

# CS 262 Lecture #2 Notes

---

## 23andMe.com

- 23andMe offers to analyze consumer's DNA so that one may better understand their own genetic information. With \$399, your genome is genotyped and the gathered information made available to you online.
- Genotyping
  - **Single nucleotide polymorphism (SNP)** variation occurs when there is a single nucleotide/letter substitution between two individuals. Between two individuals, SNPs occur once every 1000 nucleotides.
  - 99.9% of the DNA is identical amongst members of the same species. It is the remaining 0.1% – a large fraction of which are SNPs – that distinguishes one individual from another. Additionally, the locations of the common differences (ones that are hereditary) are largely fixed.
  - **Genotyping** uses biological assays to probe the locations of these common differences in the genome so as to determine the particular nucleotides that occupy these locations.
- **DNA sequencing** is the process of obtaining the nucleotide sequence of the entire genome. The process is more costly (~\$20,000), however is expected to decrease by a factor of ten every four years.

## RETROVIRUS

A virus is a sub-microscopic infectious agent that is unable to reproduce outside of a living cell. A **retrovirus** is a virus that contains an RNA genome and distinguishes itself from other RNA viruses because of its unique way of replication.

- The retrovirus consists of a flexible membrane – the envelope – with exterior **docking proteins** used to lock on and invade into host cells.

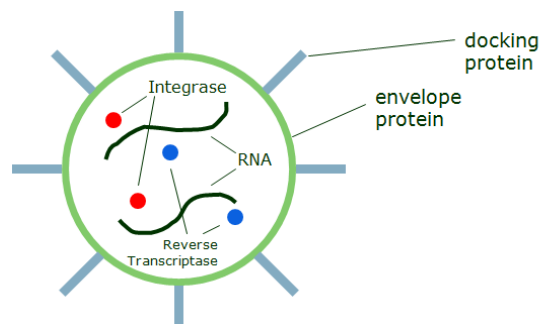


Figure 1: retrovirus structure

- Replication process
  1. After the retrovirus enters the host cell, its RNA and reverse transcriptase are released into the cell's cytoplasm. **Reverse transcription** occurs whereby reverse transcriptase uses resources from the host cell to transcribe the viral RNA to DNA. The **integrase** enzyme then inserts this retroviral DNA into the host's genome.

2. After its insertion into the host's genome, the retroviral DNA behaves like the host's own DNA. It instructs the host cell to manufacture all the components of the original retrovirus. After production, the viral components spontaneously assemble to form more retroviruses.
3. These retroviruses then "bud" out of the cell – taking some of the host's plasma membrane to form its envelope. The cycle repeats as these viruses go to find more host cells to invade.

