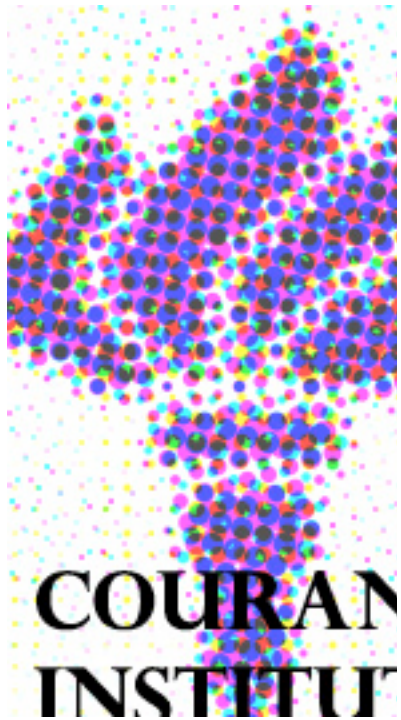
Bioinformatics

Richard Bonneau

Lecture 2: sequence alignment I.



Associated reading.

- Durbin, Eddy, et al. Ch. 2
- Papers for discussion section:
  - Gapped Blast +PSI-BLAST Paper: Nucleic Acids Res. 1997 Sep 1;25(17):3389-402
  - Proc Natl Acad Sci U S A. 1993 Jun 15;90(12):5873-7.

Pairwise Alignments I

- What do you need: i. definition of alignment ii. score to rank alignments iii. algorithm to search for alignments iv. score significance of match.
- This lecture

1. additive scoring scheme, substitution matrices (ii above)
2. alignment algorithms
3. judging match, calculating significance
4. heuristic tools, BLAST, gapped-BLAST, FASTA, PSI-BLAST
5. Useful links

- Bayes Rule:

$$P(y \mid x) = \frac{P(x \mid y)P(y)}{P(x)}$$

- Small amount of probability:

$$P(x \mid \vec{Y}) = P(x \mid y_1)P(y_1) + \ldots + P(x \mid y_n)P(y_n)$$

- Estimating P given 1 sample and a distribution, example with normal distribution, t-test. We'll use different distribution today.

(i.) alignment def

- sequence i : ABAACABA
- sequence j : ABAAABA

- alignment :

```
abaa-caba
||||..|||
abaaa-aba
```

(ii.) ranking alignments

```
alignment : abaa-caba
            ||||..|||
            abaaa-aba
```

One simple score:
alignment to gap = mismatch

$$S(x_i, y_j) = \begin{Bmatrix} 2 \text{ if match} \\ -1 \text{ mismatch} \end{Bmatrix}$$

Sum over **s**

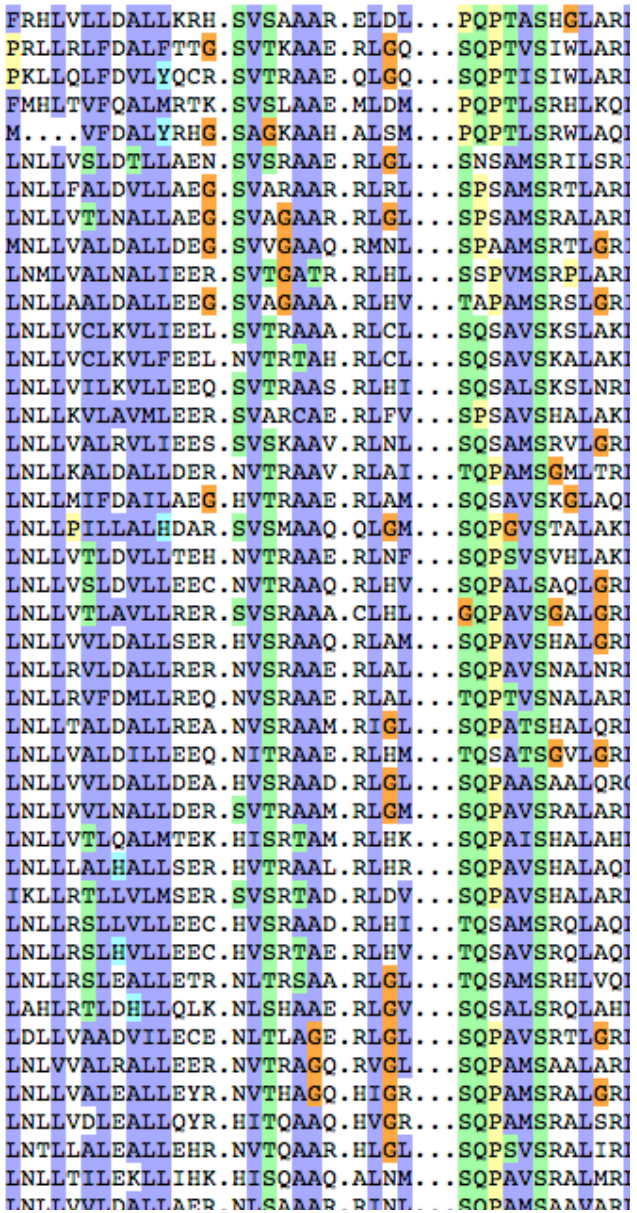
- Better score for matches approximates p(x,y|Match):

