Algorithms and Tools in Bioinformatics

Algorithms: Sequence Alignment
(adapted from Prof. Stephan Winkler)

Julia Vetter
julia.vetter@fh-hagenberg.at

SS2024

Course Content

Part I

Data, Tools, and Technologies

(1) Overview
(2) Standard Datasets/Modern File Formats
(3) Databases/Platforms
(4) Data (Pre-) Processing
(5) Tools
(6) Machine Learning

Part II

Algorithms: Sequence Alignment

(1) Motivation
(2) Similarity of Sequences/ Scoring matrices
(3) Global/Local Alignments
(4) Heuristic Methods
(5) Multiple Sequence Alignment
(6) Phylogenetic Trees

(1) Motivation

Why Sequence Comparison / Alignment?

- Biological background
  - Many proteins are members of families with similar biochemical function and/or common evolutionary origin
- Sequence comparison / alignment is done
  - to find functional, structural or evolutionary relationships
  - to identify conserved patterns
  - to find out something about an unknown / unidentified structure / protein
- Outside bioinformatics
  - structural comparison
  - plagiarism detection

Sequence Comparison

- From a computer science point of view, it is the comparison of strings over a given alphabet of characters
  - Comparison of nucleotide sequences: Alphabet consisting of the 4 characters for the 4 nucleic acids
  - Comparison of amino acid sequences: Alphabet consisting of the 20 characters for the 20 amino acids

- The **similarity** of two sequences is a measure of how well these sequences match.

- An **alignment** is the placing of one sequence on top of another to identify correspondence between similar characters or substrings.

So many possibilities to align sequences ...

```
actaccagttcatttgatacttctcaaa
             || |        |
taccattaccgtgttaactgaaaggacttaaagact
```

```
actaccagttcatttgatacttctcaaa
    |     | | | |         |
 taccattaccgtgttaactgaaaggacttaaagact
```

sequence 1

sequence 2

```
actaccagttcatttgatacttctcaaa
         | |     |       ||
    taccattaccgtgttaactgaaaggacttaaagact
```

```
actaccagttcatttgatacttctcaaa
  ||||| | |  |    |||      |
  taccattaccgtgttaactgaaaggacttaaagact
```

```
actaccagttcatttgatacttctcaaa
     |  ||    || ||  |   |
   taccattaccgtgttaactgaaaggacttaaagact
```

```
actaccagttcatttgatacttctcaaa
            |    |  |     ||
      taccattaccgtgttaactgaaaggacttaaagact
```

```
actaccagttcatttgatacttctcaaa
        | |  ||      |
        taccattaccgtgttaactgaaaggacttaaagact
```

```
   actaccagttcatttgatacttctcaaa
     |  | ||         |       ||
taccattaccgtgttaactgaaaggacttaaagact
```

```
      actaccagttcatttgatacttctcaaa
              |   |   |    |   | |
taccattaccgtgttaactgaaaggacttaaagact
```

```
 actaccagttcatttgatacttctcaaa
 |  |  |     ||
taccattaccgtgttaactgaaaggacttaaagact
```

Sequence Alignment

- Many alignments are possible
- Two sequences can always be aligned
- Sequence alignments must be evaluated (“scored)
- Often there can be more than one solution with the same score

mismatch

match

```
1 TGTCGATTAAGCGGTCGTAGCTGACCTGAGATTGCCCGATGGCGTAGTAGCTGACC 56
  |||||||||  ||||||||||   ||||||||| ||  |||||||||||||||||  |||||
1 TGTCGATTATGCGGTCGTAG..GACCTGAGTTTCCCCGATGGCGTAGTAGGTGACC 54
```

gap

Requirements for Alignment Algorithms (1)

- Algorithms must be able to evaluate gaps, insertions and replacements of different lengths, and consider the physico-chemical properties according to the biological background

- Algorithms are only accepted by the community if they work effectively, i.e. can compare an input sequence with all sequences in a database in a sufficiently short time - runtime efficiency!

- Dynamic Programming: Algorithm for searching a large solution space in $O(n^2)$ steps

Requirements for Alignment Algorithms (2)

- scoring system
  - Algorithms work correctly and mechanically on character strings, regardless of the application domain
  - Information is introduced from the application domain by specifying scores, which are a measure of the similarity of two symbols, and by introducing costs for gaps in the sequences – this is especially important for DNA/protein sequences

- Statistical evaluation of the results
  - Not only the score of an alignment has to be evaluated, but also the significance of the score depending on the length of the two character strings and the number of sequences stored in the database
  - Evaluation of the score depending on the expected value

- Databases
  - The sequences must be up-to-date

(2) Similarity of Sequences/ Scoring matrices

Calculation of Alignments

1. Selection of an appropriate scoring system, depending on the type of sequences to be compared

2. Strategy for calculating the cost of inserting gaps

3. Algorithm that, on the basis of 1) and 2), calculates an alignment with a maximum score

Scoring System – Substitution Matrix

Definition

A substitution matrix S over an alphabet $\Sigma = \{a_1, \ldots, a_\kappa\}$ has $\kappa \times \kappa$ entries, where each entry (i, j) assigns a score for a substitution of the letter a_i by the letter a_j in an alignment.

S	A	G	T	C
A	1			
G	-2	1		
T	-2	-2	1	
C	-2	-2	-2	1

Example subsitution matrix:
(used by BLAST)
$\Sigma = \{A, G, T, C\}$

For a, b $\in \Sigma$
$S(a_i, a_j) = 1$ für $a_i = a_j$
$S(a_i, a_j) = -2$ für $a_i \neq a_j$

Note: Gaps are not taken into account in substitution matrices!

Example

Example:

A = AGGACT
B = GTGAGT

P(A,B | Z) = (1/4)^6 * (1/4)^6 = ¼^12

P(A,B|Z) = Π p(a_i) Π p(b_i) (für i=1,n)

Score:
-2 + (-2) +1 +1 + (-2) +1 = -3

S	A	G	T	C
A	1			
G	-2	1		
T	-2	-2	1	
C	-2	-2	-2	1

Statistics Framework for Scoring Matrices

- Two models are compared using a likelihood function
  - Null hypothesis assumes that two sequences A and B are unrelated (in the sense of homologous, descended from a common ancestor)
  - Alignment is random, its probability is described by the model Z (for random).
  - Given a random arrangement of all symbols ai and bi in the alignment, the probability for the alignment is simply the product of the symbols occurring in the sequences:
    $P(A,B|Z) = \Pi\, p(a_i)\, \Pi\, p(b_i)$ (for i=1,n)

Statistics Framework for Scoring Matrices

- Two models are compared using a likelihood function
  - Hypothesis H1 is described by a model V (for related) that has two aligned symbols a_i and b_i with joint probabilities $q(a_i,b_i)$.
  - If the evolutionary divergence is to be evaluated, these values can, for example, express the probabilities that the symbols descend from a common ancestor c.
  - The probability of an alignment is then:
    $P(A,B|V) = \Pi\, q(a_i, b_i)$ (for i=1,n)

odds-ratio / log-odds ratio

- **odds-ratio** is the ratio of the likelihood values:

$$\frac{P(A,B\,|\,V)}{P(A,B\,|\,Z)} = \frac{\prod q(a_i,b_i)}{\prod p(a_i)\prod p(b_i)} = \prod \frac{q(a_i,b_i)}{p(a_i)\,p(b_i)}$$

