts by breaking it down into short pieces then being determined by MS based on weight before being assembled into raw sequence (similar like a jigsaw puzzle).

2) Raw Data Handling

No.	Sequence	Process
-----	----------	---------

1	DNA Sequence	 <pre> graph TD A[DNA Samples from family trio/quartet] --> B[Sequencing] B --> C[Raw data fastq] C --> D[Raw data Quality Control QC] D --> E[Clean data fastq] E --> F[Mapping] F --> G[Mark duplicates, sort and merge alignments] G --> H[Variant Calling] H --> I[Raw variants VCFs coverage] I --> J[Merged raw variants VCFs] J --> K[Raw Variant] K --> L[Analysis ready variants VCFs] L --> M[Candidate disease/phenotype associated variants] L --> N[Variant Annotation] N --> O[Singleton + linkage analysis, affected sibling pairs, affected parent-child, trio/quartet] O --> M </pre> The flowchart illustrates the process of DNA sequencing and variant calling. It starts with DNA samples from a family trio/quartet, which are then sequenced using WGS (Whole Genome Sequencing) or WES (Whole Exome Sequencing). The sequencing process involves Illumina, Ion Torrent, BG1, PacBio, or Oxford Nanopore. The raw data (fastq) is then processed through Raw data Quality Control (QC) using tools like fastQC, fastp, and trimmomatic. The cleaned data (fastq) is then mapped to the reference genome using tools like BWA, GSNAP, Bowtie2, Mosaik, or Novoalign (BAM files). The mapped data is then processed through Mark duplicates, sort and merge alignments using Picard tools. The resulting data is then processed through Variant Calling, which includes SNVs (Single Nucleotide Variants) and SNVs & Indels (Single Nucleotide Variants and Insertions/Deletions). The variant calling process involves Raw variants (VCFs) coverage, Merged raw variants (VCFs), and Raw Variant. The final output is Analysis ready variants (VCFs), which are then used for Candidate disease/phenotype associated variants. The process also includes Family-based Variant Annotation (NovoCaller, Fmfilter, FamPipe, FamSeq, dy-trio, CeNovoGear) and Variant Annotation (Polyphen, SIFT, VEP, ANNOVAR, VarAFT, ClinVar etc.). The final output is Singleton + linkage analysis, affected sibling pairs, affected parent-child, trio/quartet. • Quality Control Delete unneeded sequence • Map reads to reference genome Mark duplicate, sort and merge alignment to detect mutation and variant • Variant Calling • Phenotype associated variants Link between genetic variant to phenotype
2	Protein Sequence	By sequence comparison such as multiple sequence alignment which if the sequence similar: ☐ Sequence - to Structure -to Function Paradigm Similar sequence -> Similar structure -> Similar function ☐ Homology Common ancestor

3) Sequence Alignment

To determine the similarity between sequences and identify regions of similarity

Why is it important?:

- For biomolecular function and property prediction
Similar sequence -> Similar structure -> Similar function
- For evolution, identifying conservative region, investing mechanism
Common ancestor

Pairwise sequence alignment

Maximise the similarity of the two sequences by inserting the gap in the sequences and score the alignments.

ATCG__ -> Even though its has the same formation sequence but the position of the gap
 __ATCG in the arrangement makes them very different if we calculated it from the
 alignment score directly which ended up being wrong.

How to do the scoring?

Scoring Type	Example
Match (identical bases)	• A-A • T-T • C-C • G-G
Mismatch (Substitution)	• A-T • T-G • C-A • etc...
Gap (Insertion / deletion)	• A-<u> </u> • T-<u> </u> • C-<u> </u> • G-<u> </u>

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

A G G C C G -> $2 + (-7) + 2 + 2 + (-10) + 2$
 A T G C _ G = -9

A G G C C G -> $2 + (-7) + 2 + 2 + (-7) + (-10)$
 A T G C G _ = -18

Higher the alignment score means the more similarity between the sequences.

How can we find the best alignment?

- enumeration

The straight forward solution is to **enumerate down all** the possibilities then **combine the score alignment** then find the **highest**.

But the problem is there are too many alignments

$$\binom{2n}{n} = \frac{(2n)!}{(n!)^2}$$

If $n = 300$
There are 7×10^{88} possibilities

- Dynamic Programming
The solution to enumeration problem

B) Dynamic Programming

- Break main problems into **sub-problems**
- Solve the sub-problems optimally and **recursively**
- Utilize these optimal solutions to build the best overall solution for the initial problem

Step:

Scoring matrix:

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Gap penalty = -10

		A	C	C	G
A					
C					
G					

- 1) Fill the **first row and first column** which is the **gap penalties** and in this example is (-10)

		A	C	C	G
	0	-10	-20	-30	-40
A	-10				
C	-20				
G	-30				

As you go toward the left or down side of the gap penalties , **add the gap penalties from the previous box number.**

- 2) Next we start to fill the box on the diagonal of the box (0)

0	-10
-10	

To fill the box, there are 3 options to be considered, **top, left, and diagonal.**

Top

Previous number + gap
 $-10 + (-10)$
 -20

Left

Previous number + gap
 $-10 + (-10)$
 -20

Diagonal

Previous number +match/mismatch
 (Since in this case is A-A so is a match)
 $0 + 2$
 2

From the results, we got -20 , -20 and 2. Here we **choose the biggest number** to be put in the box which is 2. Also **put the arrow** to indicate which side the number comes from.

		A	C	C	G
	0	-10	-20	-30	-40
A	-10	2			
C	-20				
G	-30				

3) Do it to the other empty box until it fills the whole table.

		A	C	C	G
	0	-10	-20	-30	-40
A	-10	2	-8	-18	-28
C	-20	-8	4	-6	-16
G	-30	-18	-6	-3	-4

4) Translate the arrow into alignment starting from the bottom right. It may have more than 1 alignment. Horizontal arrow translates to a gap while diagonal arrow means both sequences proceeding by one letter.

In this case there are 2 alignment

A C C G	A C C G
A _ C G	A C _ G