$$P(x,y|\text{Random}) = \prod_i q_{x_i} \prod_j q_{y_j}$$

$$P(x,y|\text{Match}) = \prod_i p_{x_i y_i}$$

$$\frac{P(x,y|M)}{P(x,y|R)} = \frac{\prod_i p_{x_i y_i}}{\prod_i q_{x_i} \prod_j q_{y_j}}$$

$$\log\left(\frac{P(x,y|M)}{P(x,y|R)}\right) = S(x,y)$$

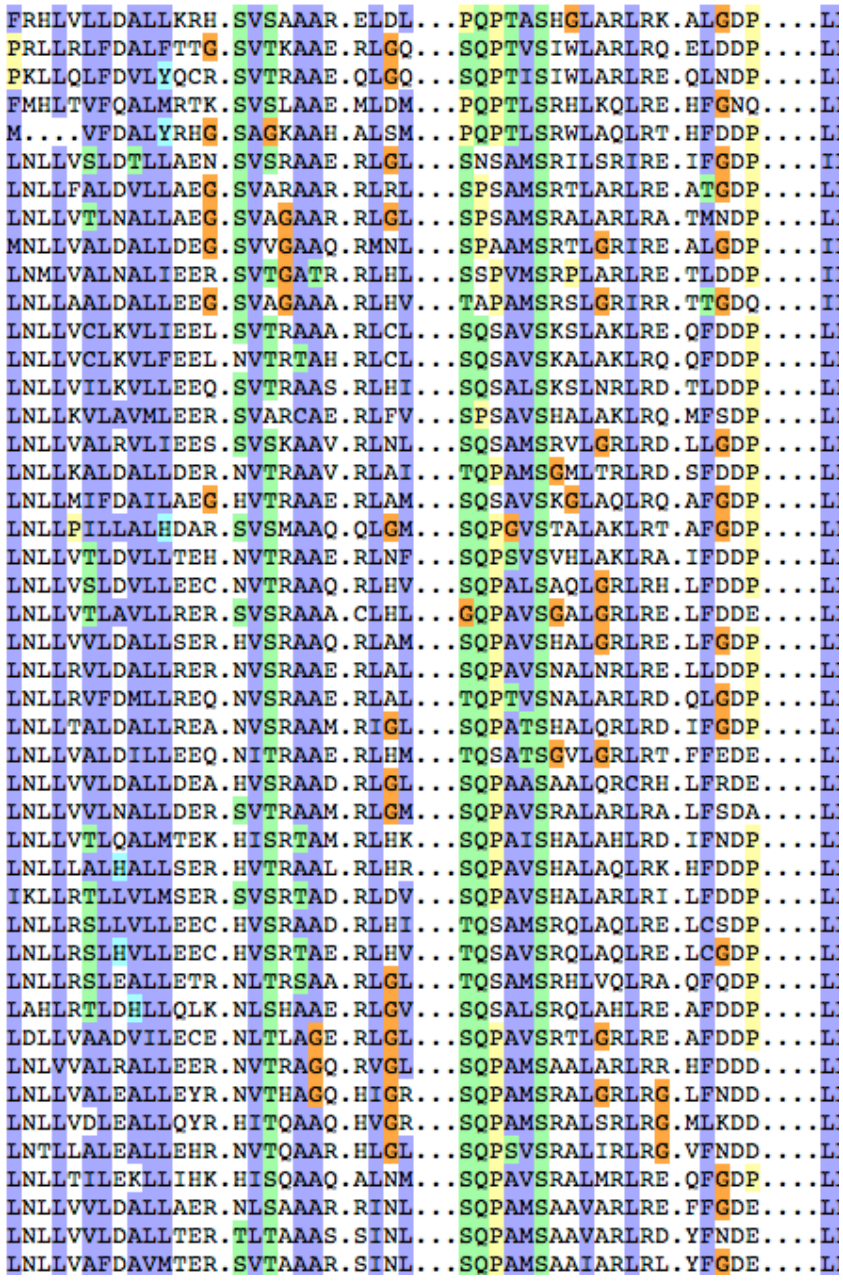


Better score for matches approximates p(x,y|Match):

Where do we get the "|M" in P(x,y|M)!!

BLOSUM 50,62:
From high quality alignments
with no redundancy
> 50, 62% identity

PAM, others



- Better score for matches approximates p(x,y|Match):
- Scoring gaps, gap penalties:
- g - length of gap; d - gap open
  e - gap extend

$$\gamma(g) = -eg == linear$$

$$\gamma(g) = -e(g-1) - d == afine$$

(iii.) Algorithm for finding optimal matches

Cost of trying every possible alignment exhaustively

$$if : n_i = n_j = n$$

$$\binom{2n}{n} = \frac{(2n)!}{(n!)^2} \simeq \frac{2^{2^n}}{\sqrt{\pi n}}$$

Brute force wont cut it…

(iii.) Algorithm for finding optimal matches

Dynamic Programming:

-global = Needleman-Wunsch 1970
Gotoh 1982

-local = Smith Waterman

(iii.) Algorithm for finding optimal matches (global)

We base larger global alignments on smaller preceding alignments.

Best alignment for x - up to x_i => y - up to y_j
based on alignment up to x_{i-1}, y_{j-1}