- **log odds-ratio** is calculated using the logarithm (base arbitrary, often 2)

- ➔ ADDITIVE SCORING SCHEME

$$Sa_ib_i = log\,\frac{q(a_i,b_i)}{p(a_i)\,p(b_i)}$$

Note: Definition does not define how the relations / probabilites q(a_i, b_i) are calculated, most empirical / biological information

```
01. TGTGGT
02. TGCGGT
03. TGTGGT
04. GGTGGT
05. AGTGGT
06. AGTGGT
07. TGTGGT
08. TATGGT
09. TGTGGT
10. TGTGGT
11. TCTGGT
12. TGCGGT
13. TGTGGC
14. TGCGGT
15. TGAGGT
16. TGTGGT
17. TGTGGT
18. TGTGGA
19. TGTGGT
20. AGTGGT
21. TGTGGT
22. AGTGGT
23. TGCGGT
24. TGTGGC
25. TGCGGT
26. TGTGGT
27. TGCGGT
28. TCTGGT
29. TGTGGT
30. TGTGGT
31. TGTGGT
32. TATGGT
33. TGAGGT
34. CGTGGT
35. TGTGGT
36. TGAGGT
37. TGTGGT
...
57. TGAGGT
```

Example: Position Specific Scoring Matrix (PSSM) Matrix V$AML1_01 from TRANSFAC

Count nucleotides ($n_{i,j}$) at each position in *N aligned sequences*

Calculate frequency:
$f_{i,j} = n_{i,j} / N$

Calculate log-likelihood per position
$w_{i,j} = ln (f_{i,j} / p_i)$
p_i = *a priori probability* of symbol *0.25*

Count-Matrix

	T	G	T	G	G	T
A	5	2	4	0	1	1
C	1	2	14	0	0	4
G	2	52	1	57	55	0
T	49	1	38	0	1	52

Frequency-Matrix

	T	G	T	G	G	T
A	.09	.04	.07	.00	.02	.02
C	.02	.04	.25	.00	.00	.07
G	.04	.91	.02	1.0	.96	.00
T	.86	.02	.67	.00	.02	.91

Weigth-Matrix

	T	G	T	G	G	T
A	-1.0	-1.9	-1.2	-4.1	-2.5	-2.5
C	-2.5	-1.9	0.0	-4.1	-4.1	-1.2
G	-1.9	1.3	-2.5	1.4	1.3	-4.1
T	1.2	-2.5	1.0	-4.1	-2.5	1.3

Position Specific Scoring Matrix - Statistics

- The probability of a certain sequence given a frequency matrix can be calculated as follows:

$$P(S) = P(s_1, \ldots, s_n) = \prod_{i=1}^{n} P(s_i)$$

Frequency-Matrix

	T	G	T	G	G	T
A	.09	.04	.07	.00	.02	.02
C	.02	.04	.25	.00	.00	.07
G	.04	.91	.02	1.0	.96	.00
T	.86	.02	.67	.00	.02	.91

Example: CGAGGT:
.02 ×.91 ×.07 × 1.0 ×.96 ×.91 = 0.001

- As a comparison, the probability of the sequence under the null model is calculated.

- Example: all basis are equiprobable: $0.25^6 = 0.0002$

Position Specific Scoring Matrix - Statistics

- Every single sequence is very unlikely.
- We calculate the likelihood ratio to see what is more likely – that there is a relation (hypothesis H1) or that there is no connex (null hypothesis)

$$LR(s_1, \ldots, s_n) = \frac{\prod_i P(s_i)}{\prod_i Q(s_i)}$$

- Calculate the logarithm

$$LLR(S) = \log\left(\frac{\prod_i P(s_i)}{\prod_i Q(s_i)}\right) = \sum_{i=1}^{n} \log \frac{P(s_i)}{Q(s_i)}$$

- Example: 0.001 / 0.0002 = 5 => log likelihood score 1.6

Position Specific Scoring Matrix - Statistics

- Position specific scores:

$$Score_i(s_i) = \log \frac{P(s_i)}{Q(s_i)}$$

- Those are collected in the scoring matrix (weight matrix) and added:

$$Score(S) = \sum_i Score_i$$

Weigth-Matrix

	T	G	T	G	G	T
A	-1.0	-1.9	-1.2	-4.1	-2.5	-2.5
C	-2.5	-1.9	0.0	-4.1	-4.1	-1.2
G	-1.9	1.3	-2.5	1.4	1.3	-4.1
T	1.2	-2.5	1.0	-4.1	-2.5	1.3

- Example: **CGAGGT : –2.5 +1.3 –1.2 +1.4 +1.3 +1.3 = 1.6**

And now for something completely different…

from

- **Position Specific**

to

- **Position Independent Scoring**
  - Identity Matrices
  - PAMs
  - BLOSUMs

Identity Matrices

- Simplest scoring matrices
- All diagonal elements have the same positive value s
- All other elements have the same negative value s
- If s=- s, a local alignment must have more matches than mismatches to get a positive total score - usually finds short, compact alignments
- If s>>- s, a match compensates for several mismatches and long and weaker alignments are usually formed

