Searching position specific scoring matrices

- Functional biological sequences typically come in families, such as Kinases, Globulins, Histons, Motorproteins, ...
- Many of the most powerful sequence analysis methods are based on identifying the relationship of an individual sequence to a sequence family.
- Sequences in a family usually diverged from each other in their primary sequence during evolution, while maintaining the same or a related function.
- As a consequence, identifying that a sequence belongs to the family, often allows inference about its function.
- Capturing all signals common to the members of a sequence family is best done by a statistical approach based on a multiple alignment (see later section of this course) of the sequences in the family.

Example 1

Consider an example of a family of 5 sequences `GTAT`, `GGCG`, `TTCG`, `GTCA`, and `GTTA` which can be represented by the following multiple alignment:

G	T	A	T
G	G	C	G
T	T	C	G
G	T	C	A
G	T	T	A

For each of the four positions in the sequences we want to capture the conservation of specific symbols by determining their probability of occurrence. This leads to a PSSM:

A	0.0	0.0	0.2	0.4
C	0.0	0.0	0.6	0.0
G	0.8	0.2	0.0	0.4
T	0.2	0.8	0.2	0.2

Now each DNA-sequence of length 4 can be scored by summing up the scores of the bases at the corresponding positions, e.g. `GTGA` gets score $0.8 + 0.8 + 0.0 + 0.4 = 2.0$.

Searching position specific scoring matrices

- Position specific scoring matrices (PSSMs) have a long history in sequence analysis.
- A high PSSM-score in some region of a sequence often indicates a possible biological relationship of this sequence to the family or motif characterized by the PSSM.
- There are several databases incorporating PSSMs, e.g. PROSITE, PRINTS, BLOCKS, or TRANSFAC.
- In this section we describe what PSSMs are and we develop methods to search PSSMs.

Searching position specific scoring matrices

- A PSSM is a representation of a multiple alignment of related sequences.
- We define it as a function $M : \mathcal{A} \times \{1, \ldots, m\} \to \mathbb{R}$, where $\mathcal{A}$ is a finite alphabet and m is the length of M.
- Usually M is represented by an $|\mathcal{A}| \times m$-matrix, see Table 1 for an example.
- Each column of the matrix reflects the frequency of occurrence of an amino acid or nucleotide at the corresponding position of the alignment.
- From now on, let M be a PSSM of length m.
- We define $score(w, M) = \sum_{h=1}^{m} M(w[h], h)$ for a sequence $w \in \mathcal{A}^m$ of length m.
- Given a sequence S of length n over alphabet $\mathcal{A}$ and a threshold value th, the *PSSM searching problem* is to find all positions j, $1 \leq j \leq n - m + 1$ in S such that $score(S[j \ldots j + m - 1], M) \geq th$.

Example 2

Reconsider the PSSM of Example 1:

A	0.0	0.0	0.2	0.4
C	0.0	0.0	0.6	0.0
G	0.8	0.2	0.0	0.4
T	0.2	0.8	0.2	0.2

Now let $S = \texttt{AGATCCTAACG}$ and $th = 1.2$. Then we obtain two solutions to the PSSM-searching problems:

position	substring	score
3	ATCC	$0.0 + 0.8 + 0.6 + 0.0 = 1.4$
6	CTAA	$0.0 + 0.8 + 0.2 + 0.4 = 1.4$

Searching position specific scoring matrices

- A simple algorithm for the PSSM searching problem slides along the sequence and computes $score(w, M)$ for each $w = S[j \ldots j + m - 1]$, j, $1 \leq j \leq n - m + 1$.
- The running time of this algorithm is $O(mn)$.
- This algorithm is used e.g. in the programs *FingerPrintScan* [Scordis et al., 1999], *BLIMPS* [Henikoff et al., 2000], *MatInspector* [Quandt et al., 1995], and *MATCH* [Kel et al., 2003].
- The technique of lookahead scoring, introduced in [Wu et al., 2000], gives an improvement over the simple algorithm.
- Lookahead scoring allows to stop the calculation of $score(w, M)$ when it is clear that the given overall score threshold th cannot be achieved.
- A similar technique was used when computing the k-environment for Blast.

