

CMSC 838T – Lecture 3

- ◆ **Pairwise sequence alignment**
 - Find similarity between two DNA / protein sequences
 - Hope similar sequence → similar function
- ◆ **This lecture**
 - Basic concepts, terminology
 - Alignment scoring metrics
 - Gap penalty
 - Scoring matrices
 - Overview of alignment algorithms
 - Dynamic programming
 - Dot matrix plot
 - FASTA
 - BLAST

CMSC 838T – Lecture 3

Motivation for Sequence Alignment

- ◆ **Large public sequence databases**
 - Genomic DNA – full DNA sequence in genome
 - mRNA – messenger RNA sequences (expressed genes)
 - cDNA – reverse transcription of mRNA (coding region)
 - Expressed sequence tags (ESTs) – short partial cDNA
 - Proteins – amino acids, structure, function
- ◆ **Alignment to similar sequence(s) in database can**
 - Transfer information (structure, function) between sequences
 - Suggest evolutionary relationships
 - Organize and classify genomic data

CMSC 838T – Lecture 3

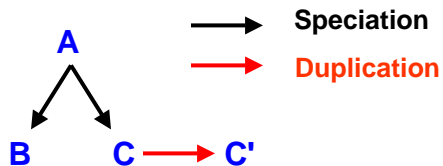
Similarity

- ◆ **Similarity**
 - Measure of closeness based on observable quantity
 - One example: % identity
 - 42 % identity = 42% of bases match exactly
- ◆ **Homology**
 - Conclusion that genes have common ancestry
 - All or nothing (42% homologous is meaningless)
 - High similarity may indicate homology
- ◆ **Conserved region**
 - Region(s) of highest similarity between homologous genes
 - May imply region perform useful / important function, thus conserved by evolution

CMSC 838T – Lecture 3

Two Types of Homology

- ◆ **Orthologs**
 - Genes with same function found in different species
 - Inherited from common ancestor
 - Differences due to speciation (evolution)
- ◆ **Paralogs**
 - Genes duplicated due to replication mutation
 - Genes assume different functions
- ◆ **Example**
 - A is ancestor to B, C
 - B, C are orthologs
 - C, C' are paralogs



CMSC 838T – Lecture 3

Alignment

◆ Alignment

- Mutual arrangement of sequences C A T C A G A T
- Gaps inserted if necessary : : : : : :
- Exhibits similarities and differences C - T C A G G T
- Score → measure of quality of alignment 1 -2 1 1 1 1 -1 1 = 3

◆ “Optimal” alignment

- Alignment with best score relative to metrics used
- May or may not have biological significance
 - ...because algorithm relies on approximations
 - Scoring matches / mismatches
 - Scoring gaps
 - many more...

CMSC 838T – Lecture 3

Global & Local Alignment

◆ Global alignment

- Best alignment of entire sequences to each other
- Q: Are two sequences generally the same?



◆ Local alignment

- Best alignment of parts of sequence
- Q: Do two sequences contain regions of high similarity?
- Biologically
 - Two sequences may differ in structure and function, but share common substructure / subfunction



◆ In general

- Use local alignment to **find** sequences with shared similarity
- Use global alignment to **compare** resulting sequences

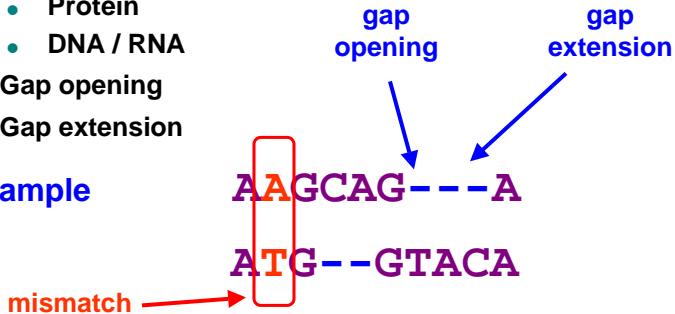
CMSC 838T – Lecture 3

Talk Outline

- ◆ Basic concepts, terminology
- ◆ Alignment scoring metrics ←
- Gap penalty
- Scoring matrices
- ◆ Overview of alignment algorithms
 - Dynamic programming
 - Dot matrix plot
 - FASTA
 - BLAST