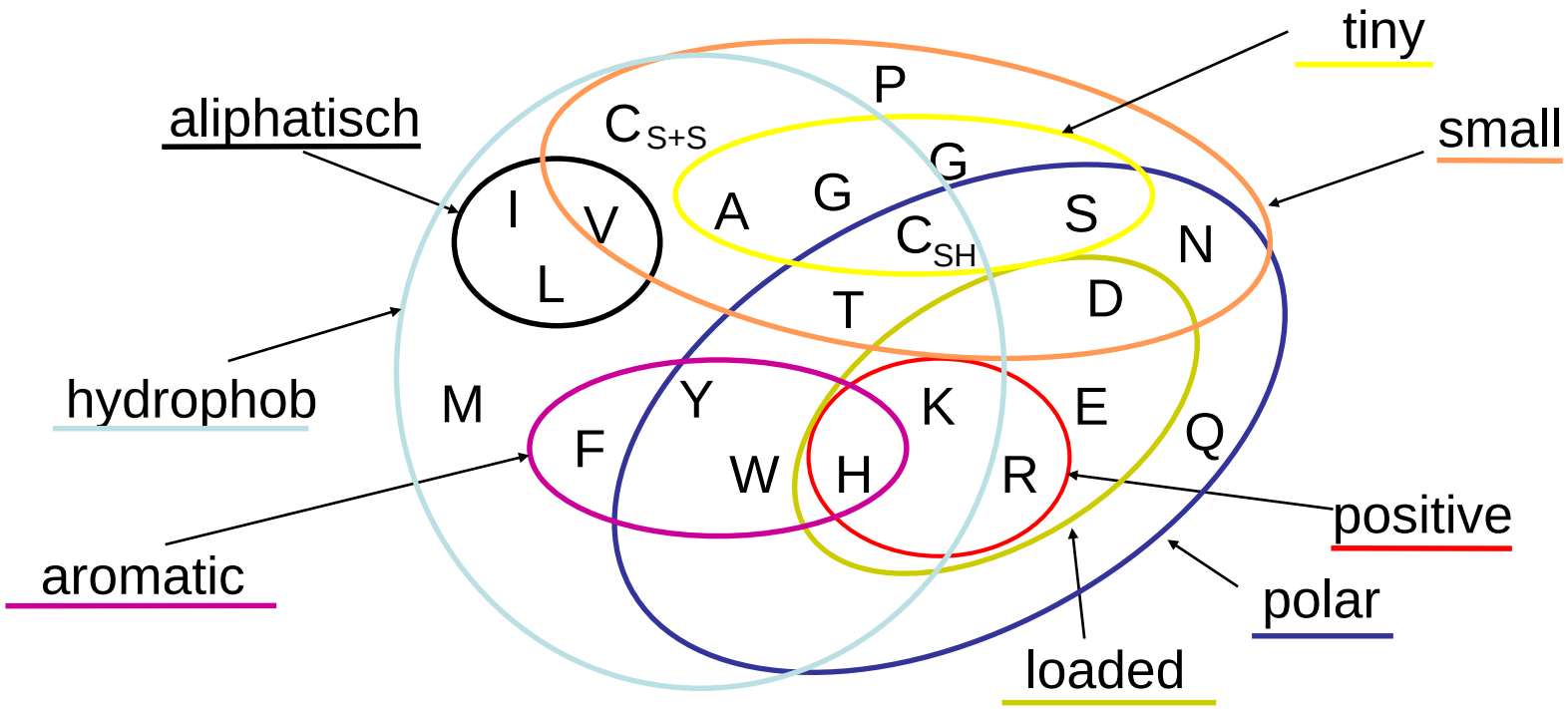
Scoring Matrices for Proteins

- Similarity measures beyond equal/unequal comparisons, implicit model of evolution:
  - Not all amino acids are the same
  - Some are more easily replaced than others
  - Certain mutations happen more easily than others
  - Some exchanges last longer than others
  - Mutations favor certain exchanges
  - Some amino acids have similar codons (AGU serine, AGA arginine)
  - These are more likely to be mutated by mutation of the DNA
  - Selection favors certain exchanges
  - Some amino acids have similar properties and structure

- Most frequently used: PAM and BLOSUM

Scoring Matrices for Proteins

Amino acids have different biochemical and physical properties, thereby increasing their relative interchangeability over time influenced by evolution

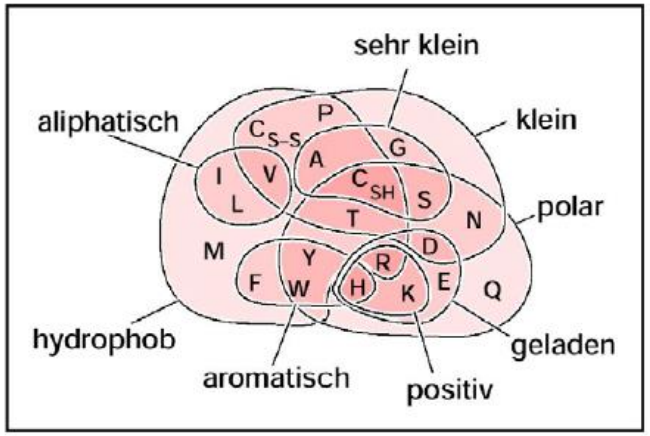


e.g.: score I - L > score von L - Q

How to get this?

How to get reasonable values for scoring matrices?
– We want to measure the biological relevance of the similarity

- Possibilty 1: chemical properties
  – Charge, size, polarity, …
  – Numerous factors with unclear weighting
  – How to express this in a scoring scheme?
  – Not applied in praxis



- Possibility 2: Observation
  – Observation of evolution
  – Measure factual mutations
  – Needs large sets of homologous sequences

PAM / Dayhoff Matrices

- Uses an empirical evolution model
- Designed by Margret Dayhoff and colleagues in the 1970s
- Model of evolution is based on the following assumption:
  - Proteins change during evolution
  - by a sequence of independent point mutations,
  - which are accepted after selection in a population and then observed in the sequences.
- Dayhoff and colleagues introduced the term point accepted mutation as a measure of evolutionary distance:
  - Two sequences A and B differ by 1 PAM unit if B arose from A through a series of accepted point mutations and there was an average of 1 point mutation per 100 residues.
- The aim is to construct a model that describes the probability of accepted mutations with sufficient accuracy → mutation model

Mutation Modell

In the mutation model M we assume that the alignment of S1 and S2 can be exmplained my mutations.
Thus, the probability of an alignment of S1 and S2 is

$$P(S_1, S_2 \mid M) = \prod_{i=1}^{n} p_{x_i} p_{x_i, y_i}$$

where p_{ab} is the probability that amino acid *a* mutates into amino acid *b* ($p_{ab} = p_{ba}$)

Random Model R

- We consider two sequences $S_1 = x_1x_2\ldots x_n$ and S2 = $y_1y_2\ldots y_n$ and their alignment (without gaps, i.e. without insertions and deletions) under two competing models.
- In the random model R, the assumption is that each amino acid a occurs independently with a probability p_a. The probability of an alignment of S1 and S2 in the random model is:

$$P(S_1, S_2 \mid R) = \prod_{i=1}^{n} p_{x_i} \prod_{j=1}^{n} p_{y_j}$$

Model Comparison

We now compare these two models:

$$\frac{P(S_1, S_2 \mid M)}{P(S_1, S_2 \mid R)} = \prod_{i=1}^{n} \frac{p_{x_i} p_{x_i, y_i}}{p_{x_i} p_{y_j}}$$

If this value is greater than 1, then the mutation model describes the alignment better than the random model; otherwise it is vice versa. Taking the logarithm gives an additive measure:

$$score(S_1, S_2) = \sum_{i=1}^{n} score(x_i, y_i) = \sum_{i=1}^{n} \log \frac{p_{x_i} p_{x_i, y_i}}{p_{x_i} p_{y_j}}$$

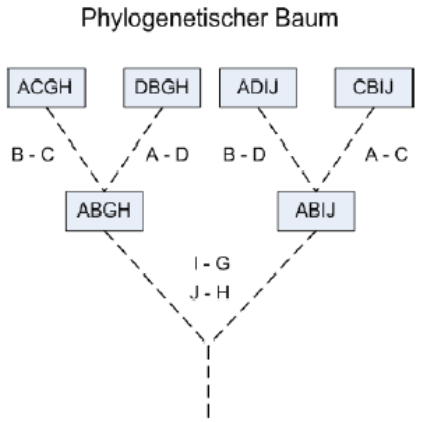
$$\text{also } score(a, b) = \log \frac{p_{a,b}}{p_b}$$

Design of the PAM1 Matrix

- Step 1: Dayhoff used alignments of sequences that were at least 85 percent identical. 34 protein families with 71 phylogenetic trees and 1572 replacements were used

- Step 2: Construct phylogenetic trees and infer lineage sequences

- Step 3: Construct a replacement matrix A by counting the replacements in all pairwise comparisons. Each A_{ij} represents the number of times amino acid *j* was replaced by amino acid *i* in all comparisons



	A	B	C	D	G	H	I	J
A			1	1				
B			1	1				
C	1	1						
D	1	1						
G							1	
H								1
I					1			
J						1		

Accepted Mutations, Transition Matrix

- A list of **accepted mutations** is obtained by identifying all known mutations of the form "amino acid i mutated into amino acid j" in (in characters: (i->j)) sequences that are already known to be "related".