Example 3

Reconsider the PSSM of Example 1:

A	0.0	0.0	0.2	0.4
C	0.0	0.0	0.6	0.0
G	0.8	0.2	0.0	0.4
T	0.2	0.8	0.2	0.2

- Let $S =$ AGATCCTAACG, $th = 1.2$ and consider the substrings CCTA and AACG at positions 5 and 8, respectively.
- For the first two characters in these substrings we obtain the score 0.
- The maximum score we can achieve for the last two characters is $0.6 + 0.4$.
- So once we have read the first two characters CC of CCTA or AA of AACG, we know that we can achieve a score of at most 1 for the entire string.
- This is below the threshold.
- So we can stop after the first two characters of these substrings.

Searching position specific scoring matrices

To explain the lookahead scoring method, define

$$pfxscore_d(w, M) = \sum_{h=1}^{d} M(w[h], h),$$

$$\max_h = \max\{M(a, h) \mid a \in \mathcal{A}\},$$

$$\sigma_d = \sum_{h=d+1}^{m} \max_h$$

for any d, $1 \leq d \leq m$.

- That is, $pfxscore_d(w, M)$ is the score for the prefix $w[1 \ldots d]$ of length d.
- Moreover, σ_d is the maximal score that can be achieved in the last $m - d$ positions of the PSSM.
- Let $th_d = th - \sigma_d$ be the *intermediate threshold* at position d, for $1 \leq d \leq m$.

The next lemma reveals an important property of prefix scores:

Lemma 1

The following statements are equivalent:

1. $pfxscore_d(w, M) \geq th_d$ *for all* d, $1 \leq d \leq m$.
2. $score(w, M) \geq th$.

Proof

1⇒2: Suppose that 1 holds. Consider the special case for $d = m$. Then $\sigma_m = \sum_{h=m+1}^{m} \max_h = 0$ and

$$\begin{aligned} score(w, M) &= \sum_{h=1}^{m} M(w[h], h) \\ &= pfxscore_m(w, M) \\ &\geq th_m \\ &= th - \sigma_m \\ &= th. \end{aligned}$$

Proof

2⇒1: Suppose that 2 holds. Let d, $1 \leq d \leq m$ be fixed but arbitrary. Then

$$\begin{aligned} score(w, M) &= \sum_{h=1}^{m} M(w[h], h) \\ &= \sum_{h=1}^{d} M(w[h], h) + \sum_{h=d+1}^{m} M(w[h], h) \\ &= pfxscore_d(w, M) + \sum_{h=d+1}^{m} M(w[h], h) \end{aligned}$$

Hence $score(w, M) \geq th$ implies
$pfxscore_d(w, M) + \sum_{h=d+1}^{m} M(w[h], h) \geq th$.

Proof

Since $M(w[h], h) \leq \max_h$ for all h, $1 \leq h \leq m$, we conclude

$$\sum_{h=d+1}^{m} M(w[h], h) \leq \sum_{h=d+1}^{m} \max_h = \sigma_d$$

and hence

$$\begin{aligned} pfxscore_d(w, M) &\geq th - \sum_{h=d+1}^{m} M(w[h], h) \\ &\geq th - \sigma_d \\ &= th_d. \end{aligned}$$

Searching position specific scoring matrices

- Lemma 1 gives a necessary condition for a PSSM-match, which can easily be exploited: When computing $score(w, M)$ by scanning w from left to right, one checks for $d = 1, 2, \ldots$, if the intermediate threshold th_d is achieved.
- If not, the computation can be stopped.
- See Algorithm 1 for pseudo-code and Table 1 for an example applying the algorithm.
- The lookahead scoring algorithm runs in $O(kn)$ time, where k is the average number of PSSM-positions per sequence start position actually evaluated.
- In the worst case, k is in $O(m)$, which leads to the worst case running time of $O(mn)$, not better than the simple algorithm.
- However, k is expected to be much smaller than m, leading to considerable speedups in practice.

