

582606 Introduction to bioinformatics

Autumn 2007
Esa Pitkänen
Master's Degree Programme in Bioinformatics (MBI)
Department of Computer Science, University of Helsinki
<http://www.cs.helsinki.fi/mbi/courses/07-08/itb/>

Introduction to bioinformatics, Autumn 2007

1

Administrative issues

- Master level course
- Obligatory course in the Master's Degree Programme in Bioinformatics
- 4 credits
- Prerequisites: basic mathematical skills
- Lectures: Tuesdays and Fridays 14-16 in Exactum C222
- Exercises: Wednesday 14-16 in Exactum C221

Introduction to bioinformatics, Autumn 2007

2

Teachers

- Esa Pitkänen, Department of Computer Science, University of Helsinki
- Prof. Elja Arjas, Department of Mathematics and Statistics, University of Helsinki
- Prof. Samuel Kaski, Laboratory of Computer and Information Science, Helsinki University of Technology
- Lauri Eronen, Department of Computer Science, University of Helsinki

Introduction to bioinformatics, Autumn 2007

3

How to enrol for the course?

- Use the registration system of the Computer Science department: <https://ilmo.cs.helsinki.fi>
- If you don't have a student number or Finnish id yet, don't worry: attend the lectures and exercises, and register when you have the id

Introduction to bioinformatics, Autumn 2007

4

How to successfully pass the course?

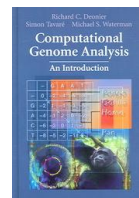
- You can get a maximum of 60 points
 - Course exam: maximum of 50 points
 - Exercises: maximum of 10 points
 - 0% completed assignments gives you 0 points, 80% gives 10 points, the rest by linear interpolation
 - "A completed assignment" means that you are willing to present your solution to the class in the exercise session
- Course will be graded on the scale 0-5
 - To get the lowest passing grade 1/5, you need to have at least 30 points
- Course exam: Wednesday 17.10. at 16.00-19.00 in A111

Introduction to bioinformatics, Autumn 2007

5

Course material

- Course book: Richard C. Deonier, Simon Tavare & Michael S. Waterman: Computational Genome Analysis – an Introduction, Springer 2005
- Available at Kumpula and Viikki science libraries; book stores (Amazon.com ~\$56, Akateeminen kirjakauppa ~75€, Yliopistokirjakauppa 71€)
- It is recommended that you have access to the course book!
- Slides for some lectures will be available on the course web page (copies in room C127)



Introduction to bioinformatics, Autumn 2007

6

Additional material

- Check the course web site
- N. C. Jones & P. A. Pevzner: An introduction to bioinformatics algorithms
- Alberts et al.: Molecular biology of the cell
- Lodish et al.: Molecular cell biology

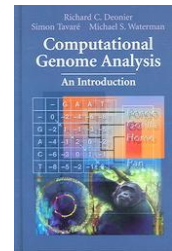


Introduction to bioinformatics, Autumn 2007

7

Course contents

- Biological background (book chapter 1)
- Probability calculus (chapters 2 and 3)
- Sequence alignment (chapter 6)
- Rapid alignment methods: FASTA and BLAST (chapter 7)
- Phylogenetic trees (chapter 12)
- Expression data analysis (chapter 11)



Introduction to bioinformatics, Autumn 2007

8

Master's Degree Programme in Bioinformatics (MBI)

- Two-year MSc programme
- Admission for 2008-2009 in January 2008
 - You need to have your Bachelor's degree ready by August 2008



MBI MASTER'S DEGREE
PROGRAMME IN BIOINFORMATICS

www.cs.helsinki.fi/mbi



News & events

Programme

Studies

Admission

People

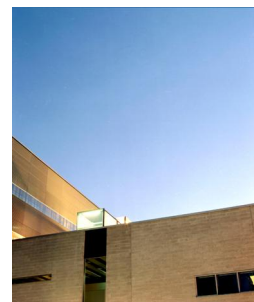
Contact

News and Events



MBI programme

- MBI educates bioinformatics professionals who
 - Specialise in computational and statistical methods
 - Work in R&D tasks in biology and medicine



Introduction to bioinformatics, Autumn 2007

10

MBI programme

- Two-year masters programme (120 cr)
- Offered jointly by the University of Helsinki (HY) and Helsinki University of Technology (TKK)
- Began in 2006 as a national programme, 2007 international admission
- Students 8 + 7 (2006 + 2007)



Introduction to bioinformatics, Autumn 2007

11

MBI programme organizers



Department of Computer Science, Department of Mathematics and Statistics, HY

Laboratory of Computer and Information Science, TKK



Faculty of Medicine, HY

Faculty of Biosciences, Faculty of Agriculture and Forestry, HY



Introduction to bioinformatics, Autumn 2007

12

Bioinformatics courses at the University of Helsinki

- Department of Computer Science
 - Practical course in biodatabases (II period): techniques for accessing and integrating data in biology databases.
 - Biological sequence analysis (III period): basic probabilistic methods for modelling and analysis of biological sequences.
 - Modeling of vision (III period): mechanisms and modeling of human perception.
 - Seminar: Regulatory networks (I & II periods)
 - Seminar: Management of biological databases (III & IV periods)

Introduction to bioinformatics, Autumn 2007

13

Bioinformatics courses at the University of Helsinki

- Department of Mathematics and Statistics
 - Statistical methods in genetic epidemiology and gene mapping (I period)
 - Mathematical modelling (I & II periods)
 - Practical course on phylogenetic analysis (IV period): recommended to take also *Biological sequence analysis*
 - Adaptive dynamics (III & IV periods)

Introduction to bioinformatics, Autumn 2007

14

Bioinformatics courses at Helsinki University of Technology

- Laboratory of Computer and Information Science
 - Computational genomics (I & II periods): Algorithms and models for biological sequences and genomics
 - Signal processing in neuroinformatics (I and II periods): overview of some of the main biomedical signal processing techniques
 - High-throughput bioinformatics (III and IV periods): computational and statistical methods for analyzing modern high-throughput biological data
 - Image analysis in neuroinformatics (III and IV periods): biomedical image processing techniques

Introduction to bioinformatics, Autumn 2007

15

Biology for methodological scientists (8 cr)

- Course organized by the Faculties of Bioscience and Medicine for the MBI programme
- Introduction to basic concepts of microarrays, medical genetics and developmental biology
- Book exam in I period (2 cr)
- Organized in three lectured modules, 2 cr each
- Each module has an individual registration so you can participate even if you missed the first module
- www.cs.helsinki.fi/mbi/courses/07-08/bfms/

Introduction to bioinformatics, Autumn 2007

16

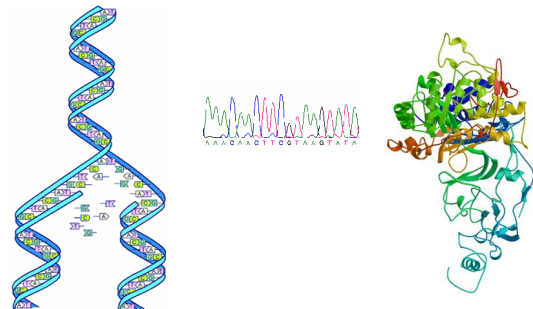
Bioinformatics courses

- Visit the website of Master's Degree Programme in Bioinformatics for up-to-date course lists:
<http://www.cs.helsinki.fi/mbi>

Introduction to bioinformatics, Autumn 2007

17

An introduction to bioinformatics



Introduction to bioinformatics, Autumn 2007

18

What is bioinformatics?

- ▮ Solving biological problems with computation?
- ▮ Collecting, storing and analysing biological data?
- ▮ Informatics - library science?

Introduction to bioinformatics, Autumn 2007

19

What is bioinformatics?

- ▮ Bioinformatics, *n.* The science of information and information flow in biological systems, esp. of the use of computational methods in genetics and genomics. (Oxford English Dictionary)
- ▮ "The mathematical, statistical and computing methods that aim to solve biological problems using DNA and amino acid sequences and related information."
-- Fredj Tekala
- ▮ "I do not think all biological computing is bioinformatics, e.g. mathematical modelling is not bioinformatics, even when connected with biology-related problems. In my opinion, bioinformatics has to do with management and the subsequent use of biological information, particular genetic information."
-- Richard Durbin

Introduction to bioinformatics, Autumn 2007

20

What is not bioinformatics?

- ▮ Biologically-inspired computation, e.g., genetic algorithms and neural networks
- ▮ However, application of neural networks to solve some biological problem, could be called bioinformatics
- ▮ What about DNA computing?

Introduction to bioinformatics, Autumn 2007

21

Related concepts

- ▮ Computational biology
 - Application of computing to biology (broad definition)
 - Often used interchangeably with bioinformatics
- ▮ Biometry: the statistical analysis of biological data
- ▮ Biophysics: "an interdisciplinary field which applies techniques from the physical sciences to understanding biological structure and function" -- British Biophysical Society
- ▮ Mathematical biology "tackles biological problems, but the methods it uses to tackle them need not be numerical and need not be implemented in software or hardware." -- Damian Counsell

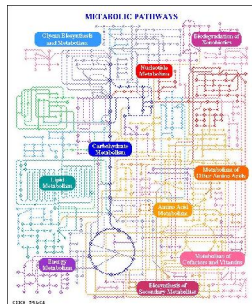
Introduction to bioinformatics, Autumn 2007

22

Related concepts

- Systems biology
 - "biology of networks"
 - integrating different levels of information to understand how biological systems work
- Computational systems biology

Overview of metabolic pathways in KEGG database, www.genome.jp/kegg/



Introduction to bioinformatics, Autumn 2007

23

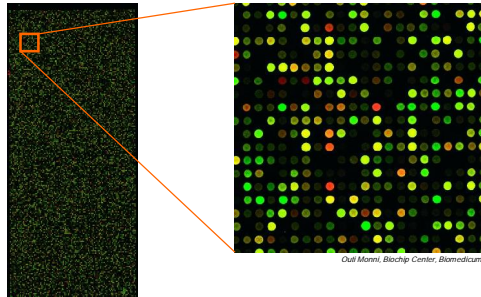
Why is bioinformatics important?

- ▮ New measurement techniques produce huge quantities of biological data
 - Advanced data analysis methods are needed to make sense of the data
 - Typical data sources produce noisy data with a lot of missing values
- ▮ Paradigm shift in biology to utilise bioinformatics in research
- ▮ To give you a glimpse of a typical situation in bioinformatics...

Introduction to bioinformatics, Autumn 2007

24

DNA microarray data



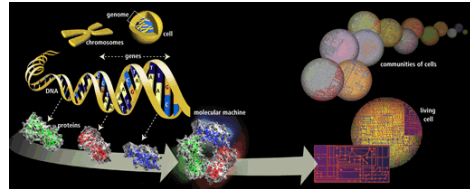
Outi Monni, Biochip Center, Biomedicum

Introduction to bioinformatics, Autumn 2007

25

Biological background

- Molecular Biology Primer: www.bioalgorithms.info

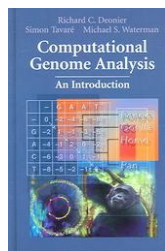


Introduction to bioinformatics, Autumn 2007

26

Today's program

- Biological background (book chapter 1)
 - Molecular primer continues
 - Recap of the most important material with respect to the course
- A word or two about exercises and the exam
- Note that the lecture on 9.10. is moved to room ExactumB222

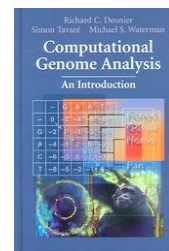


Introduction to bioinformatics, Autumn 2007

27

Course contents (18.9.)

- Biological background (book chapter 1)
- Probability calculus (chapters 2 and 3)
- **Sequence alignment (chapter 6)**
 - This week (18.9. and 21.9.)
- Rapid alignment methods: FASTA and BLAST (chapter 7)
 - Next week (25.9. and 28.9.)
- Phylogenetic trees (chapter 12)
- Expression data analysis (chapter 11)



Introduction to bioinformatics, Autumn 2007

28

Sequence Alignment (chapter 6)

- *The biological problem*
- Global alignment
- Local alignment
- Multiple alignment

Introduction to bioinformatics, Autumn 2007

29

Background: comparative genomics

- Basic question in biology: *what properties are shared among organisms?*
- Genome sequencing allows comparison of organisms at DNA and protein levels
- Comparisons can be used to
 - Find evolutionary relationships between organisms
 - Identify functionally conserved sequences
 - Identify corresponding genes in human and model organisms: develop models for human diseases

Introduction to bioinformatics, Autumn 2007

30

Homologs

- Two genes g_B and g_C evolved from the same ancestor gene g_A are called *homologs*
- Homologs usually exhibit conserved functions
- Close evolutionary relationship \Rightarrow expect a high number of homologs

$g_A = \text{agtgccgttaagtcggttc}$
 $g_B = \text{agtgccgttaaagttgtacgtc}$
 $g_C = \text{ctgactgtttgtgggttc}$

Introduction to bioinformatics, Autumn 2007

31

Sequence similarity

- Intuitively, similarity of two sequences refers to the degree of match between corresponding positions in sequence

$\text{agtgccgttaaagttgtacgtc}$
 $\text{ctgactgtttgtgggttc}$

- What about sequences that differ in length?