- It should be noted here that only undirected ((i.e. (i->j)= (j->i)) and direct mutations can be considered.

- It is also determined how often an amino acid j occurs, which is denoted by the probability p_j.

- Probabilities sum up to 1: $$\sum p_j = 1$$

Replacement Matrix and Relative Frequencies (Example)

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A																				
R	30																			
N	109	17																		
D	154	0	532																	
C	33	10	0	0																
Q	93	120	50	76	0															
E	266	0	94	831	0	422														
G	579	10	156	162	10	30	112													
H	21	103	226	43	10	243	23	10												
I	66	30	36	13	17	8	35	0	3											
L	95	17	37	0	0	75	15	17	40	253										
K	57	477	322	85	0	147	104	60	23	43	39									
M	29	17	0	0	0	20	7	7	0	57	207	90								
F	20	7	7	0	0	0	0	17	20	90	167	0	17							
P	345	67	27	10	10	93	40	49	50	7	43	43	4	7						
S	772	137	432	98	117	47	86	450	26	20	32	168	20	40	269					
T	590	20	169	57	10	37	31	50	14	129	52	200	28	10	73	696				
W	0	27	3	0	0	0	0	0	3	0	13	0	0	10	0	17	0			
Y	20	3	36	0	30	0	10	0	40	13	23	10	0	260	0	22	23	6		
V	365	20	13	17	33	27	37	97	30	661	303	17	77	10	50	43	186	0	17	

15720 replacements

Gly	0.089	Val	0.065	Arg	0.041	His	0.034
Ala	0.087	Thr	0.058	Asn	0.040	Cys	0.033
Leu	0.085	Pro	0.051	Phe	0.040	Tyr	0.030
Lys	0.081	Glu	0.050	Gln	0.038	Met	0.015
Ser	0.07o	Asp	0.047	Ile	0.037	Trp	0.010

relative frequencies of all AAs in all sequences (initial probabilities)

Replacement Matrix

List of accepted mutations $L = [(a_1, b_1), \ldots, (a_n, b_n)]$
(accepted point mutations)

$p_{a,b}$ is the probability that b is in a sequence where previously there was an a:

$$p_{a,b} = P(b \mid a) = \frac{P(a, b)}{P(a)}$$

$n_{a,b}$:= number of pairs (a, b) in L
n := number of all pairs in L

$P(a, b) \approx \frac{n_{a,b}}{n} =$ relative probability of mutation of a into b

Transition Matrix

In total we get:

$$p_{a,b} = P(b \mid a) = \frac{P(a,b)}{P(a)} \approx \frac{n_{a,b}}{n \cdot p_a}$$

We define that 1 PAM (percent accepted mutations) is that time period in which 1% of the amino acids mutate:

$$p_{a,b} := \frac{n_{a,b}}{100 \cdot n \cdot p_a}$$

$$p_{a,a} := 1 - \sum_{b \neq a} p_{a,b}$$

Transition Matrix

$$
\begin{aligned}
\sum_a p_a \cdot p_{a,a} &= \sum_a p_a \left(1 - \sum_{b \neq a} p_{a,b}\right) \\
&= \sum_a p_a - \sum_a \sum_{b \neq a} p_a \cdot p_{a,b} \\
&= 1 - \sum_a \sum_{b \neq a} p_a \cdot \frac{n_{a,b}}{100n \cdot p_a} \\
&= 1 - \frac{1}{100n} \cdot \sum_a \sum_{b \neq a} n_{a,b} \\
&= 1 - \frac{1}{100n} \cdot n \ = 0,99
\end{aligned}
$$

probability that there is no mutation is 0.99

1 – PAM Transition Matrix (Example)