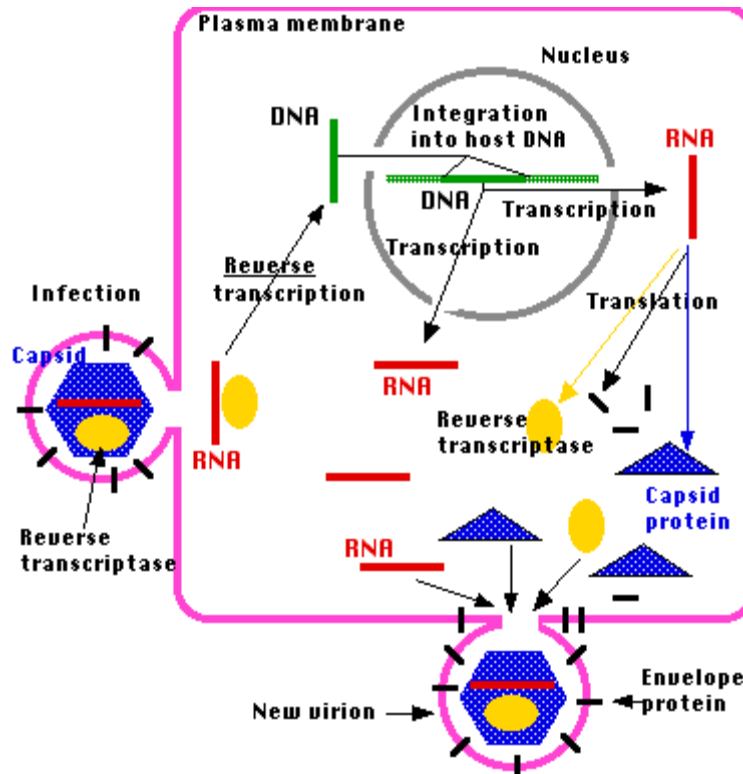


Figure 2: retrovirus invasion and replication

### EVOLUTION (DNA LEVEL)

The genomes of more than 1000 organisms have been sequenced. Through the study of the genome sequences we are able to construct an evolution tree by finding the common ancestors of different organisms.

- **Mutations** in the genome may be a result of replication errors during cell division or introduced by external damage (e.g. radiation). Repair mechanisms exist in our cells to catch and repair most changes. However, the effectiveness of such mechanisms decreases as we age, and changes in the DNA accumulates.
- DNA level evolution can be split into two general categories:
  1. **Sequence edits** include the deletion, insertion, and mutation of nucleotides.
  2. **Rearrangements** are larger changes in the DNA sequence which can be further refined into the following categories:

- a. **Inversion** refers to the event when an entire section of the DNA is reversed. The length of the affected DNA ranges from a few bases to larger regions of the chromosome that involve several genes.
- b. **Translocation** is the rearrangement of parts between nonhomologous chromosomes. The same event but between homologous chromosomes occurs regularly during meiosis and is referred to as crossover.
- c. **Duplication** is the duplication of a part of the chromosome.

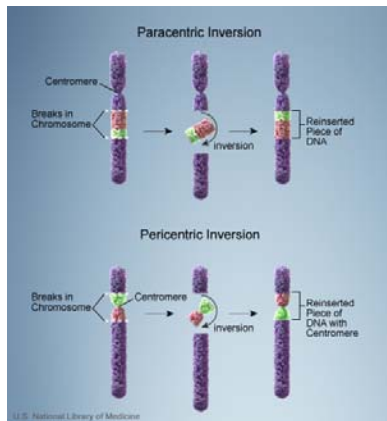


Figure 3: inversion

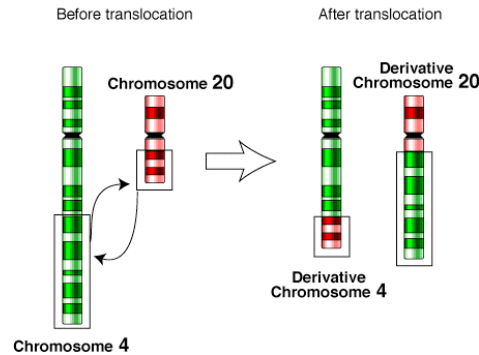


Figure 4: translocation

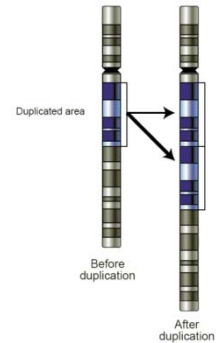


Figure 5: duplication

- There is a certain rate at which DNA errors occur and in general, the rate of error is inversely proportionally to the size of the change. However, not all DNA changes persist through time; for example, individuals with variations that lead to their death will not be able to propagate the change.
- **Conserved regions** – how conserved is a chromosomal region across different individuals is a strong indicator of the importance of the information encoded by this region.
  - By comparing the conserved regions common across organisms we can look for common biological traits maintained across the species (e.g. all mammals have placentas), and thereby assign such traits to the common ancestor of these organisms.
  - Similarly, chromosomal regions that are different across organisms (e.g. fish vs. mammal) is an indicator of an evolutionary event, one that brought on this difference.