(iii.) Algorithm for finding optimal matches (global)

1. Set F(0,0) = 0

	0	c	a	d	b	d	
0	0						
a							
c							
b							
c							
d							
b							

(iii.) Algorithm for finding optimal matches (global)

F(i,j) :
3 possible outcomes:

x_i -> y_i , $F(i,j) = F(i-1, j-1) + s(x_i, y_j)$

x_i -> gap, $F(i,j) = F(i-1, j) - d$

gap -> y_i , $F(i,j) = F(i, j-1) - d$

F(i,j) = max of three above

Simple score for this example:
Mismatch = -1
Match = 2

(iii.) Algorithm for finding optimal matches (global)

1. Set F(0,0) = 0
2. F(i,0) = -id
3. F(0,j) = -jd

	0	c	a	d	b	d	
0	0	-1	-2	-3	-4	-5	
a	-1						
c	-2						
b	-3						
c	-4						
d	-5						
b	-6						

(iii.) Algorithm for finding optimal matches (global)

1. Set F(0,0) = 0
2. F(i,0) = -id
3. F(0,j) = -jd
4. Fill in F(i,j) from top-left to bottom-right (remember pointers)

	0	c	a	d	b	d	
0	0	-1	-2	-3	-4	-5	
a	-1	-1	1				
c	-2	1					
b	-3						
c	-4						
d	-5						
b	-6						

(iii.) Algorithm for finding optimal matches (global)

1. Set F(0,0) = 0
2. F(i,0) = -id
3. F(0,j) = -jd
4. Fill in F(i,j) from top-left to bottom-right (remember pointers)
5. Traverse to read algnmnt

```
cadb-d-
.|.|.|.
-acbcdb
```

Score = 2

	0	c	a	d	b	d	
0	0	-1	-2	-3	-4	-5	
a	-1	-1	1	0	-1	-2	
c	-2	1	0	0	-1	-2	
b	-3	0	0	-1	2	1	
c	-4	-1	-1	-1	1	1	
d	-5	-2	-2	1	0	3	
b	-6	-3	-2	0	3	2	