Variable Editor - pab

Stack: PAMexample | Select data to plot

pab <20x20 double>

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.9600	2.1936e-04	7.9699e-04	0.0011	2.4129e-04	6.8000e-04	0.0019	0.0042	1.5355e-04	4.8258e-04	6.9463e-04	4.1678e-04	2.1204e-04	1.4624e-04	0.0025	0.0056	0.0043	1.0000e-12	1.4624e-04	0.0027
2	4.6546e-04	0.9858	2.6376e-04	1.0000e-12	1.5515e-04	0.0019	1.0000e-12	1.5515e-04	0.0016	4.6546e-04	2.6376e-04	0.0074	2.6376e-04	1.0861e-04	0.0010	0.0021	3.1031e-04	4.1892e-04	4.6546e-05	3.1031e-04
3	0.0017	2.7036e-04	0.9736	0.0085	1.0000e-12	7.9517e-04	0.0015	0.0025	0.0036	5.7252e-04	5.8842e-04	0.0051	1.0000e-12	1.1132e-04	4.2939e-04	0.0069	0.0027	4.7710e-05	5.7252e-04	2.0674e-04
4	0.0021	1.0000e-12	0.0072	0.9739	1.0000e-12	0.0010	0.0112	0.0022	5.8199e-04	1.7595e-04	1.0000e-12	0.0012	1.0000e-12	1.0000e-12	1.3535e-04	0.0013	7.7148e-04	1.0000e-12	1.0000e-12	2.3009e-04
5	6.3613e-04	1.9277e-04	1.0000e-12	1.0000e-12	0.9968	1.0000e-12	1.0000e-12	1.9277e-04	1.9277e-04	3.2770e-04	1.0000e-12	1.0000e-12	1.0000e-12	1.0000e-12	1.9277e-04	0.0023	1.9277e-04	1.0000e-12	5.7830e-04	6.3613e-04
6	0.0016	0.0020	8.3702e-04	0.0013	1.0000e-12	0.9805	0.0071	5.0221e-04	0.0041	1.3392e-04	0.0013	0.0025	3.3481e-04	1.0000e-12	0.0016	7.8680e-04	6.1939e-04	1.0000e-12	1.0000e-12	4.5199e-04
7	0.0034	1.0000e-12	0.0012	0.0106	1.0000e-12	0.0054	0.9730	0.0014	2.9262e-04	4.4529e-04	1.9084e-04	0.0013	8.9059e-05	1.0000e-12	5.0891e-04	0.0011	3.9440e-04	1.0000e-12	1.2723e-04	4.7074e-04
8	0.0041	7.1476e-05	0.0011	0.0012	7.1476e-05	2.1443e-04	8.0053e-04	0.9813	7.1476e-05	1.0000e-12	1.2151e-04	4.2885e-04	5.0033e-05	1.2151e-04	3.5023e-04	0.0032	3.5738e-04	1.0000e-12	1.0000e-12	6.9331e-04
9	3.9291e-04	0.0019	0.0042	8.0452e-04	1.8710e-04	0.0045	4.3032e-04	1.8710e-04	0.9863	5.6129e-05	7.4839e-04	4.3032e-04	1.0000e-12	3.7420e-04	9.3549e-04	4.8645e-04	2.6194e-04	5.6129e-05	7.4839e-04	5.6129e-04
10	0.0011	5.1578e-04	6.1894e-04	2.2351e-04	2.9228e-04	1.3754e-04	6.0175e-04	1.0000e-12	5.1578e-05	0.9828	0.0043	7.3929e-04	9.7999e-04	0.0015	1.2035e-04	3.4386e-04	0.0022	1.0000e-12	2.2351e-04	0.0114
11	7.1097e-04	1.2723e-04	2.7690e-04	1.0000e-12	1.0000e-12	5.6129e-04	1.1226e-04	1.2723e-04	2.9936e-04	0.0019	0.9744	2.9187e-04	0.0015	0.0012	3.2181e-04	2.3949e-04	3.8916e-04	9.7291e-05	1.7213e-04	0.0023
12	4.4765e-04	0.0037	0.0025	6.6755e-04	1.0000e-12	0.0012	8.1676e-04	4.7121e-04	1.8063e-04	3.3770e-04	3.0629e-04	0.9718	7.0681e-04	1.0000e-12	3.3770e-04	0.0013	0.0016	1.0000e-12	7.8535e-05	1.3351e-04
13	0.0012	7.2095e-04	1.0000e-12	1.0000e-12	1.0000e-12	8.4818e-04	2.9686e-04	2.9686e-04	1.0000e-12	0.0024	0.0088	0.0038	0.9943	7.2095e-04	1.6964e-04	8.4818e-04	0.0012	1.0000e-12	1.0000e-12	0.0033
14	3.1807e-04	1.1132e-04	1.1132e-04	1.0000e-12	1.0000e-12	1.0000e-12	1.0000e-12	2.7036e-04	3.1807e-04	0.0014	0.0027	1.0000e-12	2.7036e-04	0.9888	1.1132e-04	6.3613e-04	1.5903e-04	1.5903e-04	0.0041	1.5903e-04
15	0.0043	8.3570e-04	3.3678e-04	1.2473e-04	1.2473e-04	0.0012	4.9893e-04	6.1119e-04	6.2366e-04	8.7312e-05	5.3635e-04	5.3635e-04	4.9893e-05	8.7312e-05	0.9875	0.0034	9.1054e-04	1.0000e-12	1.0000e-12	6.2366e-04
16	0.0070	0.0012	0.0039	8.9059e-04	0.0011	4.2712e-04	7.8153e-04	0.0041	2.3628e-04	1.8175e-04	2.9080e-04	0.0015	1.8175e-04	3.6350e-04	0.0024	0.9598	0.0063	1.5449e-04	1.9993e-04	3.9077e-04
17	0.0065	2.1936e-04	0.0019	6.2516e-04	1.0968e-04	4.0581e-04	3.4000e-04	5.4839e-04	1.5355e-04	0.0014	5.7033e-04	0.0022	3.0710e-04	1.0968e-04	8.0065e-04	0.0076	0.9750	1.0000e-12	2.5226e-04	0.0020
18	1.0000e-12	0.0017	1.9084e-04	1.0000e-12	1.0000e-12	1.0000e-12	1.0000e-12	1.0000e-12	1.9084e-04	1.0000e-12	8.2697e-04	1.0000e-12	1.0000e-12	6.3613e-04	1.0000e-12	0.0011	1.0000e-12	0.9989	3.8168e-04	1.0000e-12
19	4.2409e-04	6.3613e-05	7.6336e-04	1.0000e-12	6.3613e-04	1.0000e-12	2.1204e-04	1.0000e-12	8.4818e-04	2.7566e-04	4.8770e-04	2.1204e-04	1.0000e-12	0.0055	1.0000e-12	4.6650e-04	4.8770e-04	1.2723e-04	0.9922	3.6047e-04
20	0.0036	1.9573e-04	1.2723e-04	1.6637e-04	3.2296e-04	2.6424e-04	3.6211e-04	9.4931e-04	2.9360e-04	0.0065	0.0030	1.6637e-04	7.5357e-04	9.7867e-05	4.8933e-04	4.2083e-04	0.0018	1.0000e-12	1.6637e-04	0.9732

Score Matrix PAM1

PAM1 scoring matrix:

$$score(a, b) = \frac{p_{a,b}}{p_b} = \frac{\frac{n_{a,b}}{100n \cdot p_a}}{p_b} = \frac{n_{a,b}}{100n \cdot p_a \cdot p_b}$$

In fact we apply the log10 and each entry is multiplied with factor 10 (for better readability)

PAM-n Matrices

How to PAM score matrices that represent evolutinoary distances > PAM1?

$$P^{(n)} = \underbrace{P \times P \times \ldots\ldots \times P}_{\text{n -mal}}$$

PAM- 250:

2.5 mutations per residue, ca. 20% hits (accordance), thus changes in 80% of the positions

