Bioinformatics

Sequence alignment methods

Martin Saturka

http://www.bioplexity.org/lectures/

EBI version 0.5

Creative Commons Attribution-Share Alike 2.5 License

Alignment

Approximate pattern matching - common sequence tasks

- sequence comparison, domains, motif search
- similarity and homology between sequences

Alignment approaches

- basic methods
  - dot matrices, scoring matrices
  - dynamic programming
- heuristic algorithms
  - word methods
  - fasta types, blast types
- multiple alignment
  - HMM approaches
  - MCMC based approaches

Proteins

sequences of aminoacid acyls: N-end to C-end ordering

flexible strands, enzymatic, signalling and structural functions

20 symbols alphabet

collagen helix triple: `Gly G, Pro P, HYP`
aliphatic: `Ala A, Val V, Leu L, Ile I;`
aromatic: `Phe F, Tyr Y, Trp W;`
polar: `Cys C, Ser S, Thr T - His H;`
`Met M, Pro P; Gln Q, Asn N; Tyr Y`
charged: `Asp D, Glu E; Lys K, Arg R;`
non-standard: selenocysteine `Sec`, pyrrolysine

- peptide backbone with peptide bonds
- twenty different standard residua

Dot matrix plot

simple search for diagonals of matched pairs

- sequence vs. sequence matrix plot
- dots on positions with matched pairs
- recurrences, identical regions

- diagonal smoothing
  - means of match/mismatch positions
  - mismatches and gaps just qualitatively

- how to count it more accurately:
  - similarity - scoring functions
  - gaps - start, prolongation

Similarity types

similarity definition: what is it?

- proteins
  - hydrophobic vs. hydrophilic
  - aromatic cycle
  - charge and polarity
  - size and flexibility
  - functional residua
  - secondary structure proneness
- nucleic acids
  - purine vs. pyrimidine
  - AT vs. GC
  - coding neutrality
  - RNA pairing preservation
- similarity
  - homology (ortho/para-logy) vs. homoplasy (convergence)

Similarity measures

log odds matrices

similarity scores based on percentage of changes

log odds:

- $M_{i,j} = \log q_{i,j}/(p_i \cdot p_j)$
- logarithm of the ratio of observed vs. expected frequences

matrix construction

- align two sequences of the same length
- count all the substitution mutations
  - unknown mutation direction $\rightarrow$ count it in the both ways
- compute the log odds
- linear transform and round the values

Log odd example

`01011010010101001010` $\Pr(0) = 21/40 = 0.525$

`01101011001001101010` $\Pr(1) = 19/40 = 0.475$

- expected alignment (mutation) frequencies:
  - $p_0 p_0 = (21/40)^2 = 0.275625$
  - $p_0 p_1 = (21/40) \cdot (19/40) = 0.249375$
  - $p_1 p_0 = (19/40) \cdot (21/40) = 0.249375$
  - $p_1 p_1 = (19/40)^2 = 0.225625$
- observed alignment (mutation) frequencies:
  - $0 \rightarrow 0$: $q_{00} = 14/40 = 0.35$
  - $0 \rightarrow 1$: $q_{01} = 7/40 = 0.175$
  - $1 \rightarrow 0$: $q_{10} = 7/40 = 0.175$
  - $1 \rightarrow 1$: $q_{11} = 12/40 = 0.3$
- the logarithm and the scoring:
  - $ln[(14 * 40)/(21 * 21)] = 0.23889191 \rightarrow 2$
  - $ln[(7 * 40)/(21 * 19)] = -0.35417181 \rightarrow -4$
  - $ln[(7 * 40)/(19 * 21)] = -0.35417181 \rightarrow -4$
  - $ln[(12 * 40)/(19 * 19)] = 0.28490815 \rightarrow 3$

Scoring matrices

- standard substitution scoring matrices
  - PAM for closely related species
  - BLOSUM for distant sequences
- PAM - Point Accepted Mutations
  - PAM1 for 1 point substitution per 100 aminoacids
  - PAMn computed as a stochastic processes
    - assumption of Markov process
    - amounts of subsequent changes
    - M2 = M1^2, Mn = M1^n
  - greater $n \rightarrow$ for greater evolutionary distances
- BLOSUM - BLOck SUbstitution Matrix
  - BLOSUMn
    - on datasets of sequences with at most n-% identity
    - BLOSUM100 from the total data sets
    - BLOSUM62 usually used
  - lesser $n \rightarrow$ for greater evolutionary distances

Dynamic programming approach

pairwise similarity alignment

- sequence vs. sequence matrix
- similarity, gaps computation
- analogy to the longest path search

- global search
  - to align sequence-to-sequence at whole lengths
  - Needleman-Wunsch algorithm
    - various modifications of the algorithm
- local search
  - to find maximally similar subsequence alignments
  - Smith-Waterman algorithm
    - itself a variation of the Needleman-Wunsch algorithm

Global search

- example

S_1: E S C H − E R

| | | . | |

S_2: − S C E N E −

  - alignment of two sequences
  - similarity scoring matrix
  - constant gap penalty measure

- algorithm
  - make a free matrix of size $(|S_1| + 1) \times (|S_2| + 1)$
    - rows, columns idexed by the two given sequences
  - gap values in the uppermost row and the leftmost column
    - zero-valued for tail ignoring
  - start in the upper left corner
  - pass rightward and downward
    - maximally valued alignments by taking maxima of sums of current alignments and passes to new positions

Global search example

- alignment of ESCHER and SCENE sequences

		E	S	C	H	E	R
	0	g	$2g$	$3g$	$4g$	$5g$	$6g$
S	g	$a_{1,1}$					
C	$2g$		$\searrow$ m	$\downarrow g_1$			
E	$3g$		$\rightarrow$ g_2	$a_{i,j}$			
N	$4g$						
E	$5g$						

sequences: S_1, S_2
similarity matrix: $s_{i,j}$
gap penalty: g
alignment scores: $a_{i,j}$

- start:
  - $a_{1,1} = \max(s_{S_1,S_2}, g, g)$
- iterate:
  - $a^m_{i,j} = a_{i-1,j-1} + s_{S_i,S_j}$
  - $a^{g_1}_{i,j} = a_{i-1,j} + g$, $a^{g_2}_{i,j} = a_{i,j-1} + g$
  - $a_{i,j} = \max(a^m_{i,j}, a^{g_1}_{i,j}, a^{g_2}_{i,j})$

Global search result

		E	S	C	H	E	R
	0	**-2**	-4	-6	-8	-10	-12
S	-2	0(m)	**2**(m)	0(g_2)	-2(g_2)	-4(g_2)	-6(g_2)
C	-4	-2(g_1)	0(g_1)	**11**(m)	9(g_2)	7(g_2)	5(g_2)
E	-6	1(m)	-1(g_2)	9(g_1)	**13**(m)	14(m)	12(g_2)
N	-8	-1(g_1)	2(m)	7(g_1)	**11**(g_1)	13(m)	14(m)
E	-10	-3(m)	0(g_1)	5(g_1)	9(m)	**16**(m)	**14**(g_2)

m, g_1, g_2 are for a match, gaps
$g = -2$ for the gap penalty used

part of the BLOSUM62 matrix:

	C	E	H	N	R	S
C	9	-4	-3	-3	-3	-1
E	-4	5	2	0	0	0
H	-3	2	8	1	0	-1
N	-3	0	1	6	0	1
R	-3	0	0	0	5	-1
S	-1	0	-1	1	-1	4

backward search `ESCH-ER`
for the alignment `-SCENE-`

Modifications

some of the global search variations

- linear memory usage with time doubling
  - Hirschberg's algorithm
- small alignments hashing
  - just log time speed-up

- Hirschberg's algorithm
  - quadratic memory needed for the backward search only
  - make two alignments - for the first, second half of S_1 and S_2
  - we find the middle point on S_1 of the whole alignment
  - iterate recursively on subparts of the S_1 string

Gaps

deletion / insertion regions

- a case of many small gaps is evolutionary less probable than a case of one (a few of) large gaps
- how to incorporate it into alignments?