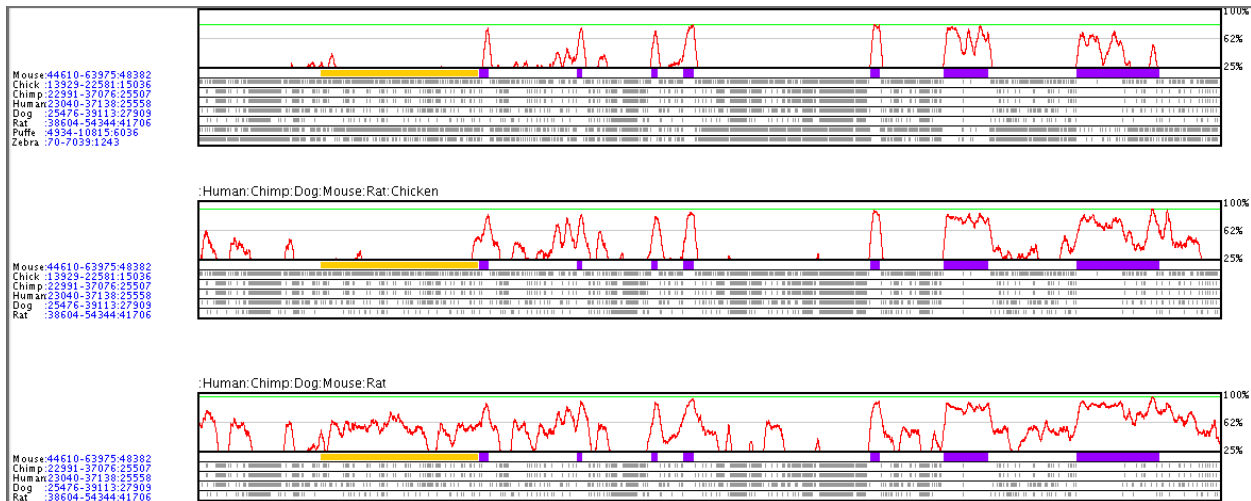


Figure 6: wiggle plot showing the regions of conservation (red line) across the genomes of various species

In the above figure, the red line height indicates the percentage of DNA conservation. Regions marked in purple are exons (coding regions). The yellow region marks the untranslated 5' end of the gene.

SEQUENCE ALIGNMENT

Sequence alignment reveals the differences between two string sequences through a strategic positioning of gaps so as to align one string sequence to another. Given two sequences  $x = x_1x_2 \dots x_n$  and  $y = y_1y_2 \dots y_m$ , an **alignment** is an assignment of gaps to positions  $0, \dots, n$  in  $x$ , and  $0, \dots, m$  in  $y$ , so as to line up each letter in one sequence with either a letter in the other sequence or a gap. In the end, we obtain two sequences of equal length. The following is an example of the alignment of two sequences:

```

AGGCTATCACCTGACCTCCAGGCCGATGCC
TAGCTATCACGACCGCGGTTCGATTTGCCCGAC
-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
TAG-CTATCAC--GACCGC--GGTCGATTTGCCCGAC

```

- **Scoring function** answers the question of “how good is an alignment?”
  - We define the function to **minimize** the **edit distance** between the sequences. An **edit** can either be an insertion/deletion of a gap or a mutation of a letter in the sequence, where the goal is to transform one sequence into the other.
  - Thus, a simple scoring function  $F$  can be defined as follows:
$$F = (\# \text{ matches}) * m - (\# \text{ mismatches}) * s - (\# \text{ gaps}) * d$$
    - + $m$  = match reward
    - $s$  = mismatch penalty
    - $d$  = gap penalty

The parameters  $m, s, d$  are user parameters and may be chosen through statistic studies of the impact of each edit or by machine learning methods.

- In general, scoring functions are a function of the number of matched, mismatched, and gapped base pairs. One can design more complicated scoring functions that are nonlinear, include more variables, and/or assign different weights to the edits according to statistics gathered from biological setting.
- Computing the optimal alignment
 

Given two strings  $x = x_1x_2 \dots x_n$  and  $y = y_1y_2 \dots y_m$ , we can envision a  $n \times m$  matrix, as in Figure 7, to assist in the computation of the optimal alignment. The  $i^{\text{th}}$  row in the matrix represents the  $i^{\text{th}}$  letter in  $x$ , and the  $j^{\text{th}}$  column in the matrix denotes the  $j^{\text{th}}$  letter of  $y$ .