Algorithm 1 (Lookahead scoring)

Input: Sequence S of length n, PSSM M of length m, threshold th
Output: all positions in S matching M.

1: compute th_d for $1 \leq d \leq m$
2: **for all** $j \leftarrow 1$ **upto** $n - m + 1$ **do**
3: $\quad score \leftarrow 0$
4: $\quad$ **for all** $d \leftarrow 1$ **upto** m **do**
5: $\quad\quad score \leftarrow score + M(S[j + d - 1], d)$
6: $\quad\quad$ **if** $score < th_d$ **then**
7: $\quad\quad\quad$ **break**
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: $\quad$ **if** $score \geq th$ **then**
11: $\quad\quad$ **print**(match at position j with score $score$)
12: $\quad$ **end if**
13: **end for**

Table 1: PSSM of length $m = 10$ of a zinc-finger motif. Let $th = 400$ be the score threshold. Then only substrings beginning with C or V can match the PSSM, since all other amino acids score below the intermediate threshold $th_1 = -7$. That is, lookahead scoring will skip over all substrings beginning with amino acids different from C and V, as $M(\mathtt{V}, 1) = 16 \geq th_1 = -7$ and $M(\mathtt{C}, 1) = 92 \geq th_1 = -7$ and $M(x, 1) < th_1 = -7$ for all other aminoacids x.

A	-19	5	7	-29	-14	-25	7	-34	7	-7
C	**92**	-17	-8	**99**	-22	-34	-8	-27	**40**	**43**
D	-45	17	-29	-55	14	-25	-25	-44	-16	16
E	-49	**22**	-28	-61	22	-16	-24	-43	-14	-7
F	-30	-28	2	-42	-28	-37	-19	-60	-9	-27
G	-36	-15	-25	-45	9	-30	-23	-41	-14	-15
H	-38	-7	-10	-47	-8	-15	-22	-8	-6	-9
I	-12	-23	25	-31	-26	-36	4	-16	-17	-24
K	-41	-8	-23	-52	15	45	-15	-38	14	-5
L	-21	-27	-4	-34	-27	-34	-10	-14	-20	-26
M	-22	-21	-5	-36	-20	-26	-8	-17	-15	-18
N	-40	-26	-25	-49	-7	-18	-19	-39	-10	-6
P	-46	18	-32	-56	-26	-35	-29	-51	-24	-25
Q	-44	-7	-26	-55	-3	-9	-21	-40	-11	25
R	-44	-13	-25	-55	**31**	**49**	11	-36	12	13
S	-30	-9	-18	-38	-13	-25	-13	-39	15	25
T	-25	9	13	-35	5	-26	**31**	-35	9	-8
V	16	-19	22	-29	-23	-33	31	-21	-13	-21
W	-35	-33	-11	-44	-30	-39	-31	-1	-16	-30
Y	-34	-25	**36**	-46	-24	-31	-22	**56**	20	-24
σ_d	407	385	349	250	219	170	139	83	43	0
th_d	-7	15	51	150	181	230	261	317	357	400


Henikoff, J., Greene, E., Pietrokovski, S., and Henikoff, S. (2000).
Increased Coverage of Protein Families with the Blocks Database Servers.
Nucleic Acids Res., 28:228–230.


Kel, A., Gößling, E., Reuter, I., Cheremushkin, E., Kel-Margoulis, O., and Wingender, E. (2003).
MATCH: a tool for searching transcription factor bindi ng sites in DNA sequences.
Nucleic Acids Res., 31:3576–3579.


Quandt, K., Frech, K., Wingender, E., and Werner, T. (1995).
MatInd and MatInspector: new fast and versatile tools for detection of consensus matches in nucleotide data.
Nucleic Acids Res., 23(23):4878–4884.


Scordis, P., Flower, D., and Attwood, T. (1999).
FingerPRINTScan: intelligent searching of the PRINTS motif database.
Bioinformatics, 15(10):799–806.


Wu, T., Nevill-Manning, C., and Brutlag, D. (2000).
Fast Probabilistic Analysis of Sequence Function using Scoring Matrices.
Bioinformatics, 16(3):233–244.