CMSC 838T – Lecture 3

Scoring Similarity

- ◆ Scoring
 - Used to compare alignments
 - Can only score aligned sequences
- ◆ Three components to scoring
 1. Match or mismatch
 - Protein
 - DNA / RNA
 2. Gap opening
 3. Gap extension
- ◆ Example

CMSC 838T – Lecture 3

Scoring Similarity - Gaps

◆ Gaps

- Can be inserted in aligned sequences
- Can represent
 - Actual insertions / deletion (**indel**) mutations
 - Regions of low sequence similarity

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

◆ Biologically

- Probability of indels seems to drop slowly ($\log(n)$?) with size

◆ Scoring gaps

- Commonly use **affine** cost model (Cost = $h + g \times \text{gap length}$)
 - h = gap opening penalty (large)
 - g = gap extension penalty (small)
- Costs empirically determined (relative to scoring matrix)

CMSC 838T – Lecture 3

Scoring Matches and Mismatches

◆ Protein

- Amino acids have varying properties
- Use **scoring matrix** (amino acid substitution)

◆ Scoring

- Some matrices based on **observation**
 - PAM - pointwise mutations in similar proteins
 - BLOSUM - mutations in locally-conserved regions of distant proteins
 - Mutations observed in specific protein families...
- Other matrices based on **chemical / physical properties**
 - Chemical similarity (e.g., hydrophobicity of residues)
 - Codon distance (# base changes to convert amino acid)

CMSC 838T – Lecture 3

Scoring Protein Similarity – PAM

◆ Observations

- Amino acids vary greatly in mutability
- In PAM 250, replaced
 - 45% tryptophans and 48% cysteines (low variability)
 - 73% glycines and 94% asparagines (high variability)

◆ Limitations

- Some replacements observed too infrequently (36 never seen)
 - Use estimated replacement rate
- Assumes amino acid replacements are independent
- Assumes sequences have average amino acid composition

◆ Updated version of PAM (1992)

- Based on 59K replacements in 16K sequences

CMSC 838T – Lecture 3

Scoring Protein Similarity – BLOSUM

◆ BLOSUM (BLOcks SUBstitution Matrix) [Henikoff + 1992]

◆ Weight derivation

- Amino acid replacement rates found in (locally aligned) conserved regions of **distantly related** proteins
- Based on proteins in BLOCKS database ($> 10^6$ substitutions)

◆ Implementation

- BLOSUM unit = weighting for similarity between proteins
- BLOSUM 25 = if two protein sequences are more than 25% identical, their contributions to weighting are normalized to 1
 - Lower BLOSUM → more divergent
- Authors suggest gap penalty 8 for BLOSUM 62 (trial-and-error)

CMSC 838T – Lecture 3

Using Protein Scoring Matrices

◆ Some suggested scoring parameters

Query Length	Scoring Matrix	Gap Creation	Gap Extension
< 35	PAM-3	9	1
35-50	PAM-70	10	1
50-85	BLOSUM-80	10	1
> 85	BLOSUM-62	10	1

◆ Exhaustive comparisons

- BLOSUM 62 good at detecting weak similarity, better than PAM

◆ More advanced scoring (uses multiple sequences)

- Position specific scoring matrix (PSSM), HMM, motif, profile

CMSC 838T – Lecture 3

Scoring Matches and Mismatches

◆ DNA / RNA

- Bases can be classified as
 - Purines (A, G) single-ringed bases
 - Pyrimidines (C, T, U) double-ringed bases
- Point replacements
 - **Transition** (more likely)
 - ◆ purine ↔ purine OR pyrimidine ↔ pyrimidine
 - **Transversion** (less likely)
 - ◆ purine ↔ pyrimidine

◆ Scoring

- Uniform: bases identical (2), different (-6)
- Weighted: bases identical (2), transition (-5), transversion (-7)
 - Assumes transitions (3 X) more likely than transversions


CMSC 838T – Lecture 3

Pairwise Sequence Alignment

- ◆ **Search protein, not DNA sequences**
 - Less accidental similarity with 20 amino acids vs. 4 bases
 - More detailed comparisons possible for amino acids
 - PAM, BLOSUM, chemical / physical similarities
 - Protein databases are much smaller than DNA databases
- ◆ **If starting with DNA**
 - Convert DNA sequence to amino acid sequence, then search
 - Programs can translate all 6 reading frames
- ◆ **When to search DNA sequences**
 - No matches found in protein databases
 - Need to find matches in untranslated regions (UTRs)

CMSC 838T – Lecture 3

Talk Outline

- ◆ **Basic concepts, terminology**
- ◆ **Alignment scoring metrics**
 - Gap penalty
 - Scoring matrices
- ◆ **Overview of alignment algorithms**
 - Dynamic programming 
 - Dot matrix plot
 - FASTA
 - BLAST

CMSC 838T – Lecture 3

Difficulty of Pairwise Sequence Alignment

◆ Consider two sequences of length n

- Number of possible global alignments

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

- For $n = 40$, there are over 10^{23} possible alignments

◆ Approaches

- “Optimal” solution
 - Dynamic programming
- Heuristic solutions
 - Dot matrix plot
 - FASTA
 - BLAST

CMSC 838T – Lecture 3

Dynamic Programming

◆ General solution technique

- Typically applied to optimization problems
- Applicable if
 1. Problem can be recursively divided into subproblems
 2. Subproblems are not independent
(i.e., share subsubproblems)

◆ Approach

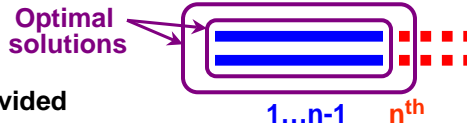
1. Recursively define value of optimal solution
2. Compute value of optimal solutions in bottom-up fashion
 - Store & reuse solutions to subproblems
3. Construct optimal solution from computed information

CMSC 838T – Lecture 3

Dynamic Programming - Alignment

◆ Alignment

- Problem can be subdivided
- Optimal alignment of n bases
 1. Incorporates optimal alignment of 1...n-1 bases
 2. Combine with best alignment for nth base



◆ Bioinformatic application

- Global alignment [Needleman-Wunsch 1970]
- Local alignment [Smith-Waterman 1981]

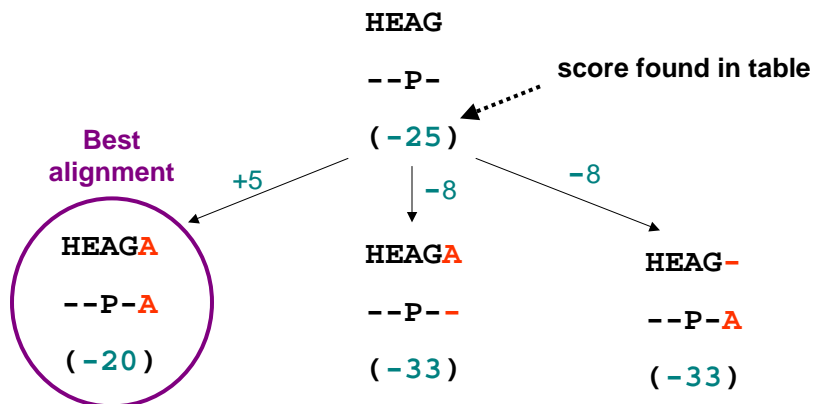
◆ Complexity

- $O(n^2)$ or $O(n \times m)$ algorithm (sequences of length n, m)
- Feasible for moderate sized sequences, not entire genomes

CMSC 838T – Lecture 3

Dynamic Programming - Alignment

Combination step either 1) matches term, or 2) inserts gap



- C1:** match top & bottom (match score = 5) **C2:** insert bottom gap (penalty = -8) **C3:** insert top gap (penalty = -8)

CMSC 838T – Lecture 3

Dynamic Programming - Global Alignment

◆ **Recursive formula**

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) & \mathbf{c1} \\ F(i-1, j) + d & \mathbf{c2} \\ F(i, j-1) + d & \mathbf{c3} \end{cases}$$

◆ **Notation**

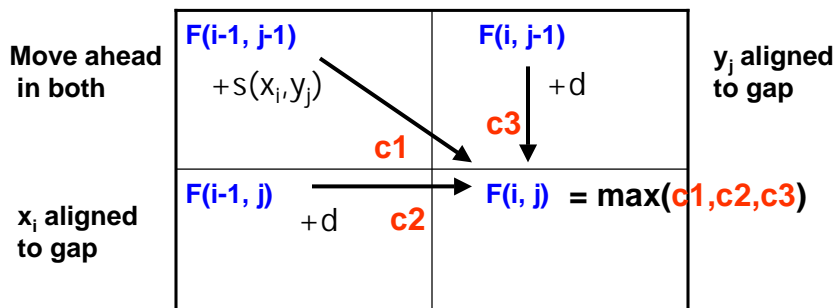
- x_i i^{th} letter of string x
- y_j j^{th} letter of string y
- $x_{1..i}$ prefix of x from letters 1 through i
- F matrix of optimal scores
-
- $F(i, j)$ represents optimal score lining up $x_{1..i}$ with $y_{1..j}$
- d gap penalty
- s scoring matrix

CMSC 838T – Lecture 3

Dynamic Programming - Global Alignment

◆ **Algorithm**

- Initialize: $F(0,0) = 0$, $F(i,0) = i \times d$, $F(0, j) = j \times d$ (*gap penalties*)
- Fill from top left to bottom right using recursive formula



While building the table, keep track of where optimal score came from, then reverse arrows

CMSC 838T – Lecture 3

Dynamic Programming – Scoring Matrix

Based on BLOSUM 50 Scoring Matrix

	H	E	A	G	A	W	G	H	E	E
P	-2	-1	-1	-2	-1	-4	-2	-2	-1	-1
A	-2	-1	5	0	5	-3	0	-2	-1	-1
W	-3	-3	-3	-3	-3	15	-3	-3	-3	-3
H	10	0	-2	-2	-2	-3	-2	10	0	0
E	0	6	-1	-3	-1	-3	-3	0	6	6
A	-2	-1	5	0	5	-3	0	-2	-1	-1
E	0	6	-1	-3	-1	-3	-3	0	6	6

CMSC 838T – Lecture 3

Dynamic Programming – Partial Table

Constant gap penalty $d = -8$

		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	-17	-25	-33	-42	-49	-57	-65	-73
A	-16										
W	-24										
H	-32										
E	-40										
A	-48										
E	-56										

CMSC 838T – Lecture 3

Dynamic Programming - Completed Table

Constant gap penalty $d = -8$

		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	-17	-25	-33	-42	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	-20	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	-5	-13	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	-3	-11	-19
E	-40	-22	-8	-16	-16	-9	-12	-15	-7	3	-5
A	-48	-30	-16	-3	-11	-11	-12	-12	-15	-5	2
E	-56	-38	-24	-11	-6	-12	-14	-15	-12	-9	1

CMSC 838T – Lecture 3

Dynamic Programming - Traceback

Trace arrows from bottom right to top left

		H	E	A	G	A	W	G	H	E	E
	0	-8	-16	-24	-32	-40	-48	-56	-64	-72	-80
P	-8	-2	-9	-17	-25	-33	-42	-49	-57	-65	-73
A	-16	-10	-3	-4	-12	-20	-28	-36	-44	-52	-60
W	-24	-18	-11	-6	-7	-15	-5	-13	-21	-29	-37
H	-32	-14	-18	-13	-8	-9	-13	-7	-3	-11	-19
E	-40	-22	-8	-16	-16	-9	-12	-15	-7	3	-5
A	-48	-30	-16	-3	-11	-11	-12	-12	-15	-5	2
E	-56	-38	-24	-11	-6	-12	-14	-15	-12	-9	1

Diagonal – both
Up – upper gap
Left – lower gap

Optimal
global
alignment

HEAGAWGHE-E
--P-AW-HEAE

CMSC 838T – Lecture 3

Dynamic Programming - Local Alignment

◆ Algorithm can also find local alignment

- Make **0** minimal score (i.e., start new alignment)
- Alignment can start / end anywhere
 - Start at highest score(s)
 - End when 0 reached
- Match with random sequence must have negative score
 - Avoid long random sequences with high scores
 - Mismatch → negative score

◆ Recursive formula

$$F(i, j) = \max \begin{cases} 0 \\ F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) + d \\ F(i, j-1) + d \end{cases}$$

Start new alignment

CMSC 838T – Lecture 3

Dynamic Programming - Local Alignment

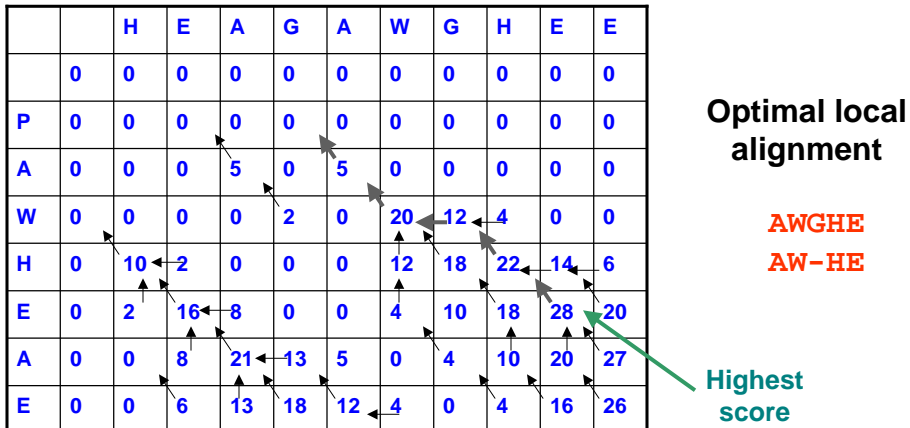
Constant gap penalty $d = -8$

		H	E	A	G	A	W	G	H	E	E
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	5	0	5	0	0	0	0	0
W	0	0	0	0	2	0	20	12	4	0	0
H	0	10	-2	0	0	0	12	18	22	-14	6
E	0	2	16	-8	0	0	4	10	18	28	20
A	0	0	8	21	-13	5	0	4	10	20	27
E	0	0	6	13	18	12	-4	0	4	16	26

CMSC 838T – Lecture 3

Dynamic Programming - Traceback

Start at highest score and trace arrows back to first 0



CMSC 838T – Lecture 3

Multiple, Repeat, & Overlap Matches

◆ Multiple alignments

- “Optimal” global alignment may be best by small margin
- Can report several high-scoring alignments

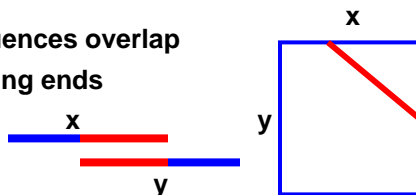
◆ Repeat matches

- Allow sections to match repeatedly
- Find repeated patterns (domain / motif)



◆ Overlap matches

- Matching when the two sequences overlap
- Does not penalize overhanging ends



CMSC 838T – Lecture 3

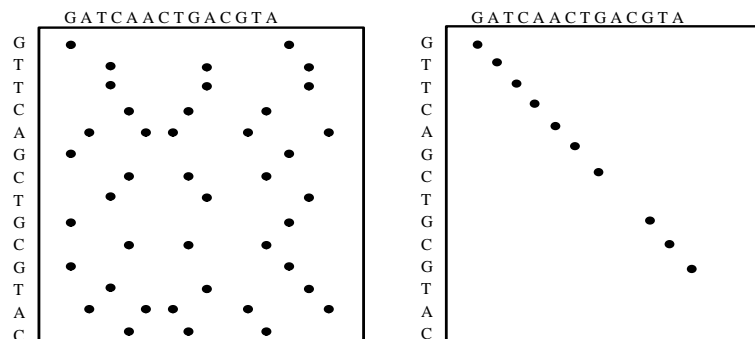
Talk Outline

- ◆ Basic concepts, terminology
- ◆ Alignment scoring metrics
 - Gap penalty
 - Scoring matrices
- ◆ Overview of alignment algorithms
 - Dynamic programming
 - Dot matrix plot ←
 - FASTA
 - BLAST

CMSC 838T – Lecture 3

Alignment Algorithms - Dot Matrix Plot

- ◆ **Method** [Gibbs & McIntyre 1970]
 1. Put two sequences on vertical and horizontal axes of graph
 2. Put dots wherever there is a match
 3. Diagonal line is region of identity (local alignment)
 4. Filter non-diagonals (minimal length, identity threshold)

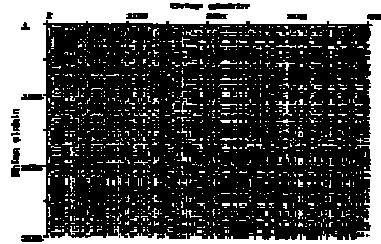


CMSC 838T – Lecture 3

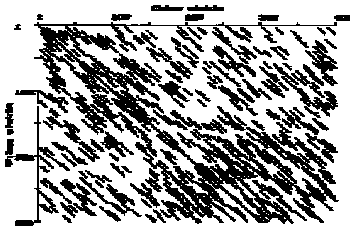
Alignment Algorithms - Dot Matrix Plot

◆ Example

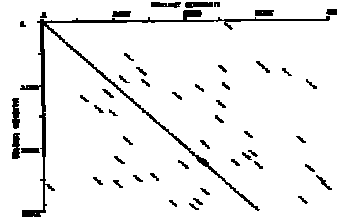
- Chimpanzee DNA (globin intergenic region) aligned with itself
- Filters needed to eliminate random matches



Filter: none



Filter: 6 of 10 bases match



Filter: 8 of 10 bases match

CMSC 838T – Lecture 3

Alignment Algorithms - Dot Matrix Plot

◆ Advantage

- Extremely simple
- Can use scoring matrix
- Can use multiple filters to rank matches
- Example
 - Red > 50 bases, 95% identical (very close)
 - Blue > 25 bases, 75% identical (close)
 - Green > 10 bases, 50% identical (distant)

◆ Limitations

- $O(n^2)$ algorithm
- Relies on visual analysis
- Difficult to find “best” alignment
- Despite multiple filters, difficult to compare alignments

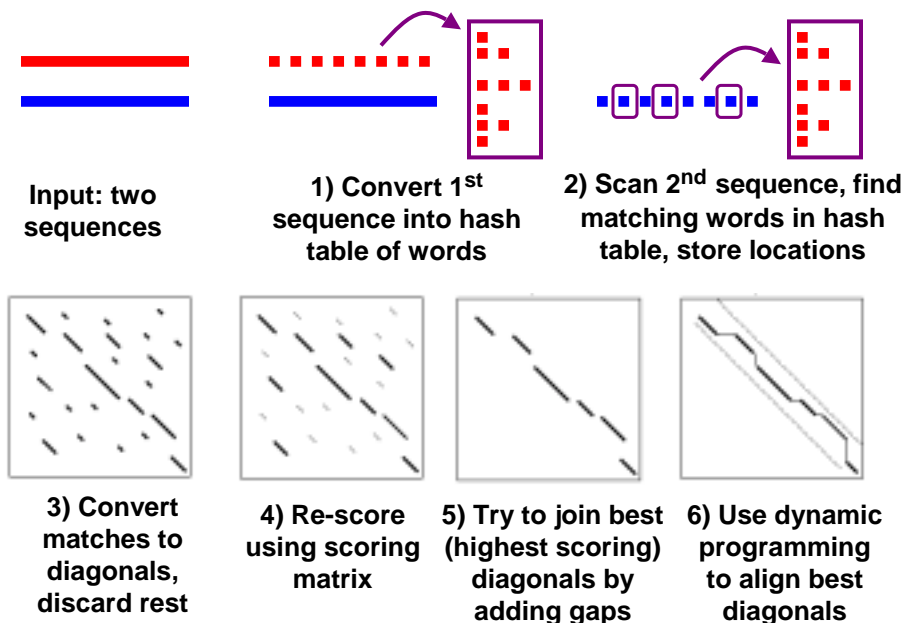
CMSC 838T – Lecture 3

Alignment Algorithms - FASTA

- ◆ **Approach** [Pearson & Lipman 1988]
 - Derived from logic of the dot matrix method
 - View sequences as sequences of short **words** (**k-tuple**)
 - DNA - 6 bases, protein - 1 or 2 amino acids
 - Start from nearby sequences of **exact** matching words
- ◆ **Motivation**
 - Good alignments should contain many exact matches
 - Hashing can find exact matches in $O(n)$ time
 - Diagonals can be formed from exact matches quickly
 - Sort matches by position ($i - j$)
 - Look only at matches near longest diagonals
 - Apply more precise alignment to small search space at end

CMSC 838T – Lecture 3

Alignment Algorithms - FASTA



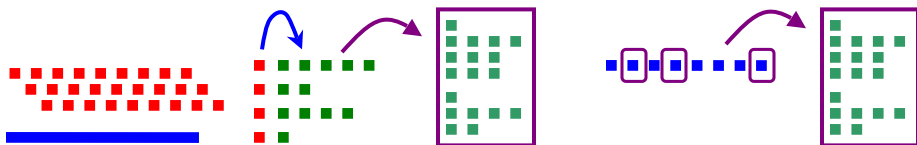
CMSC 838T – Lecture 3

Alignment Algorithms – BLAST / BLAST2

- ◆ **Approach (Basic Local Alignment Search Tool) [Altschul+ 1990]**
 - View sequences as sequences of short **words** (**k-tuple**)
 - DNA - 11 bases, protein - 3 amino acids
 - Create hash table of **neighborhood** (closely-matching) words
 - Use statistics to set threshold for “closeness”
 - Start from exact matches to neighborhood words
- ◆ **Motivation**
 - Good alignments should contain many close matches
 - Statistics can determine which matches are significant
 - Much more sensitive than % identity
 - Hashing can find matches in $O(n)$ time
 - Extending matches in both directions finds alignment
 - Yields high-scoring /maximum segment pairs (**HSP / MSP**)

CMSC 838T – Lecture 3

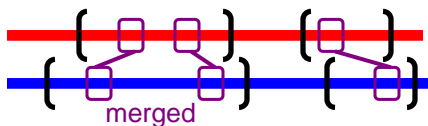
Alignment Algorithms – BLAST (Ungapped)



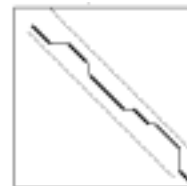
1) Convert 1st sequence into words (using all frames for given word size)

2) Calculate for each word list of “neighborhood” words (scoring threshold **T**) and enter in dictionary

3) Scan 2nd sequence, find matching words in dictionary, store locations



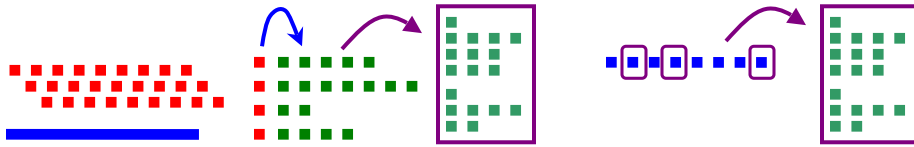
4) For each match, extend alignment in both directions while score above threshold **S**, merge segments



5) Align best segments using dynamic programming, report statistically significant matches

CMSC 838T – Lecture 3

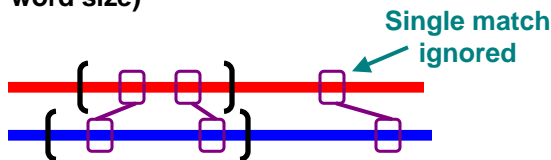
Alignment Algorithms – BLAST 2 (Gapped)



1) Convert 1st sequence into words (using all frames for given word size)

2) Calculate for each word list of “neighborhood” words (scoring threshold $T' < T$) and enter in dictionary

3) Scan 2nd sequence, find matching words in dictionary, store locations



4) If **two** nearby matches found on diagonal, form combined segment by extending alignment in both directions while score above threshold **S**

5) Align best segments using dynamic programming, report statistically significant matches



CMSC 838T – Lecture 3

Alignment - Low Complexity Regions (LCR)

◆ Regions of biased composition

◆ Examples

- Homopolymeric runs **A C T A A A A A A C T G**
- Short-period repeats **G A C A C A C A C G T A T**
- Over-represented residues **A T A C A A T A A G C A A**

◆ Yield meaningless high scores when aligned

◆ Frequently found

- Estimated 40% of human DNA
- Half of proteins in Protein Data Bank contain LCRs

