Edit Distance

Given an alignment of two strings

its cost is the sum of the number of gaps and number of mismatches

The edit distance between two strings $x_1 - x_m$, $y_1 - y_n$ is the min cost of an alignment between them.

Applications computational biology, unix diff, search, ...

Let $OPT(i,j)$ = min cost of aligning $x_1 - x_i$ and $y_1 - y_j$

Three courses of action:

(1) Align ~~Match~~ x_i and y_j.

Cost: possible mismatch between x_i & y_j plus cost of aligning $x_1 - x_{i-1}$ & $y_1 - y_{j-1}$

(2) Align $x_1 - x_{i-1}$ and $y_1 - y_j$, leave x_i unmatched

Cost: 1 (for gap) + cost of aligning $x_1 - x_{i-1}$, $y_1 - y_j$

(3) Similarly but when y_j is unmatched.

$OPT(i,0) = i$

$OPT(0,j) = j$

$\begin{cases} 0 & \text{if } x_i = y_j \\ 1 & \text{if } x_i \neq y_j \end{cases}$

$$OPT(i,j) = \min \begin{cases} \mathbb{1}_{x_i \stackrel{?}{=} y_j} + OPT(i-1, j-1) \\ 1 + OPT(i-1, j) \\ 1 + OPT(i, j-1) \end{cases}$$

Algorithm

```
For i ← 0..m
    OPT[i,0] ← i
For j ← 0..n
    OPT[0,j] ← j
For i ← 1..m
    For j ← 1..n
        b ← (x_i = y_j)?
        OPT[i,j] ← min { b + OPT[i-1, j-1],
                         1 + OPT[i-1, j],
                         1 + OPT[i, j-1] }.
```

Run-time $O(mn)$

Space $O(mn)$

Example

```
  1 2 3 4
C G A
C G G A
```

	0	1	2	3	4
0	0	1	2	3	4
1	1	0	1	2	3
2	2	1	0	1	2
3	3	2	1	1	1

```
C G  A
C G G A
```

OR

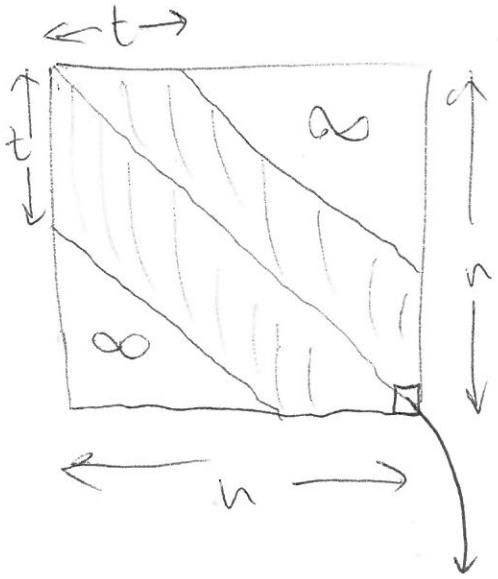
```
C G A
C G G A
```

Interestingly, under reasonable assumptions, there is no time $n^{2-\varepsilon}$ algorithm for computing the edit distance between length-n strings.

What if we suspect that the distance is at most t?

Turns out there's an $O(nt)$ time algorithm

The idea: only compute diagonal band:



This gives the cost of an alignment between the strings.

* If the dist $\leq t$, then cost = dist.