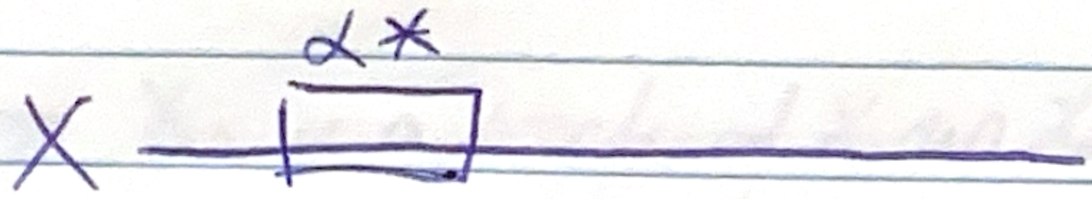
CS1810 9/18/25

Computational Biology

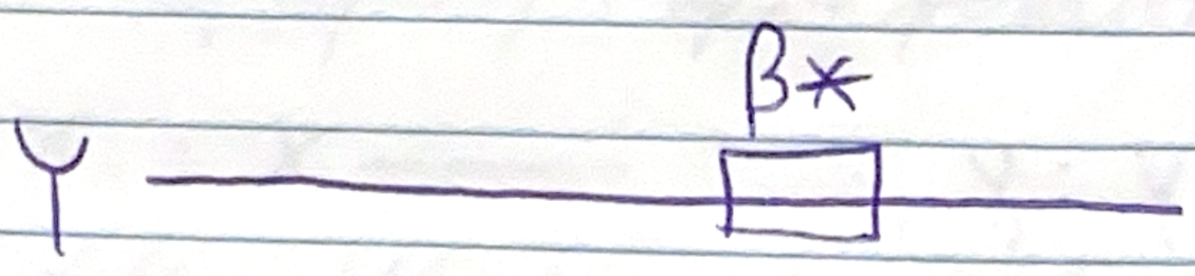
Local Alignment:

Given: 2 sequences X, Y over the same alphabet, and a scoring scheme δ.
Compute: the maximum score local alignment of X with Y.
↳ If 2 random sequences are aligned of length N, the length of the global alignment on average is $O(N)$, while the length of the local alignment on average is $O(\log N)$.

- The optimal local alignment is the alignment of 2 substrings in X and Y that produce the highest possible score. In other words, choose a substring of X and a substring of Y such that when these two substrings are aligned, the maximum score is produced.



Ex.



is the optimal local alignment.

Computational Biology

Suffixes, Prefixes, and Substrings:

Suppose $X = ACBAAA$.

↳ $Pref(X) = \{A, AC, ACB, ACBA, ACBAA, ACBAAA\}$.

↳ $Suff(X) = \{A, AA, AAA, BAAA, CBAAA, ACBAAA\}$.

↳ $Sub(X)$, there are $O(n^2)$ substrings for a string of length n.

Theorem: The prefixes of suffixes or the suffixes of prefixes for a string is equivalent to the set of substrings.

↳ $Pref(Suff(X)) = Suff(Pref(X)) = Sub(X)$.

That begs the question: For sequences X and Y of length M and N respectively, there are $O(N^2M^2)$ possible local alignments possible. How does local alignment do it in $O(NM)$ time?

Suppose X_i is a prefix of X up to position i and Y_j is a prefix of Y up to position j.

$X_i : X$ ——— i ……, $Y_j : Y$ ——— j ……

Now, define $V(i,j)$ as the value of the optimal global alignment between all pairs of suffixes of X_i and Y_j.