◆ Software can find & mask LCRs in query sequence

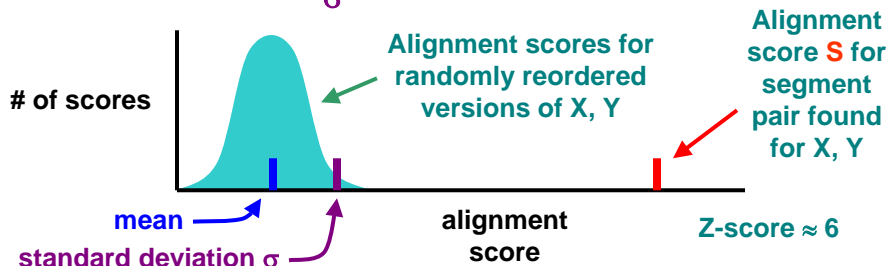
- Hard mask – replace w / wildcard (N or X) → treat as mismatch
- Soft mask – replace w / lowercase → ignore when building hash

CMSC 838T – Lecture 3

Statistical Significance of Alignment

◆ “Monte Carlo” test of significance

- Calculate score **S** for aligning sequences X, Y
- Repeat **R** times (usually **R** = 100)
 - Randomly reorder X and score alignment with original Y
 - Randomly reorder Y and score alignment with original X
- If **S** is “much” higher than random scores, then significant
 - **Z-score** = $\frac{|S - \text{mean}|}{\sigma}$ where σ = standard deviation

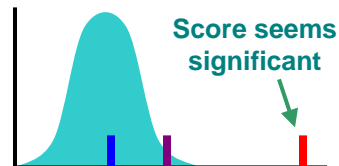


CMSC 838T – Lecture 3

Statistical Significance of Alignment

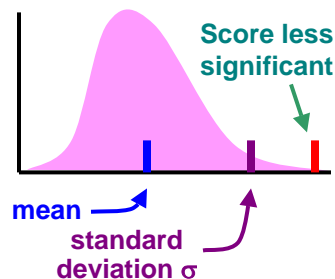
◆ Normal distribution

- Classic “bell curve”
- **Sum** of many random values
- Equivalent to alignment scores from randomly generated sequences (using **random** alignment)



◆ Extreme value distribution (EVD)

- **Maximum** of many random values
- Equivalent to alignment scores from randomly generated sequences (using **best** alignment)
- Standard deviation σ is significantly larger than for normal distribution



CMSC 838T – Lecture 3

Statistical Significance of Alignment

◆ Expected value E()

- Based on probability / statistics, the expected (random) result
- $E(S)$ = expected # of sequences found w/ alignment score S

◆ For ungapped alignment [Karlin & Altschul 1990]

- Can calculate E directly (using EVD)

$$E(S) \approx K \times N \times M \times e^{-\lambda S}$$

Where K, λ = constant scaling factors for scoring matrix, seqs.

N, M = lengths of sequences (or query & database size)

- Normalizing $S \rightarrow$ Bit-score $S' = \frac{-\lambda S - \ln K}{\ln 2} \rightarrow E(S') = N \times M \times 2^{-S'}$

◆ For gapped alignments

- Can estimate K, λ by curve-fitting EVD for random sequences
- Use estimated K, λ to calculate expected value E

CMSC 838T – Lecture 3

Interpretations of Expected Value

◆ Expected value ranges

- $E < 10^{-100}$ → very low, homologs or identical genes
- $E < 10^{-3}$ → moderate, may be related genes
- $E > 1$ → high, probably / may be unrelated
- $0.5 < E < 1$ → ??? In the “twilight zone” Try detailed search

◆ If database search

- long list of gradually declining of E values → large gene family
- long regions of moderate similarity → more significant than short regions of high identity

◆ Biological relevance

- Up to you, the biologist, to scrutinize alignments and determine if they are significant

CMSC 838T – Lecture 3

Summary

- ◆ **Examined pairwise sequence alignment**
 - Scoring metrics
 - PAM / BLOSUM for match / mismatch
 - Gap opening / extension penalties
 - Alignment algorithms
 - $O(n^2)$ – dynamic programming, dot matrix
 - $O(n)$ – FASTA, BLAST
 - Determining statistical significance
 - expected value **E**, bit-score **S'**, Z-score **Z**
- ◆ **Provides basis for multiple sequence alignment**
- ◆ **Still need to determine biological significance**