PAM:	1	30	80	110	200	250
Sequance similarity (%):	99	75	50	60	25	20

PAM 250 – log-odds Matrix

C

W

W

-8

17

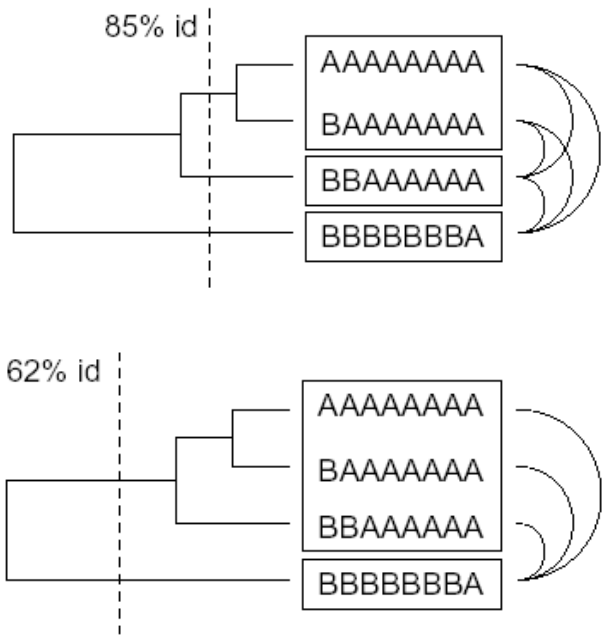
	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	B	Z
A	2	-2	0	0	-2	0	0	1	-1	-1	-2	-1	-1	-3	1	1	1	-6	-3	0	2	1
R	-2	6	0	-1	-4	1	-1	-3	2	-2	-3	3	0	-4	0	0	-1	2	-4	-2	1	2
N	0	0	2	2	-4	1	1	0	2	-2	-3	1	-2	-3	0	1	0	-4	-2	-2	4	3
D	0	-1	2	4	-5	2	3	1	1	-2	-4	0	-3	-6	-1	0	0	-7	-4	-2	5	4
C	-2	-4	-4	-5	12	-5	-5	-3	-3	-2	-6	-5	-5	-4	-3	0	-2	-8	0	-2	-3	-4
Q	0	1	1	2	-5	4	2	-1	3	-2	-2	1	-1	-5	0	-1	-1	-5	-4	-2	3	5
E	0	-1	1	3	-5	2	4	0	1	-2	-3	0	-2	-5	-1	0	0	-7	-4	-2	4	5
G	1	-3	0	1	-3	-1	0	5	-2	-3	-4	-2	-3	-5	0	1	0	-7	-5	-1	2	1
H	-1	2	2	1	-3	3	1	-2	6	-2	-2	0	-2	-2	0	-1	-1	-3	0	-2	3	3
I	-1	-2	-2	-2	-2	-2	-2	-3	-2	5	2	-2	2	1	-2	-1	0	-5	-1	4	-1	-1
L	-2	-3	-3	-4	-6	-2	-3	-4	-2	2	6	-3	4	2	-3	-3	-2	-2	-1	2	-2	-1
K	-1	3	1	0	-5	1	0	-2	0	-2	-3	5	0	-5	-1	0	0	-3	-4	-2	2	2
M	-1	0	-2	-3	-5	-1	-2	-3	-2	2	4	0	6	0	-2	-2	-1	-4	-2	2	-1	0
F	-3	-4	-3	-6	-4	-5	-5	-5	-2	1	2	-5	0	9	-5	-3	-3	0	7	-1	-3	-4
P	1	0	0	-1	-3	0	-1	0	0	-2	-3	-1	-2	-5	6	1	0	-6	-5	-1	1	1
S	1	0	1	0	0	-1	0	1	-1	-1	-3	0	-2	-3	1	2	1	-2	-3	-1	2	1
T	1	-1	0	0	[illegible]	[illegible]	0	0	-1	0	-2	0	-1	-3	0	1	3	[illegible]	[illegible]	0	2	1
W	-6	2	-4	-7	[illegible]	[illegible]	-7	-7	-3	-5	-2	-3	-4	0	-6	-2	-5	[illegible]	[illegible]	-6	-4	-4
Y	-3	-4	-2	-4	[illegible]	[illegible]	-4	-5	0	-1	-1	-4	-2	7	-5	-3	-3	[illegible]	[illegible]	-2	-2	-3
V	0	-2	-2	-2	-2	-2	-2	-1	-2	4	2	-2	2	-1	-1	-1	0	-6	-2	4	0	0
B	2	1	4	5	-3	3	4	2	3	-1	-2	2	-1	-3	1	2	2	-4	-2	0	6	5
Z	1	2	3	4	-4	5	5	1	3	-1	-1	2	0	-4	1	1	1	-4	-3	0	5	6

BLOSUMS
(Blocks Substitution Matrices)

- Similar ideas
- Count known mutations in sequences

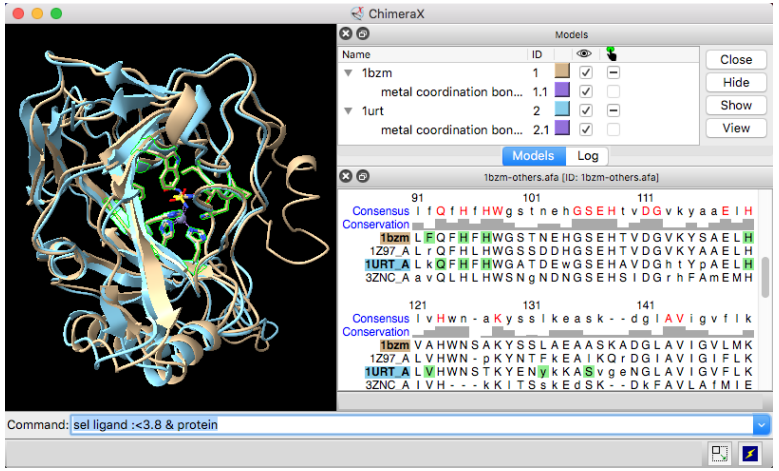
- Use trees to cluster sequences
- Cutoff threshold defines number of BLOSUM (BLOSUM60, …)

BLOSUM 80	BLOSUM 62	BLOSUM 45
PAM 1	PAM 120	PAM 250
Less divergent ←		→ *More divergent*

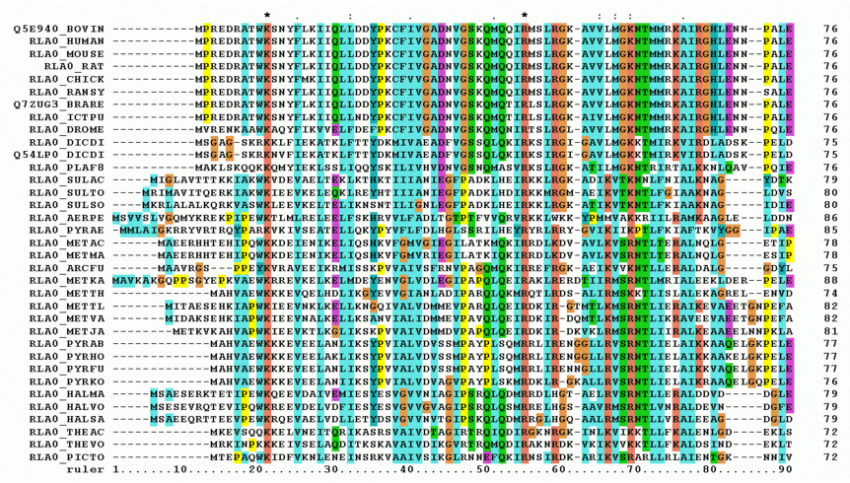


And again, now for something completely different…

- Now we know how to score pairs / alignments of sequences



https://www.rbvi.ucsf.edu/chimerax/



Wikipedia

- But: How do we get these alignments?

Score of Insertions and Deletions (Indels, Gaps)

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

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

Insertion / Deletion

Insertion of a gap: negative score

How to set the costs of a gap?