Introduction to bioinformatics, Autumn 2007

32

Similarity vs homology

- Sequence similarity is not sequence homology
 - If the two sequences g_B and g_C have accumulated enough mutations, the similarity between them is likely to be low

#mutations		#mutations	
0	$\text{agtgccgttaagtcggttc}$	64	$\text{acagtcggttcgggctattg}$
1	$\text{agtgccgttatagtcggttc}$	128	cagagcactaccgc
2	$\text{agtgccgcttatagtcggttc}$	256	$\text{cacgagtaagatatagct}$
4	$\text{agtgccgcttaaggcggttc}$	512	taatcgtgata
8	$\text{agtgccgcttcaaggcggttc}$	1024	$\text{acccttactactcctcgagtt}$
16	$\text{gggcccgttcagggggttc}$	2048	agcgacctgcccac
32	$\text{gcaggcgctcactgagggttc}$	4096	caaac

Homology is more difficult to detect over greater evolutionary distances.

Introduction to bioinformatics, Autumn 2007

33

Similarity vs homology (2)

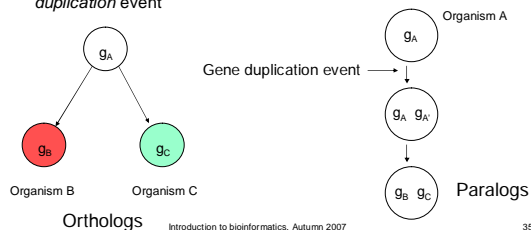
- Sequence similarity can occur by chance
 - Similarity does not imply homology*
- Consider comparing two short sequences against each other

Introduction to bioinformatics, Autumn 2007

34

Orthologs and paralogs

- We distinguish between two types of homology
 - Orthologs: homologs from two different species, separated by a *speciation* event
 - Paralogs: homologs within a species, separated by a *gene duplication* event

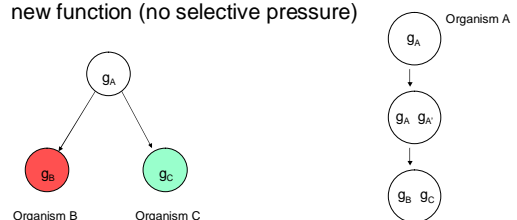


Introduction to bioinformatics, Autumn 2007

35

Orthologs and paralogs (2)

- Orthologs typically retain the original function
- In paralogs, one copy is free to mutate and acquire new function (no selective pressure)

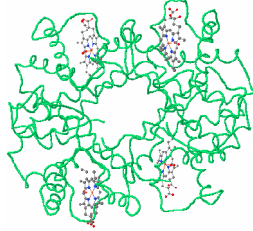


Introduction to bioinformatics, Autumn 2007

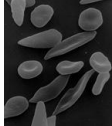
36

Paralogy example: hemoglobin

- Hemoglobin is a protein complex which transports oxygen
- In humans, hemoglobin consists of four protein subunits and four non-protein heme groups



Sickle cell diseases are caused by mutations in hemoglobin genes

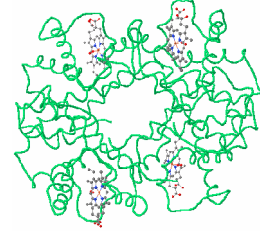


Hemoglobin A, www.rcsb.org/pdb/explore.do?structureId=1GZX
Introduction to bioinformatics, Autumn 2007

37

Paralogy example: hemoglobin

- In adults, three types are normally present
 - Hemoglobin A: 2 alpha and 2 beta subunits
 - Hemoglobin A2: 2 alpha and 2 delta subunits
 - Hemoglobin F: 2 alpha and 2 gamma subunits
- Each type of subunit (alpha, beta, gamma, delta) is encoded by a separate gene

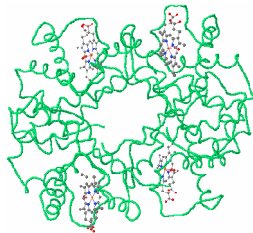


Hemoglobin A, www.rcsb.org/pdb/explore.do?structureId=1GZX
Introduction to bioinformatics, Autumn 2007

38

Paralogy example: hemoglobin

- The subunit genes are paralogs of each other, i.e., they have a common ancestor gene
- Demonstration in lecture: hemoglobin human paralogs in NCBI sequence databases
<http://www.ncbi.nlm.nih.gov/sites/entrez?db=Nucleotide>
 - Find human hemoglobin alpha, beta, gamma and delta
 - Compare sequences



Hemoglobin A, www.rcsb.org/pdb/explore.do?structureId=1GZX
Introduction to bioinformatics, Autumn 2007

39

Orthology example: insulin

- The genes coding for insulin in human (*Homo sapiens*) and mouse (*Mus musculus*) are orthologs:
 - They have a common ancestor gene in the ancestor species of human and mouse
 - Demonstration in lecture: find insulin orthologs from human and mouse in NCBI sequence databases

Introduction to bioinformatics, Autumn 2007

40

Sequence alignment

- Alignment specifies which positions in two sequences match

acgtctag	acgtctag	acgtctag
actctag-	-actctag	ac-tctag
2 matches	5 matches	7 matches
5 mismatches	2 mismatches	0 mismatches
1 not aligned	1 not aligned	1 not aligned

Introduction to bioinformatics, Autumn 2007

41

Mutations: Insertions, deletions and substitutions

Indel: insertion or deletion of a base with respect to the ancestor sequence

acgtctag
| | | | |
-actctag

Mismatch: substitution (point mutation) of a single base

- Insertions and/or deletions are called *indels*

- We can't tell whether the ancestor sequence had a base or not at indel position

Introduction to bioinformatics, Autumn 2007

42

Problems

- What sorts of alignments should be considered?
- How to score alignments?
- How to find optimal or good scoring alignments?
- How to evaluate the statistical significance of scores?

In this course, we discuss each of these problems briefly.

Course *Biological sequence analysis* tackles all four in-depth.

Introduction to bioinformatics, Autumn 2007

43

Sequence Alignment (chapter 6)

- The biological problem
- Global alignment*
- Local alignment
- Multiple alignment

Introduction to bioinformatics, Autumn 2007

44

Global alignment

- Problem: find optimal scoring alignment between two sequences (Needleman & Wunsch 1970)
- Every position in both sequences is included in the alignment
- We give score for each position in alignment

– Identity (match)	+1	WHAT
– Substitution (mismatch)	- μ	
– Indel	- δ	WH-Y
- Total score: sum of position scores

$$S(\text{WHAT}/\text{WH-Y}) = 1 + 1 - \delta - \mu$$

Introduction to bioinformatics, Autumn 2007

45

Dynamic programming

- How to find the optimal alignment?
- We use previous solutions for optimal alignments of smaller subsequences
- This general approach is known as dynamic programming

Introduction to bioinformatics, Autumn 2007

46

Introduction to dynamic programming: the money change problem

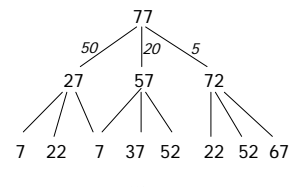
- Suppose you buy a pen for 4.23€ and pay for with a 5€ note
- You get 77 cents in change – what coins is the cashier going to give you if he or she tries to minimise the number of coins?
- The usual algorithm: start with largest coin (denominator), proceed to smaller coins until no change is left:
 - 50, 20, 5 and 2 cents
- This greedy algorithm is *incorrect*, in the sense that it does not always give you the correct answer

Introduction to bioinformatics, Autumn 2007

47

The money change problem

- How else to compute the change?
- We could consider all possible ways to reduce the amount of change
- Suppose we have 77 cents change, and the following coins: 50, 20, 5 cents
- We can compute the change with recursion
- Figure shows the recursion tree for the example



- Many values are computed more than once!
- This leads to a correct but very inefficient algorithm

Introduction to bioinformatics, Autumn 2007

48

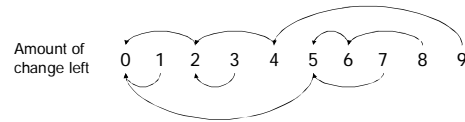
The money change problem

- We can speed the computation up by solving the change problem for all $i \leq n$
 - Example: solve the problem for 9 cents with available coins being 1, 2 and 5 cents
- Solve the problem in steps, first for 1 cent, then 2 cents, and so on
- In each step, utilise the solutions from the previous steps

Introduction to bioinformatics, Autumn 2007

49

The money change problem



- Algorithm runs in time proportional to Md , where M is the amount of change and d is the number of coin types
- The same technique of storing solutions of subproblems can be utilised in aligning sequences

Introduction to bioinformatics, Autumn 2007

50

Representing alignments and scores

Alignments can be represented in the following tabular form.

Each alignment corresponds to a path through the table.

WHAT

||

WH-Y

	-	W	H	A	T
-					
W			X		
H				X	X
Y					X

Introduction to bioinformatics, Autumn 2007

51

Representing alignments and scores

WHAT

||

WH-Y

Global alignment score $S_{3,4} = 2 - \delta - \mu$

	-	W	H	A	T
-	0				
W		1			
H			2	$2 - \delta$	
Y					$2 - \delta - \mu$

Introduction to bioinformatics, Autumn 2007

52

Filling the alignment matrix

	-	W	H	A	T
-					
W					
H					
Y					

Consider the alignment process at shaded square.

Case 1. Align H against H (match or substitution).

Case 2. Align H in WHY against - (indel) in WHAT.

Case 3. Align H in WHAT against - (indel) in WHY.

Introduction to bioinformatics, Autumn 2007

53

Filling the alignment matrix (2)

	-	W	H	A	T
-					
W					
H					
Y					

Scoring the alternatives.

Case 1. $S_{2,2} = S_{1,1} + s(2, 2)$

Case 2. $S_{2,2} = S_{1,2} - \delta$

Case 3. $S_{2,2} = S_{2,1} - \delta$

$s(i, j) = 1$ for matching positions,

$s(i, j) = -\mu$ for substitutions.

Choose the case (path) that yields the maximum score.

Keep track of path choices.

Introduction to bioinformatics, Autumn 2007

54

Global alignment: formal development

$A = a_1 a_2 a_3 \dots a_n$,

$B = b_1 b_2 b_3 \dots b_m$

$b_1 \quad b_2 \quad b_3 \quad b_4 \quad -$
 $- \quad a_1 \quad - \quad a_2 \quad a_3$

Any alignment can be written as a unique path through the matrix

Score for aligning A and B up to positions i and j:

$S_{ij} = S(a_1 a_2 a_3 \dots a_i, b_1 b_2 b_3 \dots b_j)$

Introduction to bioinformatics, Autumn 2007

55

	0	1	2	3	4
	-	b_1	b_2	b_3	b_4
0	-				
1	a_1				
2	a_2				
3	a_3				

Scoring partial alignments

Alignment of $A = a_1 a_2 a_3 \dots a_n$ with $B = b_1 b_2 b_3 \dots b_m$ can end in three ways

- Case 1: $(a_1 a_2 \dots a_{i-1}) a_i$
 $(b_1 b_2 \dots b_{j-1}) b_j$
- Case 2: $(a_1 a_2 \dots a_{i-1}) a_i$
 $(b_1 b_2 \dots b_j) -$
- Case 3: $(a_1 a_2 \dots a_i) -$
 $(b_1 b_2 \dots b_{j-1}) b_j$

Introduction to bioinformatics, Autumn 2007

56

Scoring alignments

Scores for each case:

- Case 1: $(a_1 a_2 \dots a_{i-1}) a_i$
 $(b_1 b_2 \dots b_{j-1}) b_j$
- Case 2: $(a_1 a_2 \dots a_{i-1}) a_i$
 $(b_1 b_2 \dots b_j) -$
- Case 3: $(a_1 a_2 \dots a_i) -$
 $(b_1 b_2 \dots b_{j-1}) b_j$

$$s(a_i, b_j) = \begin{cases} +1 & \text{if } a_i = b_j \\ -\mu & \text{otherwise} \end{cases}$$

$$s(a_i, -) = s(-, b_j) = -\delta$$

Introduction to bioinformatics, Autumn 2007

57

Scoring alignments (2)

- First row and first column correspond to initial alignment against indels:

$$S(i, 0) = -i \delta$$

$$S(0, j) = -j \delta$$

- Optimal global alignment score $S(A, B) = S_{n,m}$

		0	1	2	3	4
		-	b_1	b_2	b_3	b_4
0	-	0	$-\delta$	-2δ	-3δ	-4δ
1	a_1	$-\delta$				
2	a_2	-2δ				
3	a_3	-3δ				

Introduction to bioinformatics, Autumn 2007

58

Algorithm for global alignment

Input sequences A, B, $n = |A|$, $m = |B|$

Set $S_{i,0} := -\delta i$ for all i

Set $S_{0,j} := -\delta j$ for all j

for i := 1 to n

for j := 1 to m

$S_{ij} := \max\{S_{i-1,j} - \delta, S_{i-1,j-1} + s(a_i, b_j), S_{i,j-1} - \delta\}$

end

end

Algorithm takes $O(nm)$ time and space.