- affine gap penalties
  - gap start - more bad
  - gap continuation - less bad
- $g = c_1 + c_2 \cdot$ gap length
  - necessary to keep subalignment values for the cases of gap passes just less good than match passes
- general gap penalties
  - harder to compute with such scenarios

Local alignment

search for similar subsequences

- local alignment specifics
  - it does not care about unrelated parts
  - it has to outline the similar regions
  - necessary (effective) gap penalties

- local alignment algorithm
  - start as with the global alignment but iterate with making all the subalignment values non-negative!
  - find the greatest subalignment value, can be anywhere inside the alignment iteration matrix
  - trace its path backward until zero value is approached
  - can search for all sufficiently high valued subalignments

Heuristic methods

how to slove daily requests

- sequence alignments the current top-most tasks
- exact methods find alignment optima, however under too high memory and time processing requirements
  - Needleman-Wunsch, Smith-Waterman algorithms
- heuristic methods necessary for the real world demands
- general heuristic alignment outline
  - start with limited local matches
  - enlarge matches while the alignment score is large enough
- two main approaches: fasta, blast types
  - for both nucleotide and amino acid sequences

Fasta

fasta algorithm principle

- find identity matches by the dot-plot matrix
- standard sizes of the k-tuples searched
  - length of 6 for nucleotides, length of 2 for amino acids
  - look-up table or hashing for substring storage of one string
- enlarge hot-spot parts of diagonals
  - usage of substitution scoring matrices
  - still no deletions, insertions allowed
- combine nearby subsequences into local alignments
  - make a weighted directed graph
  - weighted vertices are single diagonal alignments
  - edges between possibly adjoint subalignments
  - find the most weighted path on the graph

Blast program

the current way to do pairwise sequence alignment

- effective search on genome-large databases
- approximation of the Smith-Waterman algorithm
- sufficiently fast for the world demands
  - faster than fasta
  - much faster than the optimum guaranteed search
- less sensitive approach, still largelly sufficient

- word-based search method
  - genome databases, preprocessed in advance
  - query (target) sequence that is searched in the databases

Blast method

blast algorithm principle

- words inside genome databases indexed in advance
- three steps of the search process itself
  - word search, word list expanding, match enlarging

- look for exact matches
  - words from the query string against the word database
  - standard word lengths: 3 for amino acids, 11 for nucleotides
- enlarge low sensitivity
  - initial sensitivtiy low due to the search of exact matches only
  - take all the high similarity words of the indexed database
  - similarity by a chosen substitution scoring matrix
- find pairs of nearby high-score matches
  - requires to have at least two closely located matches
  - enlarge regions of alignment pairs while high scores
  - the enlarging stage takes most of the time

Statistics on alignment scores

ideal world case

- look for outliers in search results
- outliers according to z, standrad deviations
  - $s = [1/(n-1) \cdot \sum_i^n (x_i - \mu)]^{1/2}$
  - $z_i = (x_i - \mu)/s$
- normal distribution nice but not the case for us
- binomial and Poisson distribution more accurate ones
  - both of them are skewed to the right!
  - we can find high z-score cases more likely
- binomial distribution - two parameters: n and p
  - n - number of times of processing an experiment
  - p - probability of a success in an experiment
- Poisson distribution
  - good approximation of the binomial distribution for large n and small p values
  - $\Pr(X = x) = (1/x!) \cdot e^{-np} \cdot (np)^x$

Statistics on database search

real world case

- real search pitfalls
  - databases inhomogenous with respect to sequence types
  - need to adjust parameters for general search queries
  - search results contain probable amounts of sequences of the result similarity found by chance

- extreme value distribution - for maximum segment pairs
- probability estimation
  - $P(s > S) > 1 - \exp(-Kmne^{-vS})$
  - m, n - lengths of the query string, of the database
  - K, v - parameters that are to be adjusted
  - s - score variable, S - score cut off
- the expected amount: **expect** $\doteq Kmne^{-vS}$

Variations on the BLAST

a lot of software based on the standard blast program

- PSI-BLAST: position specific iterative blast
  - for evolutionary distant relatives of a given protein
  - usage of position specific scoring matrices (PSSM)
    - size of PSSM is 20$\times$ profile length
  - iterative work of the PSI-BLAST
    - closely related proteins are found first
    - a profile of the found proteins is made
    - PSSM created for the profile on the found proteins
    - new search done with the new PSSM

- PHI-BLAST: pattern hit initiated blast
  - for pattern search in protein databases
  - search for proteins which contain the pattern and are similar to the query sequence nearby the pattern

Multiple alignment

MSA - multiple sequence alignment

- pair-wise alignment based methods
  - from the most similar pairs
- probability based profiling methods
  - profiles based on stochastic assumptions
  - HMM methods, MCMC approaches

- usage for:
  - consensus sequence determination
  - motifs, domains characterization
  - phylogenetic analysis, cladistics
  - structure and function similarity

Pair-wise MSA

heuristic multiple alignment construction

- progressive multiple alignment
  clustering approach, phylogenetic tree construction
  - create a phylogenetic tree by a clustering
    bottom-up clustering, e.g. neighbor joining
  - find the most related sequences and align them
  - iterate the alignment along the tree
    i.e. make alignment for the other sequences

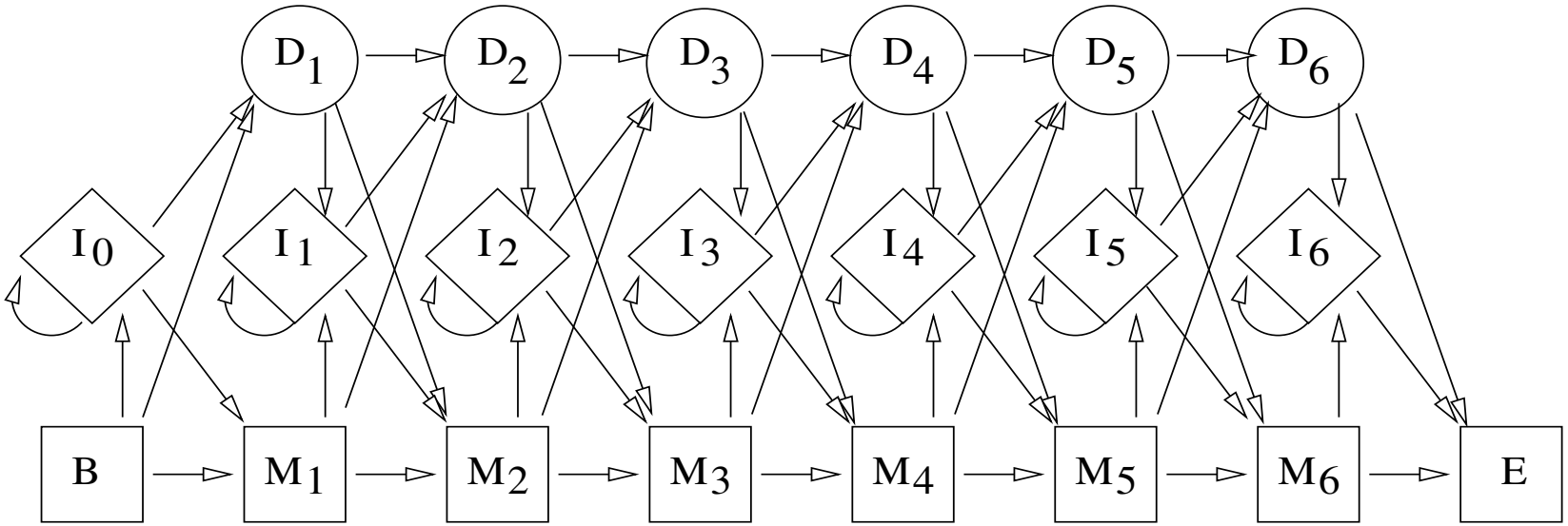
- iterative multiple alignment
  - start like the progressive multiple alignment
  - adds current alignment modifications
  - adjustig based on hill-climbing methods

HMM approach

assumption of no or weak distant interaction

- we do not look far away along the sequences
- good for proteins, not for nucleic acids
  - counting all possible matches and gaps
  - works nice with a given HMM profile
  - reasonable results on HMM profiling
- states of the HMMs
  - matches, insertions, deletions
  - matching a profile position to a sequence position
  - inserting to / deleting from a protein sequnce with respect to the given HMM profile

HMM example



SCHNEE: B - $M_{1,S}$ - $M_{2,C}$ - $M_{3,H}$ - $M_{4,N}$ - $M_{5,E}$ - $M_{6,E}$ - E

ESCHER: B - $I_{0,E}$ - $M_{1,S}$ - $M_{2,C}$ - $M_{3,H}$ - $D_{4,\_}$ - $M_{5,E}$ - $M_{6,R}$ - E

SCENE: B - $M_{1,S}$ - $M_{2,C}$ - $M_{3,E}$ - $M_{4,N}$ - $M_{5,E}$ - $D_{6,\_}$ - E

- B, E for *begin*, *end* states of the HMM alignment
- M_i, I_i, D_i for *match*, *insertion*, *deletion* states

HMM and domains

problems with gaps positioning between domains

- the model topology is for affine gap penalties
  - gaps can be rather large
  - Gibbs sampling as a solution
- model surgery
  - to remove under-used match states
  - to add match states in place of over-used insert states
- FIM - free inserion modules
  - added to both (start and stop) ends
  - without specificity on symbol outputs
  - probability ratio of inner insertions vs. FIM entry
    - decrease / icrease of amount of insertions inside domains

Markov chain Monte Carlo

- kind of sampling from a probability distribution
  - the Markov chain is constructed in such a way that its staionary distribution is the required distribution
  - several classes of methods for such constructions
  - iteration on the Markov chain has to lead to the required probability distribution, usually a slow process

- Gibbs sampling - one of the approaches
  - local alignments for a domain search
  - for distributions of *l*-mer alignments

Gibbs sampling

- genearal method outline
  - making samples of one variable out of many variables
  - subsequent samplings form a Markov chain
    - suitable values given by its stationary distribution
- herein, the sampling is of domain locations
  - positions of motifs inside given sequences as the stationary distributions
    - with high, single peaks demanded
- locations of putative *l*-mer domains
  - subsequent sampling of individual sequence locations
  - process finished when a local aligment is optimal
  - many trials and many start configurations necessary
    - dealing with problems of local optima

GS alignment

- start - choose random positions of *l*-mers inside sequences
- iterate the sampling:
  - take one of the sequences for the sampling process
  - make a (local) alignment of the rest *l*-mers
  - create profile out of the current alignment
  - try all the positions of *l*-mer on the taken sequence
  - compute probabilities of such *l*-mers given current profile
  - choose one position according to the computed probabilities
- stop - when total alignment scores do not increase

- Gibbs sampling obstacles
  - we have to take care about local optima
  - make projections on conserved residua
    - random subsequences of the *l*-mers for profile construction
  - the length of the putative domains is usually unknown

Distant domains

search for the least amount of sort reversals

- domain order
  - permutations of particular exons
- gene locations
  - rearrangements of chromosomes

- reversal algorithm, iterative
  - if there exists a decreasing strip
    - find the decreasing strip with the lowest number i inside
    - find the increasing strip with $i - 1$ number inside
    - revert the interim sequence, break amount decreases
  - otherwise revert an incresing strip
    - no more break originates, one decreasing strip more
  - finally no breaks, i.e. the identity permutation

Items to remember

Nota bene:

similarity and homology types

- Dynamic programming approach
  - Scoring matrices, gaps
  - Global and local alignments
  - Heuristics and statistics

- Multiple sequence alignment
  - HMMs: match, insertion, deletion states
  - MCMC approach, Gibbs sampling methods