1. Gaps too expensive => "meager" result

```
1 GTGATAGACACAGACCGGTGGCATTGTGGA 29
  |||    |  |            |  ||
1 GTGTCGGGAAGCAGATAACTCCGATGGTTG 29
```

2. Gaps too cheap => stretched alignment, unspecific

```
1 GTG.ATAG.ACA.CAGA..CCGGT..GGCCTTACGG 29
  ||| || | | | ||||  ||  |  |   |    |
1 GTGTAT.GGA.AGCAGATACC..TCCG...T....G 29
```

Affine Gap Costs

Score with linear costs:

$$\gamma(g) = -gd$$

Score with affine costs:

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

$\gamma(g)$ = costs for gap with length g

d = costs for opening a gap

e = costs for prolongiation of a gap

g = gap length

Affine Gap Costs

Match = 1
Mismatch = 0

Total score: 4

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

Total score: 8 - 3.2 = 4.8

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

insertion / deletion

<u>Gap parameters:</u>

$d = 3$ (gap opening)
$e = 0.1$ (gap extension)
$g = 3$ (gap length)

$\gamma(g) = -3 - (3 - 1)\ 0.1 = -3.2$

Remarks

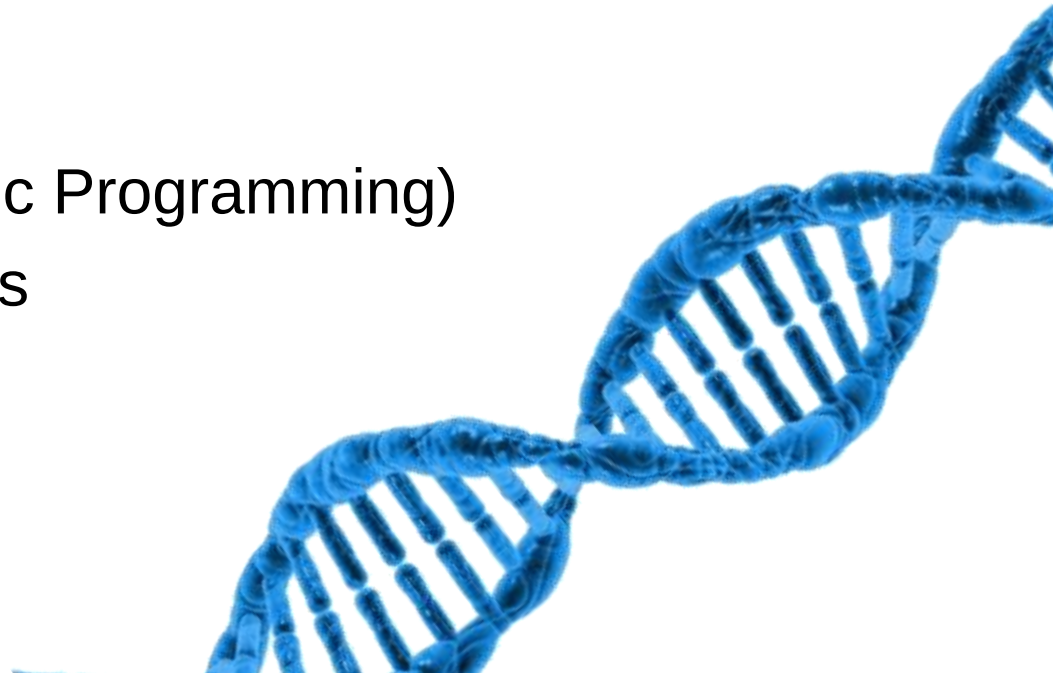
- Proper gap assessment along with the correct score matrix is crucial, especially for more difficult alignment problems (e.g. sequences with little similarity)
- As a rule, suitable gap evaluations are calculated for the score matrices and can be used by default
- “Overhanging” start and end sequences are usually not penalized
- Low gap costs can turn a local alignment into a global one, the alignment is stretched

(3) Alignments

Dotplot

Global/Local Alignment (Dynamic Programming)

Heuristic Methods



Methods of Sequence Alignments

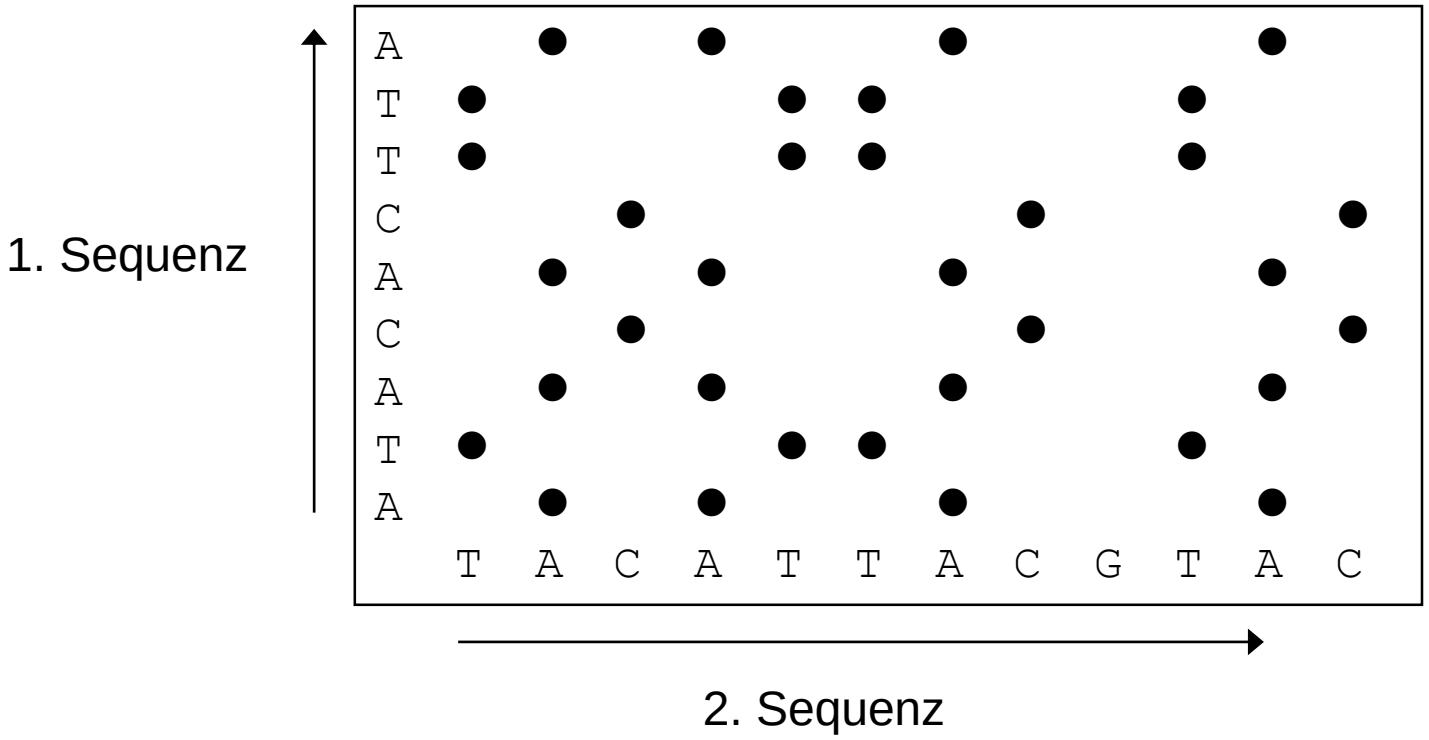
- dotplot analysis
  - first impression,
  - shows insertions/omissions, repeats