(iii.) Algorithm for finding optimal matches (global)

F(i,j) :
3 possible outcomes:

x_i -> y_i , $F(i,j) = F(i-1, j-1) + s(x_i, y_j)$

x_i -> gap, $F(i,j) = F(i-1, j) - d$

gap -> y_i , $F(i,j) = F(i, j-1) - d$

No outcome > 0, F(i,j) = 0

F(i,j) = max of three above OR 0
0 effectively starts new alignment

Both these algorithms work in O(nm) time
and O(nm) space

Should i explain big O?

(iii.) Algorithm for finding optimal matches (local - smith-waterman)

F(i,j) :
3 possible outcomes:

x_i -> y_i , $F(i,j) = F(i-1, j-1) + s(x_i, y_j)$

x_i -> gap, $F(i,j) = F(i-1, j) - d$

gap -> y_i , $F(i,j) = F(i, j-1) - d$

No outcome > 0, F(i,j) = 0

F(i,j) = max of three above OR 0
0 effectively starts new alignment

(iv.) calculating/estimating alignment significance (bayes)

We have: $P(x, y \mid M)$

We want: $P(M \mid x, y)$

I take it back, i'm skipping some steps here …
we don't want to be here all day

Bayes rule: $P(M \mid x, y) = \dfrac{P(x, y \mid M)P(M)}{P(x, y)}$

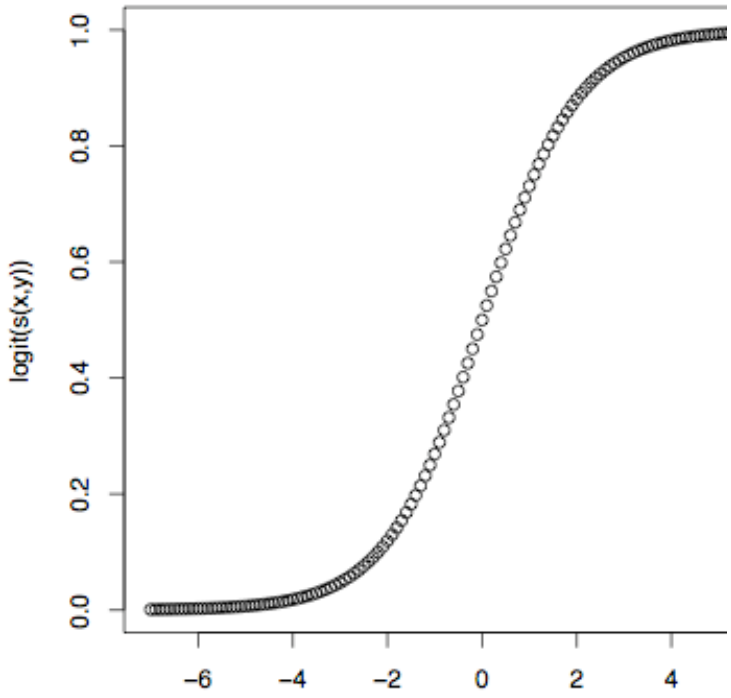
$$P(x, y) = P(x, y \mid R)P(R) + P(x, y \mid M)P(M)$$

$$P(M) = prior$$

$$P(R) = 1 - P(M)$$

$$S' = \log\left(\frac{P(M)}{P(R)}\right) + \log\left(\frac{P(x, y \mid M)}{P(x, y \mid R)}\right)$$

$$P(M \mid x, y) = \sigma(S') = \frac{e^{S'}}{1 + e^{S'}}$$



(iv.) calculating/estimating alignment significance (extreme value distribution)

$$P(M_N \leq x) \simeq \exp\left(-KNe^{\lambda(x-\mu)}\right)$$

Extreme value distribution
Free parameters:
Ungapped: S Karlin + Altschul analytical
Gapped : - Believed to follow analytical solution from Karlin (Mott 1992)
- Altschul + Gish 1996 - fit from large generated dataset.

Next week's reading

- Durbin, Eddy, et al. Ch. 3
- Papers for discussion section: