Exercise 5: Motifs and alignment

Let $P$ be a PWM of length $l$ with consensus sequence $c$.
Let $s$ be a DNA sequence of length $n > l$.

Ex: $p = \begin{bmatrix} a & b & c & d & e & f & g & h \end{bmatrix}$

$s = \text{acgcggccaatcagaggccac}$

$\text{PWM:}$

\begin{array}{cccccccc}
A & 1 & 0 & 6 & 0 & 0 & 1 & 0 \\
C & 2 & 7 & 0 & 6 & 0 & 1 & 0 \\
G & 4 & 0 & 1 & 1 & 7 & 0 & 7 \\
T & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\
\end{array}

$a) \text{ Create an algorithm that finds the best ungapped match of } c \text{ in } s.$

$b) \text{ Generalise your algorithm to find the best ungapped match of } P \text{ in } s.$
c) Create an algorithm that finds the best gapped match (alignment) of \( c \) in \( s \).
d) Generalise your algorithm to find the best ungapped match of \( P \) in \( s \).

\[ s = \text{acgcggccaatcagaggccac} \]

\[ GCACGTGG \]

Consider first alignment with consensus

\[ c = \text{GCACGTGG} \]

Input string:

\[ \text{acgcggccaatcagaggccac} \]

Scan:

\[ \text{GCACGTGG} \]

..acGCACGAGGtg..

\[ \text{GCACG---TGG} \]

..acGCACGAGGtg.. ..acGCACGAGGtg..

What kind of alignment is this? Local? Global?

Recap: global alignment

- Initialization
  - Insertion cost: -1
- Recurrence

\[
PA(S_i, R_j) = \max \left\{ PA(S_{i-1}, R_{j-1}) + \delta(s_i, r_j), \right.
\[
PA(S_{i-1}, R_j) - 1, \quad PA(S_i, R_{j-1}) - 1 \bigg\}
\]

Recap: local alignment

- Initialization
  - Insertion cost: -1
- Recurrence

\[
PA(S_i, R_j) = \max \left\{ PA(S_{i-1}, C_{j-1}) + \delta(s_i, c_j), \right.
\[
PA(S_{i-1}, C_j) - 1, \quad PA(S_i, C_{j-1}) - 1 \bigg\}
\]

This case: map the whole of \( c \) locally into \( s \)

- Initialization
  - Insertion cost: -1
- Recurrence

\[
PA(S_i, R_j) = \max \left\{ PA(S_{i-1}, C_{j-1}) + \delta(s_i, c_j), \right.
\[
PA(S_{i-1}, C_j) - 1, \quad PA(S_i, C_{j-1}) - 1 \bigg\}
\]

Variant: gaps allowed only in \( c \)

- Initialization
  - Insertion cost: -1
- Recurrence

\[
PA(S_i, R_j) = \max \left\{ PA(S_{i-1}, C_{j-1}) + \delta(s_i, c_j), \right.
\[
PA(S_{i-1}, C_j) - 1, \quad PA(S_i, C_{j-1}) - 1 \bigg\}
\]
Alignments of PWM to string

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

Gap penalty = 1

Input string:
acgcggccaatcagaggccac
CACG
A
G
gttgggcaagtccgcctctcgggagg

Scan:
GCACGTGG
..acGCACGAGGtg.. = 44
47667176
GCACG---TGG
..acGCACGAGGTGGgg.. = 45
147
47667111576

Comparing two PWMs

e) Create an algorithm that calculates the similarity (or “edit distance”) of two PWMs.

Similarity of two PWMs?

- Similarity of two PWM columns

C1    C2     Diff
A 1     0     1
C 2     6     4
G 4     1     3
T 0     0     0
Total: 8

\[ d(P_{1,c}, P_{2,c}) = \sum_{a=1}^{4} |P_{1,a} - P_{2,a}| \]

Aligning two PWMs

- Initialization?
- Recurrence?

Aligning two PWMs

- Initialization?
- Recurrence?
Aligning two PWMs

- Initialization?
  - Insertion cost: \( k \)

- Recurrence?

\[
\begin{align*}
\min & \quad PP(P_{1,i-1}, P_{2,j-1}) + d(P_{1,i}, P_{2,j}), \\
& PP(P_{1,i-1}, P_{2,j}) + k, \\
& PP(P_{1,i}, P_{2,j-1}) + k
\end{align*}
\]

\[
d(P_{1,i}, P_{2,j}) = \sum_{i=1}^{m} |p_{1,i} - p_{2,j}|
\]