Global alignment

Weighted edit distance and global (end-to-end) alignment

Weighted edit distance

In nature, not all edit operations are of the same frequency and impact:
Gaps (insertions, deletions) are more severe than simple mismatch. Mismatch can be result of sequencing error (gap cannot). Also, not all mismatches are the same (ti/tv ratio).

Operation weighted edit-distance

- Associating cost with every edit-operation

This is achieved by adjusting penalties, and giving different scores to every **edit operation (operation weighted edit distance).**

We minimize on total operation weight.

Operation weighted edit distance

Important note:
Since substitution can be modeled as deletion followed by insertion, is substitutions are allowed, keep in mind that substitution should be less penalized than deletion plus insertion.

CTCAGG vs CTCA_GG
CTCTGG CTC_ TGG

Operation weighted edit distance

Usually we use same scores for insertion/deletion (d), and specific for match (e) and mismatch (r) operation.

Base conditions:

$$D(i,0) = i \times d$$

$$D(0,j) = j \times d$$

and recurrence relation:

$$D(i,j) = \min [D(i-1,j) + d,\ D(i, j-1) + d,\ D(i-1,j-1) + \delta(i,j)], \text{ where}$$

$\delta(i,j) = r$ if $i \neq j$, else if $i = j$, $\delta(i,j) = e$.

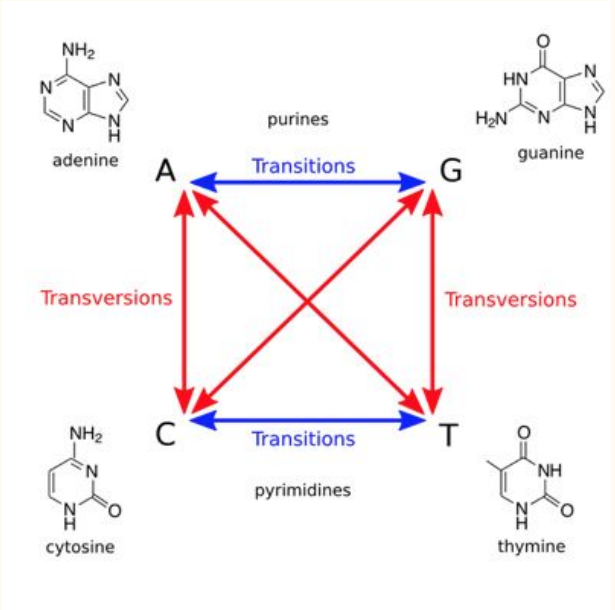
Operation weighted edit distance

Algorithmic complexity stays the same, and it’s O(nm).

Alphabet weighted edit distance

...also, not all mismatches are the same (ti/tv ratio). A to T is more costly than A to C. For indels, we want to weight by which character is being inserted/deleted.

Human *transition to transversion ratio* (AKA *ti/tv*) is ~2.1



Alphabet weighted edit distance

Solution: We score exactly on we add *alphabet-weighted* edit distance.

Note: weight depends on characters involved but not on where the characters appear in string.

Weighted edit distance

Alphabet weighted:

- Used in protein comparison
- PAM and BLOSUM scoring matrices - usually defined in maximizing (similarity) terms

- DNA strings comparison - mostly unweight or operation weight
  - Example: BLAST - match +5, mismatch -4

String similarity (global alignment)

More oftenly used way of representing string relatedness is to measure string similarity (instead of distance). - this will be much clearer in the next lesson (local alignment).

We need to score alignment in such way that every pair (position) in the alignment is given the score, and then the score of alignment is represented as the sum of scores for every position in the alignment.

String similarity (global alignment)

(Global) alignment of two strings S1 and S2 is obtained by first inserting chosen spaces (or dashes) either into (or at the end of) S1 or S2 and then placing one resulting string above the other so that so that every character (or space) in one string is opposite a unique character (or space) in another string.

- An alternative way (to edit transcript) to represent transformation of one string to another

v_intner_
wri_t_ers

Remember from the previous lecture comparison of alignment and edit transcript!

String similarity (more formally)

Def 1: (scoring letter pairs): Let $\sum$ be alphabet used for strings S1 and S2, and let $\sum$' be $\sum$ with added character "_" denoting a space. Then, for any two characters x, y in $\sum$', s(x,y) denotes the value (or *score*) obtained by aligning character x against character y.

Def 2 (alignment score): For given alignment A of S1 and S2, let S1' and S2' denote strings after the chosen insertion of spaces, and let l denote the (equal) length of two strings S1' and S2' in A. The value of alignment A is defined as $\sum_{i=1}^{l}$s(*S1'(i),S2'(i)*)

String similarity (global alignment)

Example: $\Sigma = \{a,b,c,d\}$, scoring function (matrix)

s	a	b	c	d	_
a	1	-1	-2	0	-1
b		3	-2	-1	0
c			0	-4	-2
d				3	-1
_					0

Alignment:

c a c _ d b d
c a b b d b _

Value of alignment score: $0+1\ -2\ +0\ +\ 3\ +3\ -1 = 4$

Global alignment

In practice, we set values in scoring matrix such that they are positive (or equal to zero) for match and negative for mismatch. Then we try to maximize alignment value (as stated on begging of this lecture).

$s(a, b)$:

	A	C	G	T	-
A	1	-3	-1	-3	-7
C	-3	1	-3	-1	-7
G	-1	-3	1	-3	-7
T	-3	-1	-3	1	-7
-	-7	-7	-7	-7	

gaps in b

gaps in a

-1 *Transitions* (A↔ G, C↔ T)

-3 *Transversions* (everything else)

-7 *Gaps*

Example: Scoring function reflecting that transitions are more common than transversions and mismatches are more common than gaps.

Global alignment

Def: Given a pairwise scoring matrix over the alphabet Σ', the similarity of two strings S1 and S2 is identified as the value of the alignment A of S1 and S2 that maximizes total alignment value. This is also called *optimal alignment* of S1 and S2.

Global alignment is also called Needleman-Wunsch alignment.

Global alignment

V(i,j): value of the optimal alignment for prefixes *S1*[1...i] and *S2*[1...j]

Base conditions:

$$V(0,j)=\sum_{1\le j\le k} s(\text{‘\_’}, S2(j))$$

$$V(i,0)=\sum_{1\le j\le k} s(S1(i),\text{’\_’})$$

And recurrence relation:

$$V(i,j)=\max[V(i-1, j)+s(S1(i), \text{‘\_’}), V(i, j-1)+s(\text{‘\_’}, S2(j)), V(i-1, j-1)+s(S1(i), S2(j))]$$

Global alignment

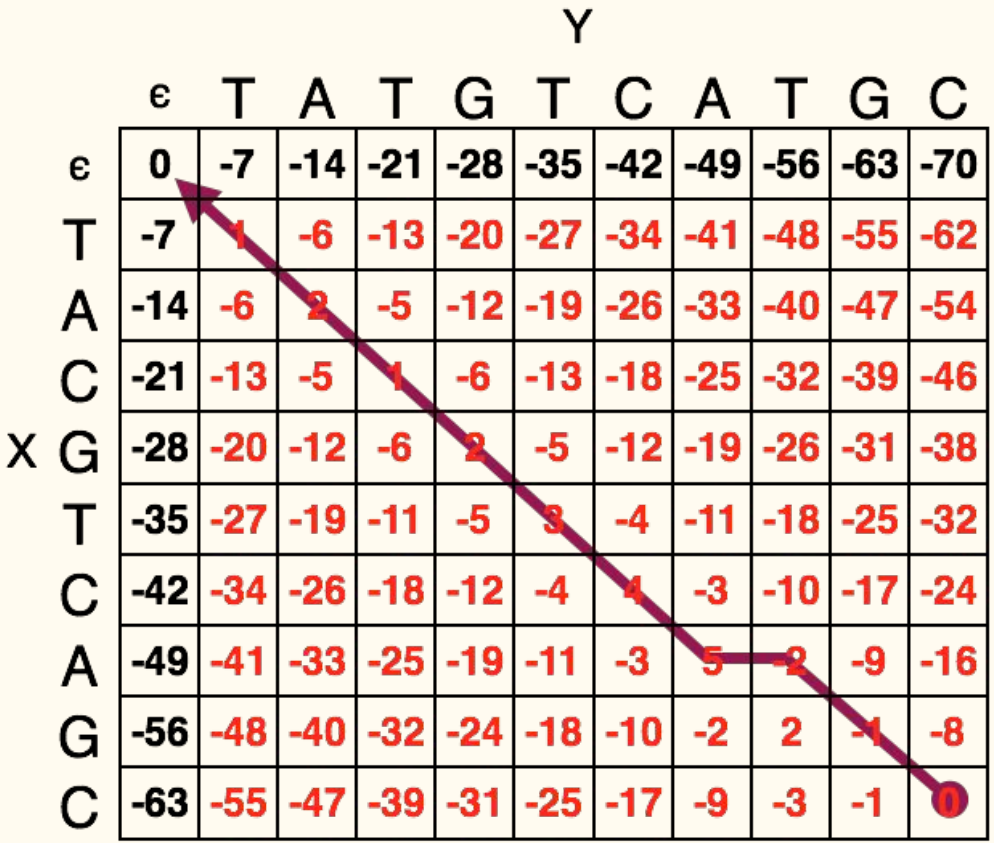
		Y									
		ϵ	T	A	T	G	T	C	A	T	G	C
	ϵ	0	-7	-14	-21	-28	-35	-42	-49	-56	-63	-70
	T	-7	1	-6	-13	-20	-27	-34	-41	-48	-55	-62
	A	-14	-6	2	-5	-12	-19	-26	-33	-40	-47	-54
	C	-21	-13	-5	1	-6	-13	-18	-25	-32	-39	-46
X	G	-28	-20	-12	-6	2	-5	-12	-19	-26	-31	-38
	T	-35	-27	-19	-11	-5	3	-4	-11	-18	-25	-32
	C	-42	-34	-26	-18	-12	-4	4	-3	-10	-17	-24
	A	-49	-41	-33	-25	-19	-11	-3	5	-2	-9	-16
	G	-56	-48	-40	-32	-24	-18	-10	-2	2	-1	-8
	C	-63	-55	-47	-39	-31	-25	-17	-9	-3	-1	0

$s(a, b)$

	A	C	G	T	-
A	1	-3	-1	-3	-7
C	-3	1	-3	-1	-7
G	-1	-3	1	-3	-7
T	-3	-1	-3	1	-7
-	-7	-7	-7	-7	

Optimal global alignment value

Global alignment




```
T A C G T C A - G C
| |   | | | |   | |
T A T G T C A T G C
```

-1 (transition) -7 (gap)

Global alignment vs weighted edit distance

Yes we can: just invert scoring matrix (larger score for similarity, smaller for difference) and minimize instead of maximize (in recurrence relation).

$s(a, b)$

	A	C	G	T	-
A	1	-3	-1	-3	-7
C	-3	1	-3	-1	-7
G	-1	-3	1	-3	-7
T	-3	-1	-3	1	-7
-	-7	-7	-7	-7	

...as long as we switch max to min:

	A	C	G	T	-
A	0	4	2	4	8
C	4	0	4	2	8
G	2	4	0	4	8
T	4	2	4	0	8
-	8	8	8	8	

Global alignment

	ϵ	T	A	T	G	T	C	A	T	G	C
ϵ	0	8	16	24	32	40	48	56	64	72	80
T	8	0	8	16	24	32	40	48	56	64	72
A	16	8	0	8	16	24	32	40	48	56	64
C	24	16	8	2	10	18	24	32	40	48	56
G	32	24	16	10	2	10	18	26	34	40	48
T	40	32	24	16	10	2	10	18	26	34	42
C	48	40	32	24	18	10	2	10	18	26	34
A	56	48	40	32	26	18	10	2	10	18	26
G	64	56	48	40	32	26	18	10	6	10	18
C	72	64	56	48	40	34	26	18	12	10	10

	A	C	G	T	-
A	0	4	2	4	8
C	4	0	4	2	8
G	2	4	0	4	8
T	4	2	4	0	8
-	8	8	8	8	

Optimal global alignment value

Global alignment vs weighted edit distance

X \ Y	ϵ	T	A	T	G	T	C	A	T	G	C
ϵ	0	8	16	24	32	40	48	56	64	72	80
T	8	0	8	16	24	32	40	48	56	64	72
A	16	8	0	8	16	24	32	40	48	56	64
C	24	16	8	2	10	18	24	32	40	48	56
G	32	24	16	10	2	10	18	26	34	40	48
T	40	32	24	16	10	2	10	18	26	34	42
C	48	40	32	24	18	10	2	10	18	26	34
A	56	48	40	32	26	18	10	2	10	18	26
G	64	56	48	40	32	26	18	10	6	10	18
C	72	64	56	48	40	34	26	18	12	10	10

$s(a, b)$

	A	C	G	T	-
A	0	4	2	4	8
C	4	0	4	2	8
G	2	4	0	4	8
T	4	2	4	0	8
-	8	8	8	8	

Same backtrace

Global alignment - complexity

Matrix-filling dynamic programming algorithm is O(mn) time and space

- Filling matrix is O(mn) space and time, and yields global alignment value
- Traceback is O(m + n) steps, yields optimal alignment / edit transcript

Global alignment - scoring schema

We can set scores how we like.

We can have mix of positive and negative scores. In the dynamic programming we can *maximize* a similarity score or *minimize* a dissimilarity penalty.