Computational Biology

Local Alignment algorithm:

↳ Initialize $V(i,0) = 0$
↳ Initialize $V(0,j) = 0$ for all i, j
FOR $i > 0$ and $j > 0$ DO

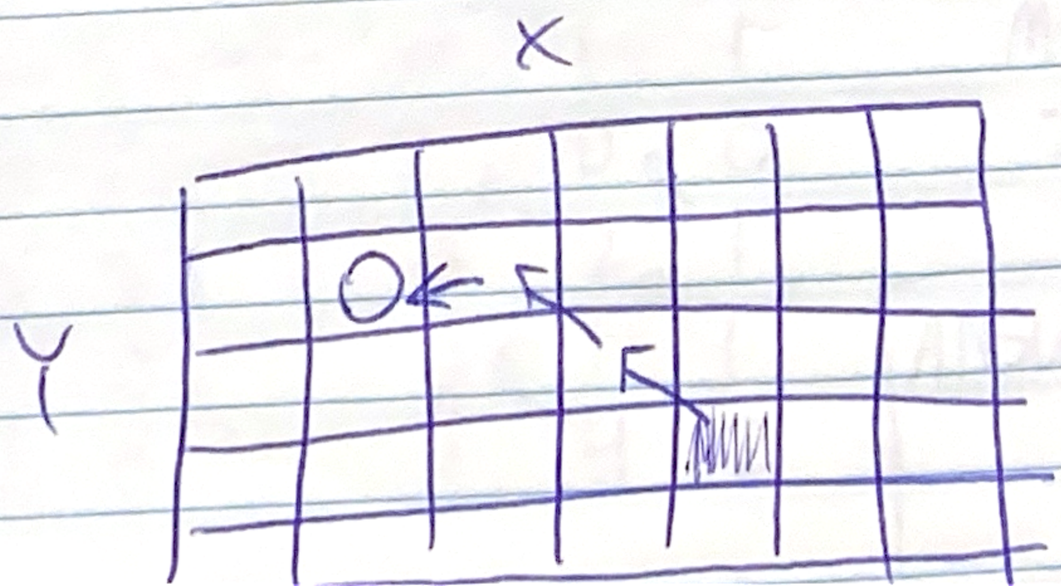
$$V(i,j) = MAX \begin{cases} 0 \\ V(i-1,j-1) + \delta(x_i, y_j) \\ V(i-1,j) + \delta(x_i, -) \\ V(i,j-1) + \delta(-, y_j) \end{cases}$$

Finally, return $V^{OLA}(X,Y) = MAX_{\substack{0<i\leq N \\ 0<j\leq M}} [V(i,j)]$
(OLA = optimal local alignment)

Note: In global alignment we chose the bottom right corner of the DP table as the alignment score. However, in local alignment, we choose the maximum value among ALL the cells of the DP table.

To find the local alignment that is optimal, backtrack from the max score back to a 0 value. AKA follow the backpointers until you encounter a 0.

Ex.



Computational Biology

To understand why this algorithm works, here is a useful theorem:

Thm: $V^{OLA} = MAX[V(i,j), 1 \le i \le N, 1 \le j \le M]$, where V^{OLA} is the alignment between substrings of X and Y resulting in the maximum score.

Proof- We know that $V^{OLA} \ge MAX[V(i,j), 1 \le i \le N, 1 \le j \le M]$ because $V(i,j)$ considers suffixes of prefixes, which are therefore also substrings.
We also know that $MAX[V(i,j), 1 \le i \le N, 1 \le j \le M] \ge V^{old}$ because the left hand side considers all possible suffixes of prefixes and therefore all possible substring pairs. This means that the MAX score over that will be at least the score resulting from one specific pair of substrings (aka V^{OLA}).

↳ $V^{OLA} = MAX[V(i,j), 1 \le i \le N, 1 \le j \le M]$.

Ex.

		A	C	T	C	C
	0	0	0	0	0	0
A	0	1	0	0	0	0
C	0	0	2	0	1	1
A	0	1	0	1	0	0
C	0	0	2	0	2	1
T	0	0	0	3	1	1
C	0	0	1	1	4	2

MATCH: +1
MISMATCH: -1
INDEL: -2

ANSWER:
A C T C
| | | |
A C T C