Introduction to bioinformatics, Autumn 2007

59

Global alignment: example

$\mu = 1$

$\delta = 2$

	-	T	G	G	T	G
-	0	-2	-4	-6	-8	-10
A	-2					
T	-4					
C	-6					
G	-8					
T	-10					?

Introduction to bioinformatics, Autumn 2007

60

Global alignment: example (2)

$$\mu = 1$$

$$\delta = 2$$

ATCGT-
| | |
-TGGTG

	-	T	G	G	T	G
-	0	-2	-4	-6	-8	-10
A	-2	-1	-3	-5	-7	-9
T	-4	-1	-2	-4	-4	-6
C	-6	-3	-2	-3	-5	-5
G	-8	-5	-2	-1	-3	-4
T	-10	-7	-4	-3	0	-2

Introduction to bioinformatics, Autumn 2007

61

Sequence Alignment (chapter 6)

- The biological problem
- Global alignment
- Local alignment
- Multiple alignment

Introduction to bioinformatics, Autumn 2007

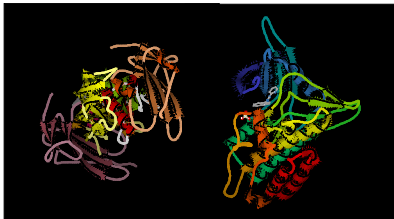
62

Local alignment: rationale

- Otherwise dissimilar proteins may have local regions of similarity
-> Proteins may share a function

Human bone morphogenic protein receptor type II precursor (left) has a 300 aa region that resembles 291 aa region in TGF- β receptor (right).

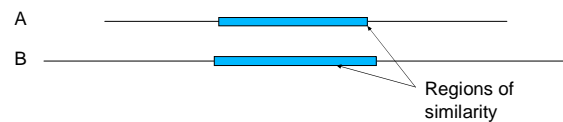
The shared function here is protein kinase.



Introduction to bioinformatics, Autumn 2007

63

Local alignment: rationale



- Global alignment would be inadequate
- Problem: find the highest scoring *local* alignment between two sequences
- Previous algorithm with minor modifications solves this problem (Smith & Waterman 1981)

Introduction to bioinformatics, Autumn 2007

64

From global to local alignment

- Modifications to the global alignment algorithm
 - Look for the highest-scoring path in the alignment matrix (not necessarily through the matrix), or in other words:
 - Allow preceding and trailing indels without penalty

Introduction to bioinformatics, Autumn 2007

65

Scoring local alignments

$$A = a_1 a_2 a_3 \dots a_n, B = b_1 b_2 b_3 \dots b_m$$

Let I and J be intervals (substrings) of A and B , respectively: $I \subset A, J \subset B$

Best local alignment score:

$$M(A, B) = \max\{S(I, J) : I \subset A, J \subset B\}$$

where $S(I, J)$ is the score for substrings I and J .

Introduction to bioinformatics, Autumn 2007

66

Allowing preceding and trailing indels

- First row and column initialised to zero:

$$M_{i,0} = M_{0,j} = 0$$

b1 b2 b3
- - a1

	0	1	2	3	4
0	-	b ₁	b ₂	b ₃	b ₄
1	0	0	0	0	0
2	a ₁	0			
3	a ₂	0			
4	a ₃	0			

Introduction to bioinformatics, Autumn 2007

67

Recursion for local alignment

$$M_{i,j} = \max \{ \begin{aligned} &M_{i-1,j-1} + s(a_i, b_j), \\ &M_{i-1,j} - \delta, \\ &M_{i,j-1} - \delta, \\ &0 \end{aligned} \}$$

	-	T	G	G	T	G
-	0	0	0	0	0	0
A	0	0	0	0	0	0
T	0	1	0	0	1	0
C	0	0	0	0	0	0
G	0	0	1	1	0	1
T	0	1	0	0	2	0

Introduction to bioinformatics, Autumn 2007

68

Finding best local alignment

- Optimal score is the highest value in the matrix

$$M(A, B) = \max\{S(I, J) : I \subset A, J \subset B\}$$

$$= \max_{i,j} M_{i,j}$$

- Best local alignment can be found by backtracking from the highest value in M

	-	T	G	G	T	G
0	0	0	0	0	0	0
A	0	0	0	0	0	0
T	0	1	0	0	1	0
C	0	0	0	0	0	0
G	0	0	1	1	0	1
T	0	1	0	0	2	0

Introduction to bioinformatics, Autumn 2007

69

Local alignment: example

	0	1	2	3	4	5	6	7	8	9	10
0	-	G	G	C	T	C	A	A	T	C	A
1	A	0									
2	C	0									
3	C	0									
4	T	0									
5	A	0									
6	A	0									
7	G	0									
8	G	0									

Introduction to bioinformatics, Autumn 2007

70

Local alignment: example

Scoring

Match: +2

Mismatch: -1

Indel: -2

C T - A A
C T C A A

	0	1	2	3	4	5	6	7	8	9	10
0	-	G	G	C	T	C	A	A	T	C	A
1	A	0	0	0	0	0	2	2	0	0	2
2	C	0	0	0	2	0	2	0	1	1	2
3	C	0	0	0	2	1	2	1	0	0	3
4	T	0	0	0	0	4	2	1	0	2	1
5	A	0	0	0	0	2	3	4	3	1	3
6	A	0	0	0	0	0	1	5	6	4	2
7	G	0	2	2	0	0	0	3	4	5	3
8	G	0	2	4	2	0	0	1	2	3	4

Introduction to bioinformatics, Autumn 2007

71

Non-uniform mismatch penalties

- We used uniform penalty for mismatches:

$$s('A', 'C') = s('A', 'G') = \dots = s('G', 'T') = \mu$$

- Transition mutations (A->G, G->A, C->T, T->C) are approximately twice as frequent than transversions (A->T, T->A, A->C, G->T)

- use non-uniform mismatch

penalties collected into a substitution matrix

	A	C	G	T
A	1	-1	-0.5	-1
C	-1	1	-1	-0.5
G	-0.5	-1	1	-1
T	-1	-0.5	-1	1

Introduction to bioinformatics, Autumn 2007

72

Gaps in alignment

- Gap is a succession of indels in alignment

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

- Previous model scored a length k gap as $w(k) = -k\delta$
- Replication processes may produce longer stretches of insertions or deletions
 - In coding regions, insertions or deletions of codons may preserve functionality

Introduction to bioinformatics, Autumn 2007

73

Gap open and extension penalties (2)

- We can design a score that allows the penalty opening gap to be larger than extending the gap:

$$w(k) = -\alpha - \beta(k - 1)$$

- Gap open cost α , Gap extension cost β
- Our previous algorithm can be extended to use $w(k)$ (not discussed on this course)

Introduction to bioinformatics, Autumn 2007

74

Amino acid sequences

- We have discussed mainly dna sequences
- Amino acid sequences can be aligned as well
- However, the design of the substitution matrix is more involved because of the larger alphabet
- More on the topic in the course Biological sequence analysis

Introduction to bioinformatics, Autumn 2007

75

Demonstration of the EBI web site

- European Bioinformatics Institute (EBI) offers many biological databases and bioinformatics tools at <http://www.ebi.ac.uk/>

Introduction to bioinformatics, Autumn 2007

76

Sequence Alignment (chapter 6)

- The biological problem
- Global alignment
- Local alignment
- Multiple alignment

Introduction to bioinformatics, Autumn 2007

77

Multiple alignment

- Consider a set of n sequences on the right
 - Orthologous sequences from different organisms
 - Paralogs from multiple duplications
- How can we study relationships between these sequences?

```
aggcgagctgctgagtgcta
cgttagattgacgtgac
ttccggctgacgac
gacacggcgaacgga
agtgtgcccacgagcaggac
gcgggctgtgagcgcta
aagcggcctgtgtgcccta
atgctgtgccagtgtgta
agtcgagccccgagtgac
agtccgagtc
actcgggtgc
```

Introduction to bioinformatics, Autumn 2007

78

Optimal alignment of three sequences

- Alignment of $A = a_1a_2\dots a_i$ and $B = b_1b_2\dots b_j$ can end either in $(-, b_j)$, (a_i, b_j) or $(a_i, -)$
- $2^2 - 1 = 3$ alternatives
- Alignment of A , B and $C = c_1c_2\dots c_k$ can end in $2^3 - 1$ ways: $(a_i, -, -)$, $(-, b_j, -)$, $(-, -, c_k)$, $(-, b_j, c_k)$, $(a_i, -, c_k)$, $(a_i, b_j, -)$ or (a_i, b_j, c_k)
- Solve the recursion using three-dimensional dynamic programming matrix: $O(n^3)$ time and space
- Generalizes to n sequences but impractical with moderate number of sequences

Introduction to bioinformatics, Autumn 2007

79

Multiple alignment in practice

- In practice, real-world multiple alignment problems are usually solved with heuristics
- Progressive multiple alignment
 - Choose two sequences and align them
 - Choose third sequence w.r.t. two previous sequences and align the third against them
 - Repeat until all sequences have been aligned
 - Different options how to choose sequences and score alignments

Introduction to bioinformatics, Autumn 2007

80

Multiple alignment in practice

- Profile-based progressive multiple alignment: CLUSTALW
 - Construct a distance matrix of all pairs of sequences using dynamic programming
 - Progressively align pairs in order of decreasing similarity
 - CLUSTALW uses various heuristics to contribute to accuracy

Introduction to bioinformatics, Autumn 2007

81

Additional material

- R. Durbin, S. Eddy, A. Krogh, G. Mitchison: Biological sequence analysis
- N. C. Jones, P. A. Pevzner: An introduction to bioinformatics algorithms
- Course Biological sequence analysis in Spring 2008

Introduction to bioinformatics, Autumn 2007

82

Chapter 7: Rapid alignment methods: FASTA and BLAST

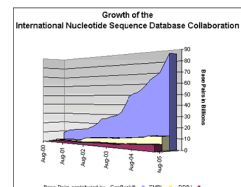
- The biological problem*
- Search strategies*
- FASTA
- BLAST

Introduction to bioinformatics, Autumn 2007

83

The biological problem

- Global and local alignment algorithms are slow in practice
- Consider the scenario of aligning a *query sequence* against a large database of sequences
 - New sequence with unknown function



- For instance, the size of NCBI GenBank in January 2007 was 65,369,091,950 bases (61,132,599 sequences)

Introduction to bioinformatics, Autumn 2007

84

Problem with large amount of sequences

- Exponential growth in both number and total length of sequences
- Possible solution: Compare against model organisms only
- With large amount of sequences, changes are that matches occur by random
 - Need for statistical analysis

Introduction to bioinformatics, Autumn 2007

85

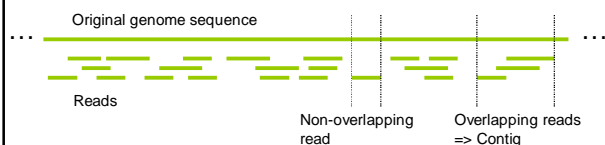
Application of sequence alignment: shotgun sequencing

- Shotgun sequencing is a method for sequencing whole-organism genomes
 - First, a large number of short sequences (~500-1000 bp), or reads are generated from the genome
 - Reads are contiguous subsequences (substrings) of the genome
 - Due to sequencing errors and repetitions in the reads, the genome has to be covered multiple times by reads

Introduction to bioinformatics, Autumn 2007

86

Shotgun sequencing

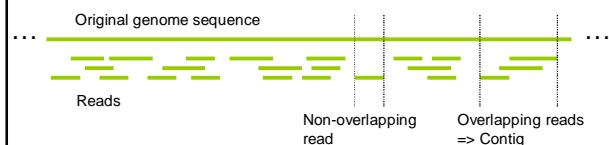


- Ordering of the reads is initially unknown
- Overlaps resolved by aligning the reads
- In a 3×10^9 bp genome with 500 bp reads and 5x coverage, there are $\sim 10^7$ reads and $\sim 10^7(10^7-1)/2 = \sim 5 \times 10^{13}$ pairwise sequence comparisons

Introduction to bioinformatics, Autumn 2007

87

Shotgun sequencing



- $\sim 5 \times 10^{13}$ pairwise sequence comparisons
- Recall that local alignment takes $O(nm)$ time, where n and m are sequence lengths
- Already with $n=m=500$, the computation cost is prohibitive

Introduction to bioinformatics, Autumn 2007

88

Search strategies

- How to speed up the computation?
 - Find ways to limit the number of pairwise comparisons
- Compare the sequences at word level to find out common words
 - Word means here a k-tuple (or a k-word), a substring of length k

Introduction to bioinformatics, Autumn 2007

89

Analyzing the word content

- Example query string l : TGATGATGAAGACATCAG
- For $k = 8$, the set of k -tuples of l is

TGATGATG
 GATGATGA
 ATGATGAA
 TGATGAAG
 ...
 GACATCAG

Introduction to bioinformatics, Autumn 2007

90

Analyzing the word content

- There are $n-k+1$ k -tuples in a string of length n
- If at least one word of I is not found from another string J , we know that I differs from J
- Need to consider statistical significance: I and J might share words by chance only
- Let $n=|I|$ and $m=|J|$

Word lists and comparison by content

- The k -words of I can be arranged into a table of word occurrences $L_w(I)$
- Consider the k -words when $k=2$ and $I=\text{GCATCGGC}$:

GC, CA, AT, TC, CG, GG, GC

AT: 3

CA: 2

CG: 5

GC: 1, 7 ← Start indices of k -word GC in I

GG: 6

TC: 4

Building $L_w(I)$ takes $O(n)$ time

Common k-words

- Number of common k -words in I and J can be computed using $L_w(I)$ and $L_w(J)$
- For each word w in I , there are $|L_w(J)|$ occurrences in J
- Therefore I and J have $\sum_w |L_w(I)| |L_w(J)|$ common words
- This can be computed in $O(n + m + 4^k)$ time
 - $O(n + m)$ time to build the lists
 - $O(4^k)$ time to calculate the sum

Common k-words

$I = \text{GCATCGGC}$

$J = \text{CCATCGGCATCG}$

$L_w(I)$	$L_w(J)$	Common words
AT: 3	AT: 3, 9	2
CA: 2	CA: 2, 8	2
	CC: 1, 7	0
CG: 5	CG: 5, 11	2
GC: 1, 7	GC: 6	2
GG: 6		0
TC: 4	TC: 4, 10	2
		10 in total

Properties of the common word list

- Exact matches can be found using binary search (e.g., where TCGT occurs in I ?)
 - $O(\log 4^k)$ time
- For large k , the table size is too large to compute the common word count in the previous fashion
- Instead, an approach based on merge sort can be utilised (details skipped, see course book)
- The common k -word technique can be combined with the local alignment algorithm to yield a rapid alignment approach

Chapter 7: Rapid alignment methods: FASTA and BLAST

- The biological problem
- Search strategies
- FASTA
- BLAST

FASTA

- FASTA is a multistep algorithm for sequence alignment (Wilbur and Lipman, 1983)
- The sequence file format used by the FASTA software is widely used by other sequence analysis software
- Main idea:
 - Choose regions of the two sequences that look promising (have some degree of similarity)
 - Compute local alignment using dynamic programming in these regions

Introduction to bioinformatics, Autumn 2007

97

FASTA outline

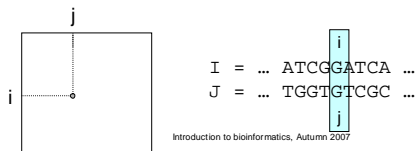
- FASTA algorithm has five steps:
 1. Identify common k-words between I and J
 2. *Score diagonals with k-word matches, identify 10 best diagonals*
 3. Rescore initial regions with a substitution score matrix
 4. Join initial regions using gaps, penalise for gaps
 5. Perform dynamic programming to find final alignments

Introduction to bioinformatics, Autumn 2007

98

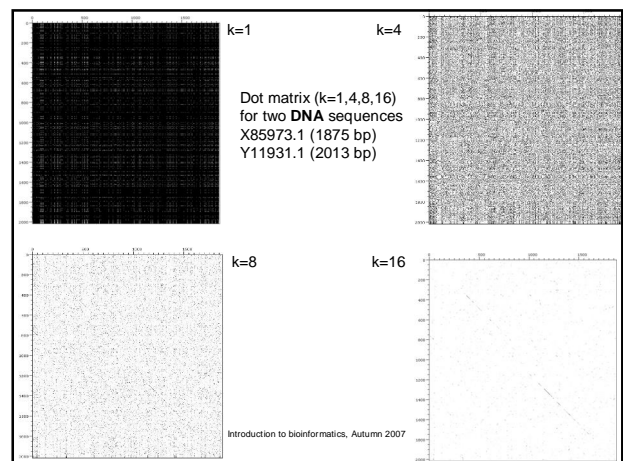
Dot matrix comparisons

- Word matches in two sequences I and J can be represented as a *dot matrix*
- Dot matrix element (i, j) has "a dot", if the word starting at position i in I is identical to the word starting at position j in J
- The dot matrix can be plotted for various k

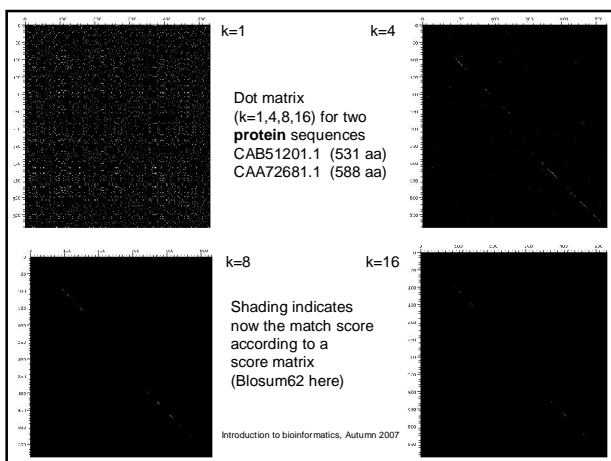


Introduction to bioinformatics, Autumn 2007

99



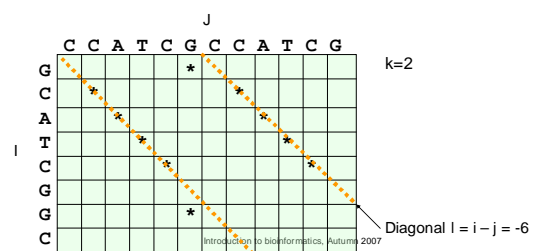
Introduction to bioinformatics, Autumn 2007



Introduction to bioinformatics, Autumn 2007

Computing diagonal sums

- We would like to find high scoring diagonals of the dot matrix
- Lets index diagonals by the offset, $l = i - j$



Introduction to bioinformatics, Autumn 2007

102

Computing diagonal sums

- As an example, let's compute diagonal sums for $I = \text{GCATCGGC}$, $J = \text{CCATCGCCATCG}$, $k = 2$
- 1. Construct k-word list $L_w(J)$
- 2. Diagonal sums S_i are computed into a table, indexed with the offset and initialised to zero

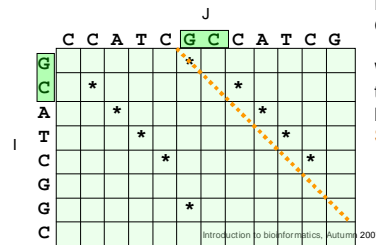
1	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
S_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Introduction to bioinformatics, Autumn 2007

103

Computing diagonal sums

- 3. Go through k-words of I , look for matches in $L_w(J)$ and update diagonal sums



For the first 2-word in I , GC , $L_{\text{GC}}(J) = \{6\}$.

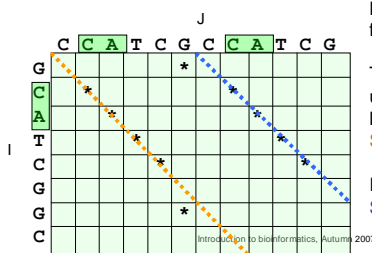
We can then update the sum of diagonal
 $I = i - j = 1 - 6 = -5$ to
 $S_{-5} := S_{-5} + 1 = 0 + 1 = 1$

Introduction to bioinformatics, Autumn 2007

104

Computing diagonal sums

- 3. Go through k-words of I , look for matches in $L_w(J)$ and update diagonal sums



Next 2-word in I is CA ,
for which $L_{\text{CA}}(J) = \{2, 8\}$.

Two diagonal sums are updated:

$I = i - j = 2 - 2 = 0$
 $S_0 := S_0 + 1 = 0 + 1 = 1$

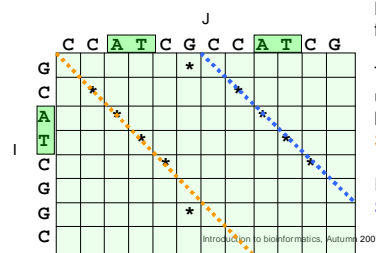
$I = i - j = 2 - 8 = -6$
 $S_{-6} := S_{-6} + 1 = 0 + 1 = 1$

Introduction to bioinformatics, Autumn 2007

105

Computing diagonal sums

- 3. Go through k-words of I , look for matches in $L_w(J)$ and update diagonal sums



Next 2-word in I is AT ,
for which $L_{\text{AT}}(J) = \{3, 9\}$.

Two diagonal sums are updated:

$I = i - j = 3 - 3 = 0$
 $S_0 := S_0 + 1 = 1 + 1 = 2$

$I = i - j = 3 - 9 = -6$
 $S_{-6} := S_{-6} + 1 = 1 + 1 = 2$

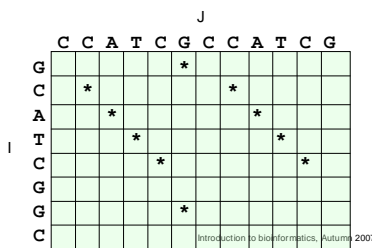
Introduction to bioinformatics, Autumn 2007

106

Computing diagonal sums

After going through the k-words of I , the result is:

1	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
S_1	0	0	0	0	4	1	0	0	0	4	1	0	0	0	0	0	0



Introduction to bioinformatics, Autumn 2007

107

Algorithm for computing diagonal sum of scores

$S_i := 0$ for all $1 - m \leq i \leq n - 1$

Compute $L_w(J)$ for all words w

for $i := 1$ to $n - k - 1$ do

$w := I_{i+1} \dots I_{i+k-1}$

for $j \in L_w(J)$ do

$I := i - j$

$S_I := S_I + 1$ ← Match score is here 1

end

end

Introduction to bioinformatics, Autumn 2007

108

FASTA outline

- FASTA algorithm has five steps:
 1. Identify common k-words between I and J
 2. Score diagonals with k-word matches, identify 10 best diagonals
 3. *Rescore initial regions with a substitution score matrix*
 4. *Join initial regions using gaps, penalise for gaps*
 5. Perform dynamic programming to find final alignments

Introduction to bioinformatics, Autumn 2007

109

Rescoring initial regions

- Each high-scoring diagonal chosen in the previous step is rescored according to a score matrix
- This is done to find subregions with identities shorter than k
- Non-matching ends of the diagonal are trimmed

I: C C A T C G C C A T C G 75% identity, no 4-word identities
J: C C A A C G C A A T C A

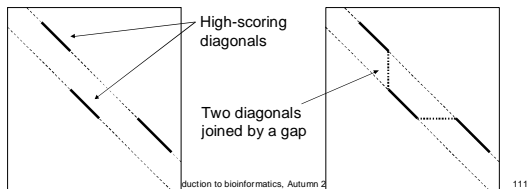
I': C C A T C G C C A T C G 33% identity, one 4-word identity
J': A C A T C A A A T A A A

Introduction to bioinformatics, Autumn 2007

110

Joining diagonals

- Two offset diagonals can be joined with a gap, if the resulting alignment has a higher score
- Separate gap open and extension are used
- Find the best-scoring combination of diagonals



Introduction to bioinformatics, Autumn 2007

111

FASTA outline

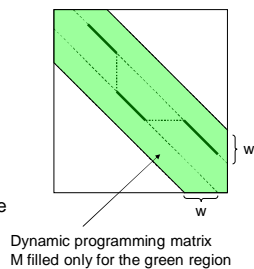
- FASTA algorithm has five steps:
 1. Identify common k-words between I and J
 2. Score diagonals with k-word matches, identify 10 best diagonals
 3. Rescore initial regions with a substitution score matrix
 4. Join initial regions using gaps, penalise for gaps
 5. *Perform dynamic programming to find final alignments*

Introduction to bioinformatics, Autumn 2007

112

Local alignment in the highest-scoring region

- Last step of FASTA: perform local alignment using dynamic programming around the highest-scoring
- Region to be aligned covers $-w$ and $+w$ offset diagonal to the highest-scoring diagonals
- With long sequences, this region is typically very small compared to the whole $n \times m$ matrix



Introduction to bioinformatics, Autumn 2007

113

Properties of FASTA

- Fast compared to local alignment using dynamic programming only
 - Only a narrow region of the full matrix is aligned
- Increasing parameter k decreases the number of hits: increases specificity, decreases sensitivity
- FASTA can be very specific when identifying long regions of low similarity
 - Specific method does not produce many incorrect results
 - Sensitive method produces many of the correct results

Introduction to bioinformatics, Autumn 2007

114

Properties of FASTA

- FASTA looks for initial exact matches to query sequence
 - Two proteins can have very different amino acid sequences and still be biologically similar
 - This may lead into a lack of sensitivity with diverged sequences

Introduction to bioinformatics, Autumn 2007

115

Demonstration of FASTA at EBI

- <http://www.ebi.ac.uk/fasta/>
- Note that parameter ktup in the software corresponds to parameter k in lectures

Introduction to bioinformatics, Autumn 2007

116

Chapter 7: Rapid alignment methods: FASTA and BLAST

- The biological problem
- Search strategies
- FASTA
- BLAST**

Introduction to bioinformatics, Autumn 2007

117

BLAST: Basic Local Alignment Search Tool

- BLAST (Altschul et al., 1990) and its variants are some of the most common sequence search tools in use
- Roughly, the basic BLAST has three parts:
 - Find local alignments between the query sequence and a database sequence ("seed hits")
 - Extend seed hits into high-scoring local alignments
 - Calculate p-values and a rank ordering of the local alignments
- High-scoring local alignments are called high scoring segment pairs (HSPs)
- Gapped BLAST introduced in 1997 allows for gaps in alignments

Introduction to bioinformatics, Autumn 2007

118

Finding seed hits

- First, we generate a set of *neighborhood sequences* for given k, match score matrix and threshold T
- Neighborhood sequences of a k-word w include all strings of length k that, when aligned against w, have the alignment score at least T
- For instance, let I = GCATCGGC, J = CCATCGCCATCG and k = 5, match score be 1, mismatch score be 0 and T = 4

Introduction to bioinformatics, Autumn 2007

119

Finding seed hits

- I = GCATCGGC, J = CCATCGCCATCG, k = 5, match score 1, mismatch score 0, T = 4
- This allows for one mismatch in each k-word
- The neighborhood of the first k-word of I, GCATC, is GCATC and the 15 sequences

$$\begin{matrix} \text{A} & & \text{A} & & \text{C} & & \text{A} & & \text{A} \\ \text{GCATC} & , & \text{G} & \text{GCATC} & , & \text{GC} & \text{GTC} & , & \text{GCA} & \text{CC} & , & \text{GCAT} & \text{G} \\ \text{T} & & \text{T} & & \text{T} & & \text{G} & & \text{T} \end{matrix}$$

Introduction to bioinformatics, Autumn 2007

120

Finding seed hits

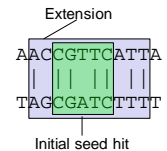
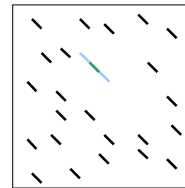
- I = GCATCGGC has 4 k-words and thus $4 \times 16 = 64$ 5-word patterns (seed hits) to locate in J
- These patterns can be found using exact search in time proportional to the sum of pattern lengths + length of J + number of matches (Aho-Corasick algorithm)
 - Methods for pattern matching are developed on course 58093 String processing algorithms

Introduction to bioinformatics, Autumn 2007

121

Extending seed hits: original BLAST

- Initial seed hits are extended
- Extensions do not add gaps to the alignment
- Sequence is extended into a HSP until the alignment score drops below the maximum attained score minus a threshold parameter value
- All statistically significant HSPs reported



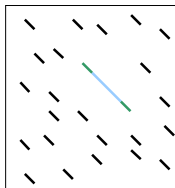
Altschul S.F., Gish W., Miller W., Myers E.W. and Lipman D.J., *J. Mol. Biol.*, 215, 403-410, 1990

Introduction to bioinformatics, Autumn 2007

122

Extending seed hits: gapped BLAST

- In later version of BLAST, two seed hits have to be found on the same diagonal
 - Hits have to be non-overlapping
 - If the hits are closer than A (additional parameter), then they are joined into a HSP
- Threshold value T is lowered to achieve comparable sensitivity
- If the resulting HSP achieves a score at least S_g , a *gapped extension* is triggered



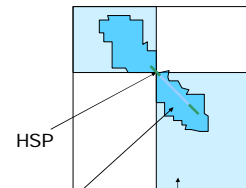
Altschul S.F., Madden T.L., Schäffer A.A., Zhang J., Zhang Z., Miller W., and Lipman D.J., *Nucleic Acids Res.* 1:25(17), 3389-402, 1997

Introduction to bioinformatics, Autumn 2007

123

Gapped extensions of HSPs

- Local alignment is performed starting from the HSP
- Dynamic programming matrix filled in "forward" and "backward" directions (see figure)
- Skip cells where value would be X_g below the best alignment score found so far



Region searched with score above cutoff parameter Region potentially searched by the alignment algorithm

Introduction to bioinformatics, Autumn 2007

124

Estimating the significance of results

- In general, we have a score $S(D, X) = s$ for a sequence X found in database D
- BLAST rank-orders the sequences found by p-values
- The p-value for this hit is $P(S(D, Y) \geq s)$ where Y is a random sequence
 - Measures the amount of "surprise" of finding sequence X
- A smaller p-value indicates more significant hit
 - A p-value of 0.1 means that one-tenth of random sequences would have as large score as our result

Introduction to bioinformatics, Autumn 2007

125

Estimating the significance of results

- In BLAST, p-values are computed roughly as follows
- There are nm places to begin an optimal alignment in the $n \times m$ alignment matrix
- Optimal alignment is preceded by a mismatch and has t matching (identical) letters
 - (Assume match score 1 and mismatch score 0)
- Let $p = P(\text{two random letters are equal})$
- The probability of having a mismatch and then t matches is $(1-p)p^t$

Introduction to bioinformatics, Autumn 2007

126

Estimating the significance of results

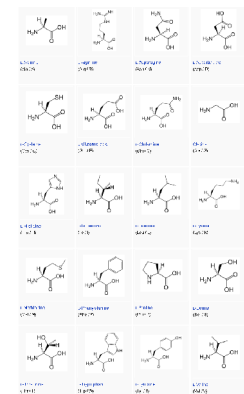
- We model this event by a Poisson distribution (why?) with mean $\lambda = nm(1-p)p^t$
- $P(\text{there is local alignment } t \text{ or longer}) = 1 - P(\text{no such event}) = 1 - e^{-\lambda} = 1 - \exp(-nm(1-p)p^t)$
- An equation of the same form is used in Blast:
- E-value = $P(S(D, Y) \geq s) \approx 1 - \exp(-nm\gamma\xi^s)$ where $\gamma > 0$ and $0 < \xi < 1$
- Parameters γ and ξ are estimated from data

Introduction to bioinformatics, Autumn 2007

127

Scoring amino acid alignments

- We need a way to compute the score $S(D, X)$ for aligning the sequence X against database D
- Scoring DNA alignments was discussed previously
- Constructing a scoring model for amino acids is more challenging
 - 20 different amino acids vs. 4 bases
- Figure shows the molecular structures of the 20 amino acids



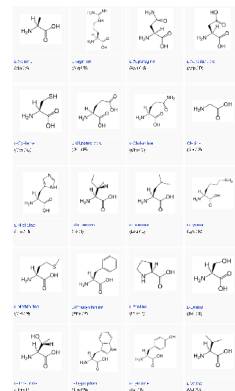
http://en.wikipedia.org/wiki/List_of_standard_amino_acids

Introduction to bioinformatics, Autumn 2007

128

Scoring amino acid alignments

- Substitutions between chemically similar amino acids are more frequent than between dissimilar amino acids
- We can check our scoring model against this



http://en.wikipedia.org/wiki/List_of_standard_amino_acids

Introduction to bioinformatics, Autumn 2007

129

Score matrices

- Scores $s = S(D, X)$ are obtained from score matrices
- Let $A = a_1a_2\dots a_n$ and $B = b_1b_2\dots b_n$ be sequences of equal length (no gaps allowed to simplify things)
- To obtain a score for alignment of A and B , where a_i is aligned against b_i , we take the ratio of two probabilities
 - The probability of having A and B where the characters match (match model M)
 - The probability that A and B were chosen randomly (random model R)

Introduction to bioinformatics, Autumn 2007

130

Score matrices: random model

- Under the random model, the probability of having X and Y is

$$P(A, B|R) = \prod_i q_{a_i} \prod_i q_{b_i}$$

where q_{x_i} is the probability of occurrence of amino acid type x_i

- Position where an amino acid occurs does not affect its type

Introduction to bioinformatics, Autumn 2007

131

Score matrices: match model

- Let p_{ab} be the probability of having amino acids of type a and b aligned against each other given they have evolved from the same ancestor c
- The probability is

$$P(A, B|M) = \prod_i p_{a_i b_i}$$

Introduction to bioinformatics, Autumn 2007

132

Score matrices: log-odds ratio score

- We obtain the score S by taking the ratio of these two probabilities

$$\frac{P(A,B|M)}{P(A,B|R)} = \frac{\prod_i p_{a_i b_i}}{\prod_i q_{a_i} \prod_i q_{b_i}} = \prod_i \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}}$$

and taking a logarithm of the ratio

$$S = \log_2 \frac{P(A,B|M)}{P(A,B|R)} = \sum_{i=1}^n \log_2 \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}} = \sum_{i=1}^n s(a_i, b_i)$$

Score matrices: log-odds ratio score

$$S = \log_2 \frac{P(A,B|M)}{P(A,B|R)} = \sum_{i=1}^n \log_2 \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}} = \sum_{i=1}^n s(a_i, b_i)$$

- The score S is obtained by summing over character pair-specific scores:

$$s(a, b) = \log_2 \frac{p_{ab}}{q_a q_b}$$

- The probabilities q_a and p_{ab} are extracted from data

Calculating score matrices for amino acids

- Probabilities q_a are in principle easy to obtain:
 - Count relative frequencies of every amino acid in a sequence database

$$s(a, b) = \log_2 \frac{p_{ab}}{q_a q_b}$$

Calculating score matrices for amino acids

- To calculate p_{ab} we can use a known pool of aligned sequences
- BLOCKS is a database of highly conserved regions for proteins
- It lists multiply aligned, ungapped and conserved protein segments
- Example from BLOCKS shows genes related to human gene associated with DNA-repair defect xeroderma pigmentosum

Block PRO0851A
 ID: XRODRMPGMNTB; BLOCK
 AC: PRO0851A; distance from previous block: (52,131)
 DE: Xeroderma pigmentosum group B protein; signature
 BL: adapted; width=21; seqs=8; 99.5%-985; strength=1287
 XPB_HUMAN [P19447 (74) RPLWVAPDGHIFLEAFSPVTK 54
 XPB_MOUSE [P49135 (74) RPLWVAPDGHIFLEAFSPVTK 54
 P91579 (80) RPLVAPDGHIFLEAFSPVTK 67
 XPB_BROME [Q02870 (84) RPLWVAPDGHIFLEAFSPVTK 79
 RA25_YEAST [Q00578 (131) PLWISPSDRIILSFSPPLAR 100
 Q38861 (52) RPLWVAPDGHIFLEAFSPVTK 71
 Q13748 (90) PLWISPSDRIILSFSPPLAR 100
 Q00835 (79) RPLWVAPDGHIFLEAFSPVTK 86

<http://blocks.fhcrc.org>

BLOSUM matrix

- BLOSUM is a score matrix for amino acid sequences derived from BLOCKS data
- First, count pairwise matches $f_{x,y}$ for every amino acid type pair (x, y)
- For example, for column 3 and amino acids L and W, we find 8 pairwise matches: $f_{L,W} = f_{W,L} = 8$

RPLWVAPD
 RPLWVAPR
 RPLWVAPN
 PLWISPSD
 RPLWACAD
 PLWINPID
 RPLWVCPD

Creating a BLOSUM matrix

- Probability p_{ab} is obtained by dividing f_{ab} with the total number of pairs (note difference with course book):

$$p_{ab} = f_{ab} / \sum_{x=1}^{20} \sum_{y=1}^x f_{xy}$$

- We get probabilities q_a by

$$q_a = \sum_{b=1}^{20} p_{ab}$$

RPLWVAPD
 RPLWVAPR
 RPLWVAPN
 PLWISPSD
 RPLWACAD
 PLWINPID
 RPLWVCPD

Creating a BLOSUM matrix

- The probabilities p_{ab} and q_a can now be plugged into

$$s(a, b) = \log_2 \frac{p_{ab}}{q_a q_b}$$

to get a 20 x 20 matrix of scores $s(a, b)$.

- Next slide presents the BLOSUM62 matrix
 - Values scaled by factor of 2 and rounded to integers
 - Additional step required to take into account expected evolutionary distance
 - Described in the course book in more detail

Introduction to bioinformatics, Autumn 2007

139

BLOSUM62

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	Z	X	*
A	4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-2	-1	1	0	-3	-2	0	-2	-1	0	-1	-4
R	-1	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1	-1	-3	-2	-3	-1	0	-1	-4
N	-2	0	6	1	-3	0	0	0	1	-3	-3	0	-2	-3	-2	1	0	-4	-2	-3	3	0	-1	-4
D	-2	-1	6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0	-1	-4	-3	-3	4	1	-1	-4	
C	0	-3	-3	9	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1	-1	-2	-2	-1	-3	-3	-2	-4	
Q	-1	1	0	0	-3	5	2	-2	0	-3	-2	1	0	-3	-1	0	-1	-2	-1	-2	0	3	-1	-4
E	-1	0	0	2	-4	2	5	-2	0	-3	-3	1	-2	-3	-1	0	-1	-3	-2	-2	1	4	-1	-4
G	0	-2	0	-1	-3	-2	-2	6	-2	-4	-4	-2	-3	-3	-2	0	-2	-2	-3	-3	-1	-2	-1	-4
H	-2	0	1	-1	-3	0	0	-2	8	-3	-3	-1	-2	-1	-2	-1	-2	-2	-2	-3	0	0	-1	-4
I	-1	-3	-3	-1	-3	-3	-4	-3	4	2	-3	1	0	-3	-2	-1	-3	-1	3	-3	-3	-1	-4	
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4	-2	2	0	-3	-2	-1	-2	-1	1	-4	-3	-1	-4
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5	-1	-3	-1	0	-1	-3	-2	-2	0	1	-1	-4
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5	0	-2	-1	-1	-1	1	-3	-1	-1	-4	
F	-2	-3	-3	-3	-2	-3	-3	-1	0	0	-3	0	6	-4	-2	-2	1	3	-1	-3	-3	-1	-4	
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7	-1	-1	-4	-3	-2	-2	-1	-2	-4
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4	1	-3	-2	0	0	0	0	-4
T	0	-1	0	-1	-1	-1	-2	-2	-1	-1	-1	-2	-1	1	5	-2	-2	0	-1	-1	0	-1	-4	
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11	2	-3	-4	-3	-2	-4
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	7	-1	-3	-2	-1	-4	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4	-3	-2	-1	-4
B	-2	-1	3	4	-3	0	1	-1	0	-3	-4	0	-3	-3	-2	0	-1	-4	-3	4	1	-1	-4	
Z	-1	0	0	1	-3	3	4	-2	0	-3	-3	1	-1	-3	-1	0	-1	-3	-2	-2	1	4	-1	-4
X	0	-1	-1	-1	-2	-1	-1	-1	-1	-1	-1	-1	-1	-2	0	0	-2	-1	-1	-1	-1	-1	-1	-4
*	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	-4	1

Introduction to bioinformatics, Autumn 2007

140

Using BLOSUM62 matrix

MQLEANADTSV

| | |

LQEQAQAQGE

$$s = \sum_{i=1}^{11} s(a_i, b_i)$$

$$= 2 + 5 - 3 - 4 + 4 + 0 + 4 + 0 - 2 + 0 + 1$$

$$= 7$$

Introduction to bioinformatics, Autumn 2007

141

Demonstration of BLAST at NCBI

- <http://www.ncbi.nlm.nih.gov/BLAST/>

Introduction to bioinformatics, Autumn 2007

142

Inferring the Past: Phylogenetic Trees (chapter 12)

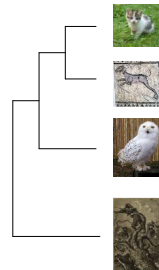
- The biological problem*
- Parsimony and distance methods
- Models for mutations and estimation of distances
- Maximum likelihood methods

Introduction to bioinformatics, Autumn 2007

143

Phylogeny

- We want to study ancestor-descendant relationships, or *phylogeny*, among groups of organisms
- Groups are called *taxa* (singular: *taxon*)
- Organisms are usually called *operational taxonomic units* or *OTUs* in the context of phylogeny

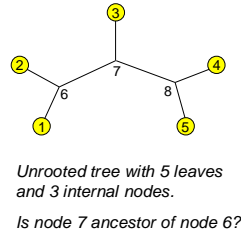


Introduction to bioinformatics, Autumn 2007

144

Phylogenetic trees

- Leaves (external nodes) ~ species, observed (OTUs)
- Internal nodes ~ ancestral species/divergence events, not observed
- Unrooted tree does not specify ancestor-descendant relationships beyond the observation "leaves are not ancestors"

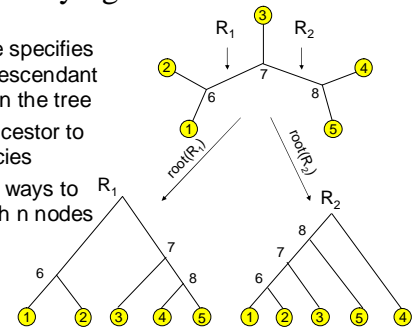


Introduction to bioinformatics, Autumn 2007

145

Phylogenetic trees

- Rooting a tree specifies all ancestor-descendant relationships in the tree
- Root is the ancestor to the other species
- There are $n-1$ ways to root a tree with n nodes



Introduction to bioinformatics, Autumn 2007

146

Questions

- Can we enumerate all possible phylogenetic trees for n species (or sequences?)
- How to score a phylogenetic tree with respect to data?
- How to find the best phylogenetic tree given data?

Introduction to bioinformatics, Autumn 2007

147

Finding the best phylogenetic tree: naive method

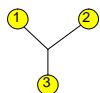
- How can we find the phylogenetic tree that best represents the data?
- Naive method: enumerate all possible trees
- How many different trees are there of n species?
- Denote this number by b_n

Introduction to bioinformatics, Autumn 2007

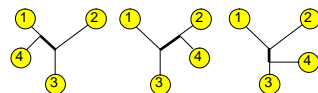
148

Enumerating unordered trees

- Start with the only unordered tree with 3 leaves ($b_3 = 1$)



- Consider all ways to add a leaf node to this tree



- Fourth node can be added to 3 different branches (edges), creating 1 new internal branch
- Total number of branches is n external and $n-3$ internal branches
- Unrooted tree with n leaves has $2n-3$ branches

Introduction to bioinformatics, Autumn 2007

149

Enumerating unordered trees

- Thus, we get the number of unordered trees

$$b_n = (2(n-1)-3)b_{n-1} = (2n-5)b_{n-1}$$

$$= (2n-5) * (2n-7) * \dots * 3 * 1$$

$$= (2n-5)! / ((n-3)!2^{n-3}), n > 2$$
- Number of rooted trees b'_n is

$$b'_n = (2n-3)b_n = (2n-3)! / ((n-2)!2^{n-2}), n > 2$$
 that is, the number of unordered trees times the number of branches in the trees

Introduction to bioinformatics, Autumn 2007

150

Number of possible rooted and unrooted trees

n	B _n	b _n
3	1	3
4	3	15
5	15	105
6	105	945
7	954	10395
8	10395	135135
9	135135	2027025
10	2027025	34459425
20	2.22E+020	8.20E+021
30	8.69E+036	4.95E+038

Introduction to bioinformatics, Autumn 2007

151

Too many trees?

- ▮ We can't construct and evaluate every phylogenetic tree even for a smallish number of species
- ▮ Better alternative is to
 - Devise a way to evaluate an individual tree against the data
 - Guide the search using the evaluation criteria to reduce the search space

Introduction to bioinformatics, Autumn 2007

152

Inferring the Past: Phylogenetic Trees (chapter 12)

- ▮ The biological problem
- ▮ *Parsimony and distance methods*
- ▮ Models for mutations and estimation of distances
- ▮ Maximum likelihood methods

Introduction to bioinformatics, Autumn 2007

153

Parsimony method

- ▮ The parsimony method finds the tree that explains the observed *sequences* with a minimal number of substitutions
- ▮ Method has two steps
 - Compute smallest number of substitutions for a given tree with a *parsimony algorithm*
 - Search for the tree with the minimal number of substitutions

Introduction to bioinformatics, Autumn 2007

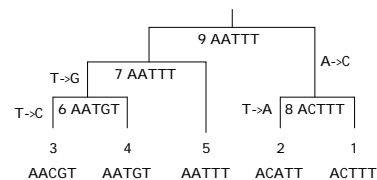
154

Parsimony: an example

- ▮ Consider the following short sequences
 - 1 ACTTT
 - 2 ACATT
 - 3 AACGT
 - 4 AATGT
 - 5 AATTT
- ▮ There are 105 possible rooted trees for 5 sequences
- ▮ Example: which of the following trees explains the sequences with least number of substitutions?

Introduction to bioinformatics, Autumn 2007

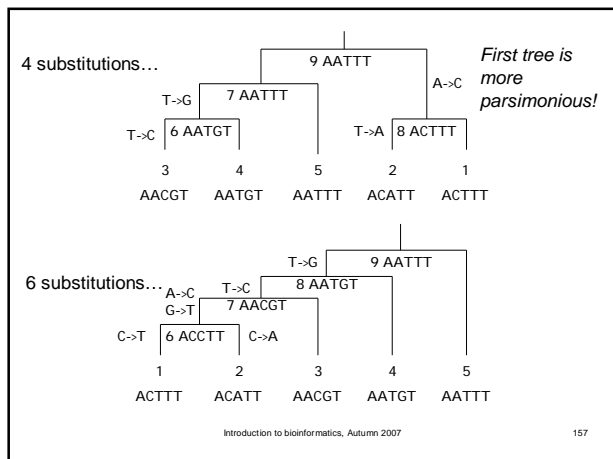
155



This tree explains the sequences with 4 substitutions

Introduction to bioinformatics, Autumn 2007

156



Computing parsimony

- Parsimony treats each site (position in a sequence) independently
- Total parsimony cost is the sum of parsimony costs of each site
- We can compute the minimal parsimony cost for a given tree by
 - First finding out possible assignments at each node, starting from leaves and proceeding towards the root
 - Then, starting from the root, assign a letter at each node, proceeding towards leaves

Introduction to bioinformatics, Autumn 2007 158

Labelling tree nodes

- An unrooted tree with n leaves contains $2n-1$ nodes altogether
- Assign the following labels to nodes in a rooted tree
 - leaf nodes: 1, 2, ..., n
 - internal nodes: $n+1, n+2, \dots, 2n-1$
 - root node: $2n$
- The label of a child node is always smaller than the label of the parent node

Introduction to bioinformatics, Autumn 2007 159

Parsimony algorithm: first phase

- Find out possible assignments at every node for each site u independently. Denote site u in sequence i by $s_{i,u}$.

For $i := 1, \dots, n$ do

$F_i := \{s_{i,u}\}$ % possible assignments at node i

$L_i := 0$ % number of substitutions up to node i

For $i := n+1, \dots, 2n-1$ do

Let j and k be the children of node i

If $F_j \cap F_k = \emptyset$ then $L_i := L_j + L_k + 1$, $F_i := F_j \cup F_k$

else $L_i := L_j + L_k$, $F_i := F_j \cap F_k$

Introduction to bioinformatics, Autumn 2007 160

Parsimony algorithm: first phase

Choose $u = 3$ (for example, in general we do this for all u)

$F_1 := \{T\}$
 $L_1 := 0$

$F_2 := \{A\}$
 $L_2 := 0$

$F_3 := \{C\}$, $L_3 := 0$

$F_4 := \{T\}$, $L_4 := 0$

$F_5 := \{T\}$, $L_5 := 0$

Introduction to bioinformatics, Autumn 2007 161

Parsimony algorithm: first phase

$F_6 := F_3 \cup F_4 = \{C, T\}$
 $L_6 := L_3 + L_4 + 1 = 1$

$F_7 := F_5 \cap F_6 = \{T\}$
 $L_7 := L_5 + L_6 = 1$

$F_8 := F_1 \cup F_2 = \{A, T\}$
 $L_8 := L_1 + L_2 + 1 = 1$

$F_9 := F_7 \cap F_8 = \{T\}$
 $L_9 := L_7 + L_8 = 2$

\Rightarrow Parsimony cost for site 3 is 2

Introduction to bioinformatics, Autumn 2007 162

Parsimony algorithm: second phase

- Backtrack from the root and assign $x \in F_i$ at each node
- If we assigned y at parent of node i and $y \in F_i$, then assign y
- Else assign $x \in F_i$ by random

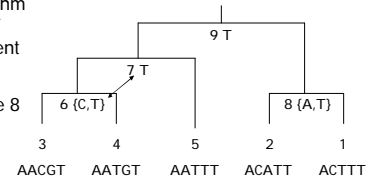
Introduction to bioinformatics, Autumn 2007

163

Parsimony algorithm: second phase

At node 6, the algorithm assigns T because T was assigned to parent node 7 and $T \in F_6$.

T is assigned to node 8 for the same reason.



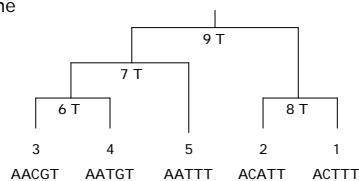
The other nodes have only one possible letter to assign

Introduction to bioinformatics, Autumn 2007

164

Parsimony algorithm

First and second phase are repeated for each site in the sequences, summing the parsimony costs at each site



Introduction to bioinformatics, Autumn 2007

165

Properties of parsimony algorithm

- Parsimony algorithm requires that the sequences are of same length
 - First align the sequences against each other and remove indels
 - Then compute parsimony for the resulting sequences
- Is the most parsimonious tree the correct tree?
 - Not necessarily but it explains the sequences with least number of substitutions
 - We can assume that the probability of having fewer mutations is higher than having many mutations

Introduction to bioinformatics, Autumn 2007

166

Finding the most parsimonious tree

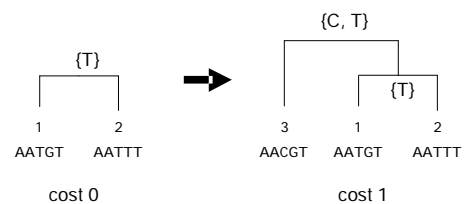
- Parsimony algorithm calculates the parsimony cost for a given tree...
- ...but we still have the problem of finding the tree with the lowest cost
- Exhaustive search (enumerating all trees) is in general impossible
- More efficient methods exist, for example
 - Probabilistic search
 - Branch and bound

Introduction to bioinformatics, Autumn 2007

167

Branch and bound in parsimony

- We can exploit the fact that adding edges to a tree can only increase the parsimony cost



Introduction to bioinformatics, Autumn 2007

168

Branch and bound in parsimony

Branch and bound is a general search strategy where

- Each solution is potentially generated
- Track is kept of the best solution found
- If a partial solution cannot achieve better score, we abandon the current search path

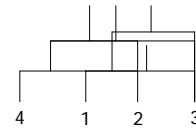
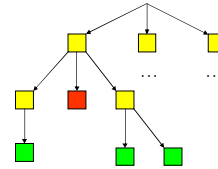
In parsimony...

- Start from a tree with 1 sequence
- Add a sequence to the tree and calculate parsimony cost
- If the tree is complete, check if found the best tree so far
- If tree is not complete and cost exceeds best tree cost, do not continue adding edges to this tree

Introduction to bioinformatics, Autumn 2007

169

Branch and bound graphically



Partial tree, no best complete tree constructed yet

Complete tree: calculate parsimony cost and store

Partial tree, cost exceeds the cost of the best tree this far

Introduction to bioinformatics, Autumn 2007

170

Distance methods

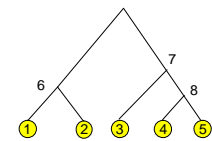
- The parsimony method works on sequence (character string) data
- We can also build phylogenetic trees in a more general setting
- Distance methods* work on a set of pairwise distances d_{ij} for the data
- Distances can be obtained from phenotypes as well as from genotypes (sequences)

Introduction to bioinformatics, Autumn 2007

171

Distances in a phylogenetic tree

- Distance matrix $D = (d_{ij})$ gives pairwise distances for *leaves* of the phylogenetic tree
- In addition, the phylogenetic tree will now specify distances between leaves and internal nodes
 - Denote these with d_{ij} as well



Distance d_{ij} states how far apart species i and j are evolutionary (e.g., number of mismatches in aligned sequences)

Introduction to bioinformatics, Autumn 2007

172

Distances in evolutionary context

- Distances d_{ij} in evolutionary context satisfy the following conditions
 - Symmetry: $d_{ij} = d_{ji}$ for each i, j
 - Distinguishability: $d_{ij} \neq 0$ if and only if $i \neq j$
 - Triangle inequality: $d_{ij} \leq d_{ik} + d_{kj}$ for each i, j, k
- Distances satisfying these conditions are called metric
- In addition, evolutionary mechanisms may impose additional constraints on the distances
 - \triangleright *additive* and *ultrametric* distances

Introduction to bioinformatics, Autumn 2007

173

Additive trees

- A tree is called *additive*, if the distance between any pair of leaves (i, j) is the sum of the distances between the leaves and the first node k that they share in the tree

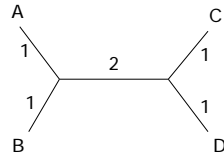
$$d_{ij} = d_{ik} + d_{jk}$$
- "Follow the path from the leaf i to the leaf j to find the exact distance d_{ij} between the leaves."

Introduction to bioinformatics, Autumn 2007

174

Additive trees: example

	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0



Introduction to bioinformatics, Autumn 2007

175

Ultrametric trees

- A rooted additive tree is called a *ultrametric tree*, if the distances between any two leaves i and j , and their common ancestor k are equal

$$d_{ik} = d_{jk}$$

- Edge length d_{ij} corresponds to the time elapsed since divergence of i and j from the common parent
- In other words, edge lengths are measured by a *molecular clock* with a constant rate

Introduction to bioinformatics, Autumn 2007

176

Identifying ultrametric data

- We can identify distances to be ultrametric by the three-point condition:
D corresponds to an ultrametric tree if and only if for any three **species** i , j and k , the distances satisfy $d_{ij} \leq \max(d_{ik}, d_{kj})$
- If we find out that the data is ultrametric, we can utilise a simple algorithm to find the corresponding tree

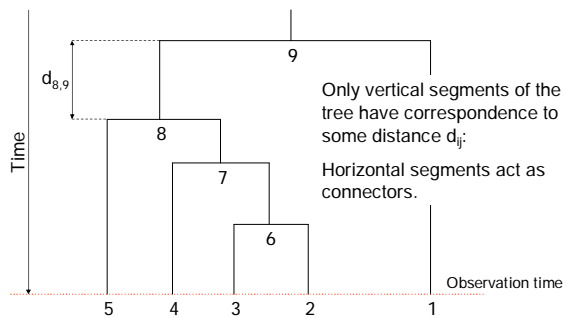
Introduction to bioinformatics, Autumn 2007

177

Ultrametric trees

Time

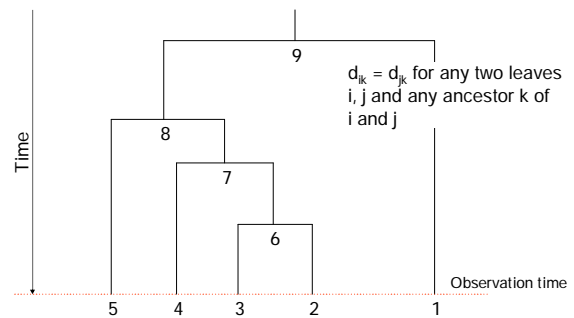
Ultrametric trees



Introduction to bioinformatics, Autumn 2007

179

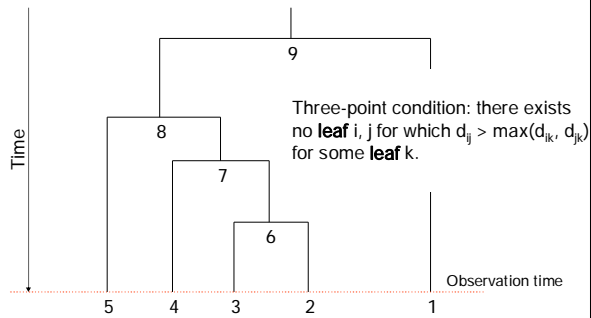
Ultrametric trees



Introduction to bioinformatics, Autumn 2007

180

Ultrametric trees



Introduction to bioinformatics, Autumn 2007

181

UPGMA algorithm

- UPGMA (unweighted pair group method using arithmetic averages) constructs a phylogenetic tree via clustering
- The algorithm works by at the same time
 - Merging two clusters
 - Creating a new node on the tree
- The tree is built from leaves towards the root
- UPGMA produces an ultrametric tree

Introduction to bioinformatics, Autumn 2007

182

Cluster distances

- Let distance d_{ij} between clusters C_i and C_j be

$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq}$$

that is, the average distance between points (species) in the cluster.

Introduction to bioinformatics, Autumn 2007

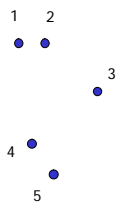
183

UPGMA algorithm

- Initialisation
 - Assign each point i to its own cluster C_i
 - Define one leaf for each sequence, and place it at height zero
- Iteration
 - Find clusters i and j for which d_{ij} is minimal
 - Define new cluster k by $C_k = C_i \cup C_j$, and define d_{kl} for all l
 - Define a node k with children i and j . Place k at height $d_{ij}/2$
 - Remove clusters i and j
- Termination:
 - When only two clusters i and j remain, place root at height $d_{ij}/2$

Introduction to bioinformatics, Autumn 2007

184



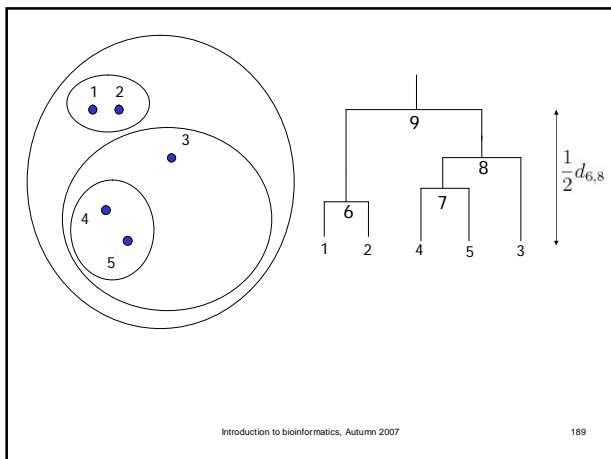
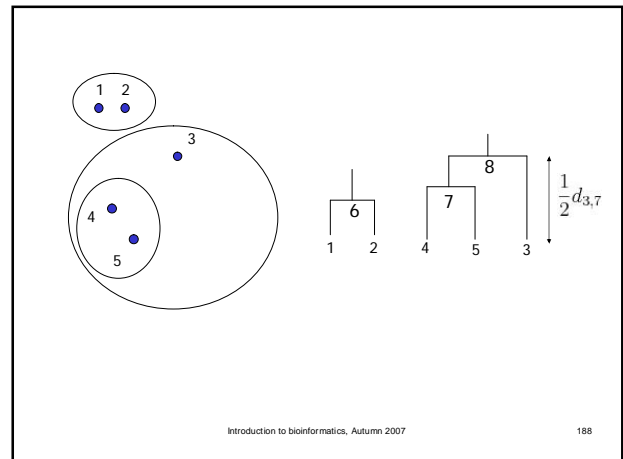
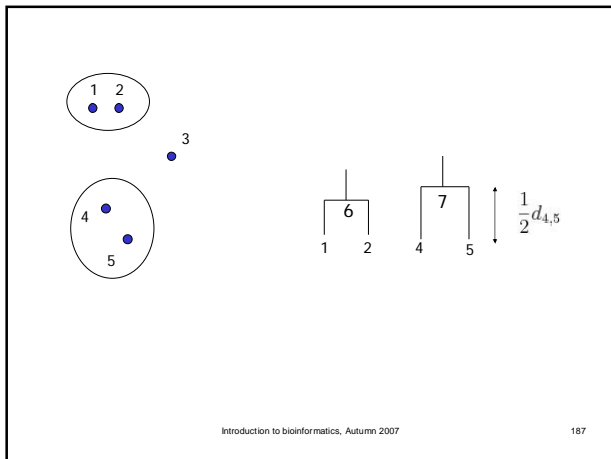
Introduction to bioinformatics, Autumn 2007

185



Introduction to bioinformatics, Autumn 2007

186



UPGMA implementation

- In naive implementation, each iteration takes $O(n^2)$ time with n sequences \Rightarrow algorithm takes $O(n^3)$ time
- The algorithm can be implemented to take only $O(n^2)$ time (Gronau & Moran, 2006)

Introduction to bioinformatics, Autumn 2007 190

Problem solved?

- We now have a simple algorithm which finds a ultrametric tree
 - If the data is ultrametric, then there is exactly one ultrametric tree corresponding to the data (we skip the proof)
 - The tree found is then the "correct" solution to the phylogeny problem, if the assumptions hold
- Unfortunately, the data is not ultrametric in practice
 - Measurement errors distort distances
 - Basic assumption of a molecular clock does not hold usually very well*

Introduction to bioinformatics, Autumn 2007 191

Incorrect reconstruction of non-ultrametric data by UPGMA

Diagram illustrating the incorrect reconstruction of non-ultrametric data by UPGMA. The left dendrogram shows the correct reconstruction of non-ultrametric data, while the right dendrogram shows an incorrect ultrametric reconstruction by UPGMA.