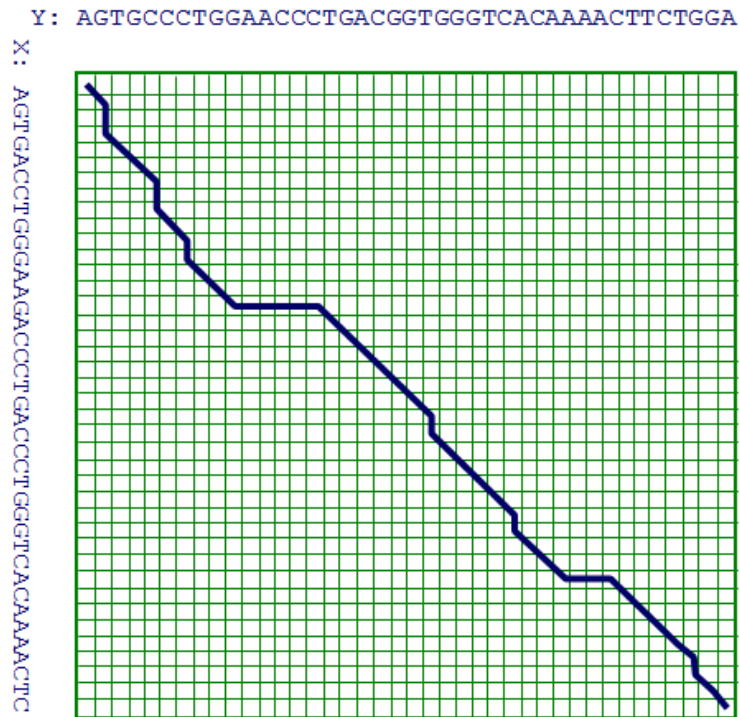


Figure 7: Needleman-Wunsch matrix

- An alignment can be found by tracing a path beginning from the top left corner that leads to the bottom right corner. Where, at cell  $(i, j)$  of the matrix,
  1. A **downward move** to cell  $(i + 1, j)$  is equivalent to adding a gap to the horizontal  $y$  sequence that aligns with the  $i^{\text{th}}$  letter in the vertical  $x$  sequence.
  2. A **rightward move** to cell  $(i, j + 1)$  is equivalent to adding a gap to the vertical  $x$  sequence that aligns with the  $j^{\text{th}}$  letter in the horizontal  $y$  sequence.
  3. A **diagonal move** to cell  $(i + 1, j + 1)$  is equivalent to aligning the  $i^{\text{th}}$  letter in the vertical  $x$  sequence with the  $j^{\text{th}}$  letter in the horizontal  $y$  sequence.
- There are  $\gg 2^n$  possible paths from the top left corner to the bottom right corner of the matrix! Which is the optimal? A **brute force approach** would be to enumerate all such paths, compute the score for each, and output the path with the best score.

- **Dynamic programming** is a commonly used technique for solving problems that can be broken down into sub-problems, which are in turn solved by further division of the problem until we reach some simple case whose solution can be found in constant time. An important observation is that the **alignment score is additive**. That is, suppose we know that the first  $i$  letters of  $x$  aligns with the first  $j$  letters of  $y$  and that the  $i+1$  to  $n$  letters of  $x$  aligns with the  $j+1$  to  $m$  letters of  $y$ . Then, the optimal alignment score of  $x$  and  $y$  is the sum of the optimal alignment scores of the two subsequences.

- This technique can be applied when:
  1. There are a polynomial number of sub-problems.
  2. The original problem is one of the sub-problems.
  3. Each sub-problem can be easily solved from the solutions of smaller sub-problems.
- In the sequence alignment problem, there are 3 possibilities when aligning  $x_i$  to  $y_j$ :
  1.  $x_i$  aligns to  $y_j$ , in which case the alignment score  $F(i, j)$  of the first  $i$  letters of  $x$  with the first  $j$  letters of  $y$  can be defined as:

$$F(i, j) = F(i-1, j-1) + \begin{cases} m & \text{if } x_i = y_j \\ -s & \text{o.w.} \end{cases}$$

2.  $x_i$  aligns to a gap:

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

3.  $y_j$  aligns to a gap:

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

By assuming that  $F(i-1, j-1)$ ,  $F(i-1, j)$ , and  $F(i, j-1)$  are the best alignment scores for their respective subsequences, we can determine the optimal alignment score for  $F(i, j)$  by taking the max over these 3 possible cases:

$$F^*(i, j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases} \quad \text{where } s(x_i, y_j) = m \text{ if } x_i = y_j, -s \text{ o.w.}$$

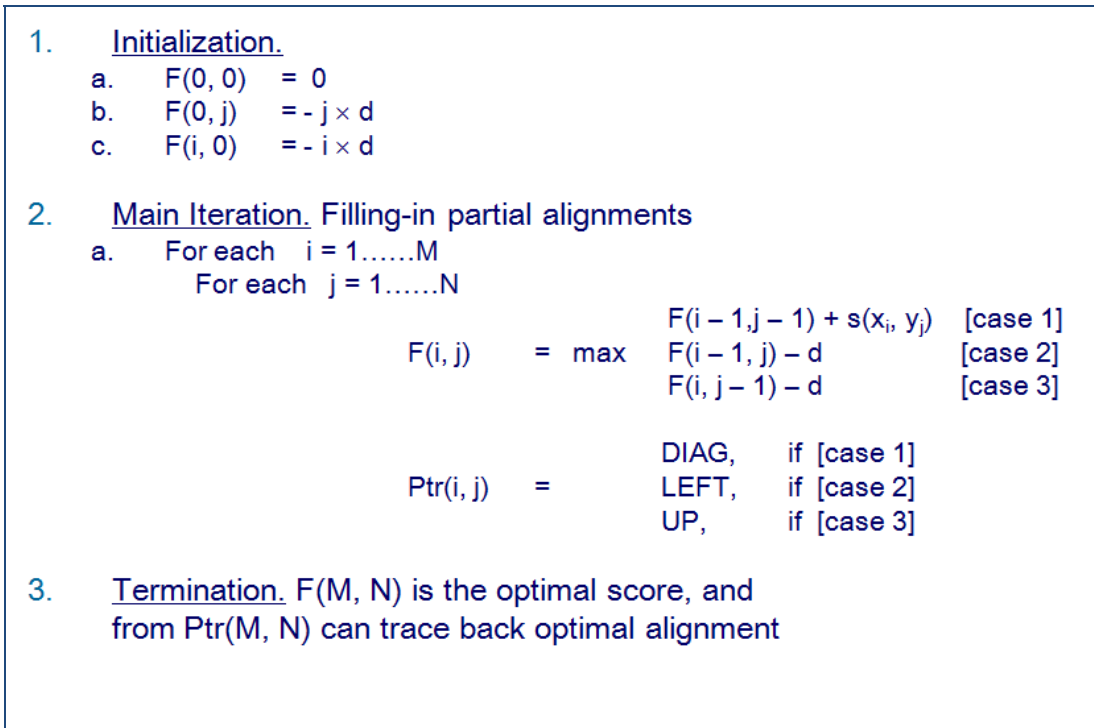


Figure 8: pseudo code for finding the optimal sequence alignment using dynamic programming

The pseudo code for this algorithm is presented in Figure 8 above. As we fill in the matrix, in addition to keeping track of the alignment scores (so as to compute future alignment scores), it is also necessary to keep a **back-pointer**. This back-pointer points to the cell from which the alignment score was used to find the current cell's score. After termination, we obtain the actual alignment – rather than just the score – by tracing from the back-pointer in the bottom-right corner cell backwards to the upper-left corner cell.

- Time & space
  - This algorithm fills in all cells in a  $n \times m$  matrix, where the filling of each cell takes constant time (by examining the 3 cells above, to the left, and diagonally to the upper-left of the current). Thus, the **time complexity** of this algorithm is  $O(nm)$ , where  $n$  and  $m$  are the lengths of the input sequences.  $O(nm)$  **space** is used to store the alignment score and back-pointer for each cell.