Tree which corresponds to non-ultrametric distances Incorrect ultrametric reconstruction by UPGMA algorithm

Introduction to bioinformatics, Autumn 2007 192

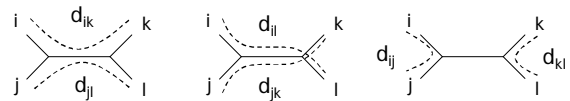
Checking for additivity

- How can we check if our data is additive?
- Let i, j, k and l be four *distinct* species
- Compute 3 sums: $d_{ij} + d_{kl}$, $d_{ik} + d_{jl}$, $d_{il} + d_{jk}$

Introduction to bioinformatics, Autumn 2007

193

Four-point condition



- The sums are represented by the three figures
 - Left and middle sum cover all edges, right sum does not
- Four-point condition:** i, j, k and l satisfy the four-point condition if two of the sums $d_{ij} + d_{kl}$, $d_{ik} + d_{jl}$, $d_{il} + d_{jk}$ are the same, and the third one is smaller than these two

Introduction to bioinformatics, Autumn 2007

194

Checking for additivity

- An $n \times n$ matrix D is additive if and only if the four point condition holds for every 4 distinct elements $1 \leq i, j, k, l \leq n$

Introduction to bioinformatics, Autumn 2007

195

Finding an additive phylogenetic tree

- Additive trees can be found with, for example, the neighbor joining method (Saitou & Nei, 1987)
- The neighbor joining method produces unrooted trees, which have to be rooted by other means
 - A common way to root the tree is to use an outgroup
 - Outgroup is a species that is known to be more distantly related to every other species than they are to each other
 - Root node candidate: position where the outgroup would join the phylogenetic tree
- However, in real-world data, even additivity usually does not hold very well

Introduction to bioinformatics, Autumn 2007

196

Neighbor joining algorithm

- Neighbor joining works in a similar fashion to UPGMA
 - Find clusters C_1 and C_2 that minimise a function $f(C_1, C_2)$
 - Join the two clusters C_1 and C_2 into a new cluster C
 - Add a node to the tree corresponding to C
 - Assign distances to the new branches
- Differences in
 - The choice of function $f(C_1, C_2)$
 - How to assign the distances

Introduction to bioinformatics, Autumn 2007

197

Neighbor joining algorithm

- Recall that the distance d_{ij} for clusters C_i and C_j was

$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq}$$

- Let $u(C_i)$ be the separation of cluster C_i from other clusters defined by

$$u(C_i) = \frac{1}{n-2} \sum_{C_j} d_{ij}$$

where n is the number of clusters.

Introduction to bioinformatics, Autumn 2007

198

Neighbor joining algorithm

- Instead of trying to choose the clusters C_i and C_j closest to each other, neighbor joining at the same time
 - Minimises the distance between clusters C_i and C_j and
 - Maximises the separation of both C_i and C_j from other clusters

Introduction to bioinformatics, Autumn 2007

199

Neighbor joining algorithm

- Initialisation as in UPGMA
- Iteration
 - Find clusters i and j for which $d_{ij} - u(C_i) - u(C_j)$ is minimal
 - Define new cluster k by $C_k = C_i \cup C_j$, and define d_{kl} for all l
 - Define a node k with children i and j . Remove clusters i and j
 - Assign length $\frac{1}{2} d_{ij} + \frac{1}{2} (u(C_i) - u(C_j))$ to the edge $i \rightarrow k$
 - Assign length $\frac{1}{2} d_{ij} + \frac{1}{2} (u(C_j) - u(C_i))$ to the edge $j \rightarrow k$
- Termination:
 - When only one cluster remains

Introduction to bioinformatics, Autumn 2007

200

Neighbor joining algorithm: example

	a	b	c	d	i	$u(i)$
a	0	6	7	5	a	$(6+7+5)/2 = 9$
b		0	11	9	b	$(6+11+9)/2 = 13$
c			0	6	c	$(7+11+6)/2 = 12$
d				0	d	$(5+9+6)/2 = 10$

i, j	d_{ij}	$-u(C_i)$	$-u(C_j)$	
a,b	6	-	9	$-13 = -16$
a,c	7	-	9	$-12 = -14$
a,d	5	-	9	$-10 = -14$
b,c	11	-	13	$-12 = -14$
b,d	9	-	13	$-10 = -14$
c,d	6	-	12	$-10 = -16$

Choose either pair to join

Introduction to bioinformatics, Autumn 2007

201

Neighbor joining algorithm: example

	a	b	c	d	i	$u(i)$
a	0	6	7	5	a	$(6+7+5)/2 = 9$
b		0	11	9	b	$(6+11+9)/2 = 13$
c			0	6	c	$(7+11+6)/2 = 12$
d				0	d	$(5+9+6)/2 = 10$

i, j	d_{ij}	$-u(C_i)$	$-u(C_j)$	
a,b	6	-	9	$-13 = -16$
a,c	7	-	9	$-12 = -14$
a,d	5	-	9	$-10 = -14$
b,c	11	-	13	$-12 = -14$
b,d	9	-	13	$-10 = -14$
c,d	6	-	12	$-10 = -16$

$d_{af} = \frac{1}{2} 6 + \frac{1}{2} (9 - 13) = 1$
 $d_{bf} = \frac{1}{2} 6 + \frac{1}{2} (13 - 9) = 5$

Introduction to bioinformatics, Autumn 2007

202

Inferring the Past: Phylogenetic Trees (chapter 12)

- The biological problem
- Parsimony and distance methods
- Models for mutations and estimation of distances*
- Maximum likelihood methods

Introduction to bioinformatics, Autumn 2007

203

Estimation of distances

- Many alternative ways to derive the distances d_{ij} exist
 - Simple solution: align each sequence pair and use the alignment score
 - This would not take into account that a change in base might revert back to the original base
 - We would then underestimate the distances
- Next: derivation of a simple stochastic model for the evolution of a DNA sequence
- Obtain the distances from the model

Introduction to bioinformatics, Autumn 2007

204

Estimation of distances

Key assumptions:

- mutations at sites are rare events in the course of time => poisson process
- sites evolve individually and by an identical mechanism
- number of mismatched bases is a sum of mutations at individual sites => binomial variable

Introduction to bioinformatics, Autumn 2007

205

A stochastic model for base substitutions

- Consider a single homologous site in two sequences
- Assume the sites diverged for time length t : the sites are separated by time $2t$
- Suppose that the number of substitutions in any branch of length t has a Poisson distribution with mean λt
- Probability that k substitutions occur is given by the Poisson probability $e^{-\lambda t} (\lambda t)^k / (k!)$, $k = 0, 1, 2, \dots$

Introduction to bioinformatics, Autumn 2007

206

Substitutions at one site

- General model: $P(\text{substitution results in base } j \mid \text{site was base } i) = m_{ij}$
- Felsenstein model: $m_{ij} = \pi_j$, with $\pi_j \geq 0$ and $\pi_1 + \pi_2 + \pi_3 + \pi_4 = 1$
 - The previous base does not affect the outcome!
- Assume that the set of probabilities π_j is same at every position in the sequence

Introduction to bioinformatics, Autumn 2007

207

Substitutions at one site (2)

- Probability $q_{ij}(t)$ that a base i at time 0 is substituted by a base j a time t later
- $q_{ij}(t) = e^{-\lambda t} + (1 - e^{-\lambda t}) \pi_j$, if $i = j$
- $q_{ij}(t) = (1 - e^{-\lambda t}) \pi_j$, otherwise

Introduction to bioinformatics, Autumn 2007

208

Substitutions at one site (3)

- We assume stationarity: distribution of base frequencies is the same for every time t
- In other words, we want that $P(\text{base a time } t \text{ later} = j) = \pi_j^0$ where π_j^0 is the frequency of base j at time 0.
- For our simple model, this can be shown to hold

Introduction to bioinformatics, Autumn 2007

209

Estimating distances

- Distances should take into account the mutation mechanism
- Average of λt substitutions occur at a particular site on a branch of length t
- However, some of the substitutions do not change the base ($A \rightarrow A$ or $A \rightarrow G \rightarrow A$, for example)

Introduction to bioinformatics, Autumn 2007

210

Mean number of substitutions in time t

- What is the chance H that a substitution actually changes a base?
- $H = \sum \pi_i(1 - \pi_i) = 1 - \sum \pi_i^2$
- Average number of real substitutions is then $\lambda t H$
- Distance K between two sequences is
 $K = 2\lambda t H$

Introduction to bioinformatics, Autumn 2007

211

Estimating distances from sequence data

- We want to estimate $K = 2\lambda t H$ from sequence data
- The chance $F_{ij}(t)$ that we observe a base i in one sequence and a base j in another is
 $F_{ij}(t) = \sum_l \pi_l q_{li}(t) q_{lj}(t)$
by averaging over the possible ancestral nucleotides

Introduction to bioinformatics, Autumn 2007

212

Estimating distances from sequence data

- Expression $F_{ij}(t) = \sum_l \pi_l q_{li}(t) q_{lj}(t)$ can be simplified by assuming that the mutation process is reversible:
 $\pi_i m_{ij} = \pi_j m_{ji}$ for all $i \neq j$
- From this it can be shown that
 $\pi_i q_{ij}(t) = \pi_j q_{ji}(t)$ for all i, j and $t > 0$
- Now the model simplifies into $F_{ij}(t) = \pi_i q_{ij}(2t)$

Introduction to bioinformatics, Autumn 2007

213

Estimating distances from sequence data

- What is the probability $F = F(t)$ that the bases at a particular position in two immediate descendants of the same ancestor are identical?
 $F = \sum_i \pi_i q_{ii}(2t) = e^{-2\lambda t} + (1 - e^{-2\lambda t})(1 - H)$

Introduction to bioinformatics, Autumn 2007

214

Putting the sites together

- Assume that
 - sites evolve independently of one other and
 - mutation process is identical at each site
 - The two sequences have been aligned against each other and gaps have been removed
- Do the bases at site i in the sequences differ?
 $X_i = 1$ if the ith pair of sites differs
 $X_i = 0$ otherwise

Introduction to bioinformatics, Autumn 2007

215

Putting the sites together (2)

- $P(X_i = 1) = 1 - F = (1 - e^{-2\lambda t})H$
- Now $D = X_1 + \dots + X_s$ is the number of mismatched pairs of bases
- D is a binomial random variable with parameters s and $1 - F$
- Notice that D is the Hamming distance for the sequences

Introduction to bioinformatics, Autumn 2007

216

Putting the sites together (3)

- ▮ F is unknown and has to be estimated from the sequence data
- ▮ Recall that the observed proportion of successes is a good estimator of the binomial success probability: estimate $1 - F$ with D/s
- ▮ $D/s = (1 - e^{-2\lambda t})H$
- ▮ $2\lambda t = -\log(1 - D/(sH))$
- ▮ Finally, we obtain $K = 2\lambda tH = -H \log(1 - D/(sH))$

Introduction to bioinformatics, Autumn 2007

217

Jukes-Cantor formula

- ▮ Estimate $2\lambda tH = -H \log(1 - D/(sH))$ of the distance K is known as the Jukes-Cantor formula
- ▮ When H (chance that a substitution actually occurs) approaches 1, the estimate decreases and approaches the Poisson mean $2\lambda t$
- ▮ H is usually not known and has to be estimated from the data as well

Introduction to bioinformatics, Autumn 2007

218

Inferring the Past: Phylogenetic Trees (chapter 12)

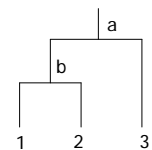
- ▮ The biological problem
- ▮ Parsimony and distance methods
- ▮ Models for mutations and estimation of distances
- ▮ *Maximum likelihood methods*

Introduction to bioinformatics, Autumn 2007

219

Maximum likelihood methods

- ▮ Consider the tree on the right with three sequences
- ▮ Probability $p(i_1, i_2, i_3)$ of observing bases i_1, i_2 and i_3 can be computed by summing over all possible ancestral bases,



$$p(i_1, i_2, i_3) = \sum_a \sum_b \pi_a q_{ai3}(t_2) q_{ab}(t_2 - t_1) q_{bi2}(t_1) q_{bi1}(t_1)$$

- ▮ Hard to compute for complex trees

Introduction to bioinformatics, Autumn 2007

220

Maximum likelihood estimation

- ▮ We would like to calculate likelihood $p(i_1, i_2, \dots, i_n)$ in the general case
- ▮ Calculations can be arranged using the peeling algorithm (see exercises)
- ▮ Basic idea is to move all summation signs as far to the right as possible

Introduction to bioinformatics, Autumn 2007

221

Maximum likelihood estimation

- ▮ Likelihood for the data is then obtained by multiplying the likelihoods of individual sites
- ▮ General recipe for maximum likelihood estimation:
 - Maximize over all model parameters for a *given* tree
 - Maximize previous expression over *all* possible trees

Introduction to bioinformatics, Autumn 2007

222

Problems with tree-building

- | Assumptions
 - Sites evolve independently of one other
 - Sites evolve according to the same stochastic model
 - The tree is rooted
 - The sequences are aligned
 - Vertical inheritance

Introduction to bioinformatics, Autumn 2007

223

Additional material on phylogenetic trees

- | Durbin, Eddy, Krogh, Mitchison: Biological sequence analysis
- | Jones, Pevzner: An introduction to bioinformatics algorithms
- | Gusfield: Algorithms on strings, trees, and sequences

Introduction to bioinformatics, Autumn 2007

224