- Dynamic programming
  - gives optimal alignment,
  - all possible combinations,
  - cpu intensive

- Word methods
  - Collect "islands“
  - fast
  - Heuristic
  - used for DB searches (special slides desk)

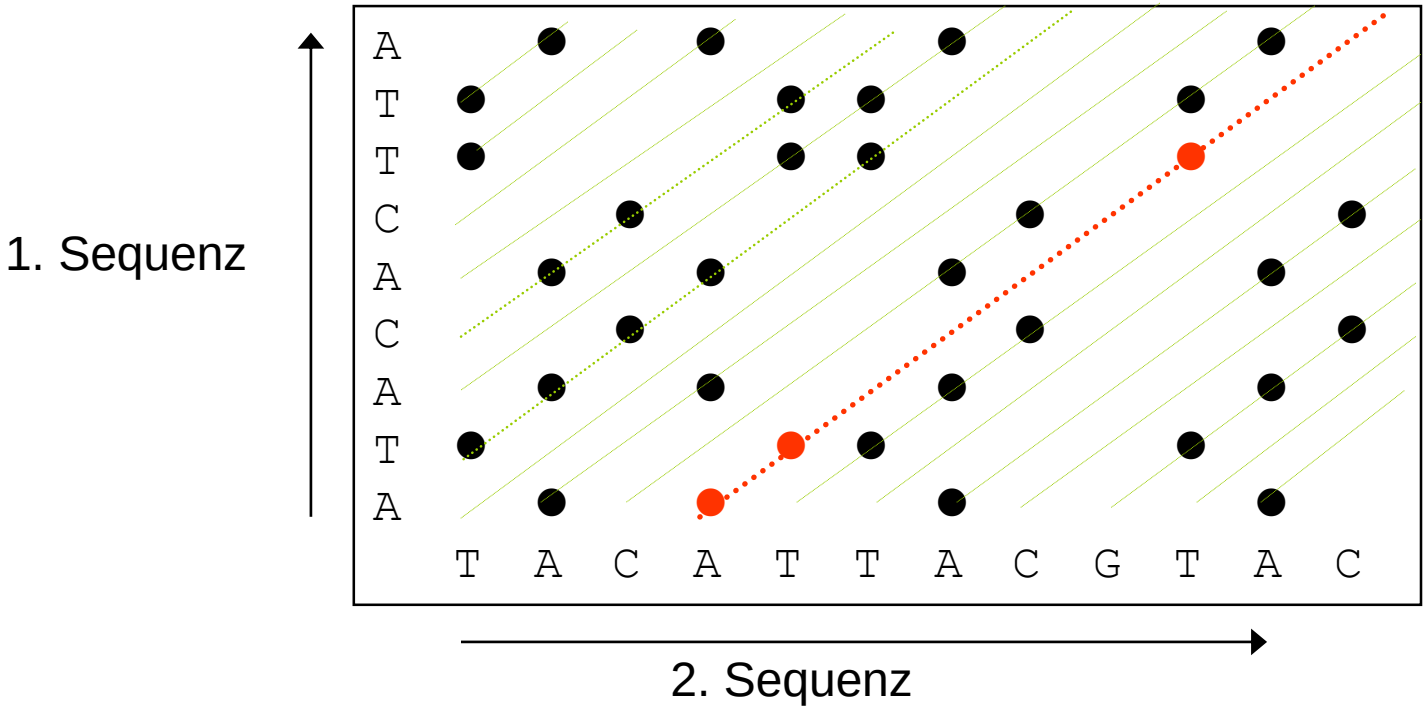
Dotplots

Overview of all possible alignments



Dotplot -2

Each diagonal corresponds to a possible alignment



Possible alignment:

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

Chars -> Words
Dotplots with Windowing

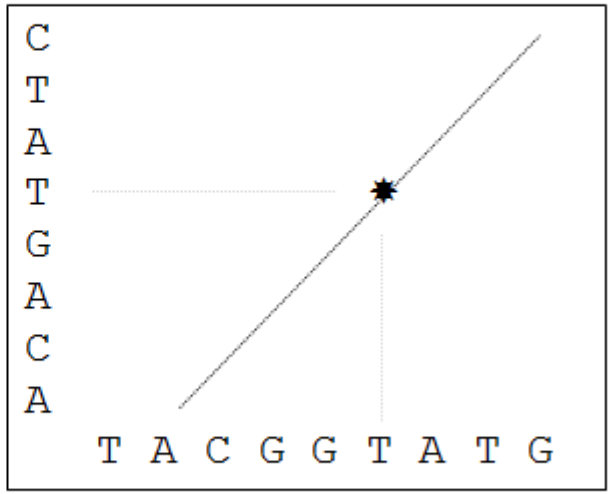
```
T A C G G T A T G
  | | | | | |
  A C A G T A T C

T A C G G T A T G
  | | | | | |
  A C A G T A T C

T A C G G T A T G
  | | | | | |
  A C A G T A T C

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

Wortgröße = 3 = Fenstergröße



Windows / Stringency

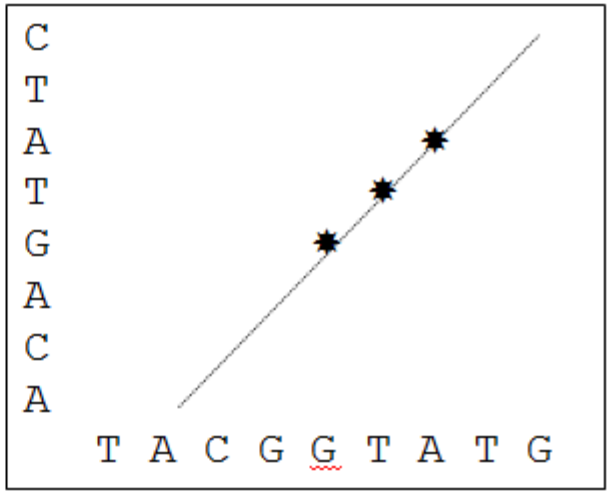
```
T A C G G T A T G
  |   | | | |
  T C A G T A T C

T A C G G T A T G
  |   | | | |
  T C A G T A T C

T A C G G T A T G
  |   | | | |
  T C A G T A T C

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

Fenster = 5 / Stringenz = 4



Windows / Stringency

... also for amino acids…

Score = 11

```
PTHPLASKTQILPEDLASEDLTI
|||||||     |     || ||     ✸
PTHPLAGERAIGLARLAEEDFGM
                    →
```

Score = 11

```
PTHPLASKTQILPEDLASEDLTI
||||||     |     || ||     ✸
PTHPLAGERAIGLARLAEEDFGM
                    →
```

Score = 7

```
PTHPLASKTQILPEDLASEDLTI
|||||     |     || ||
PTHPLAGERAIGLARLAEEDFGM
                    →
```

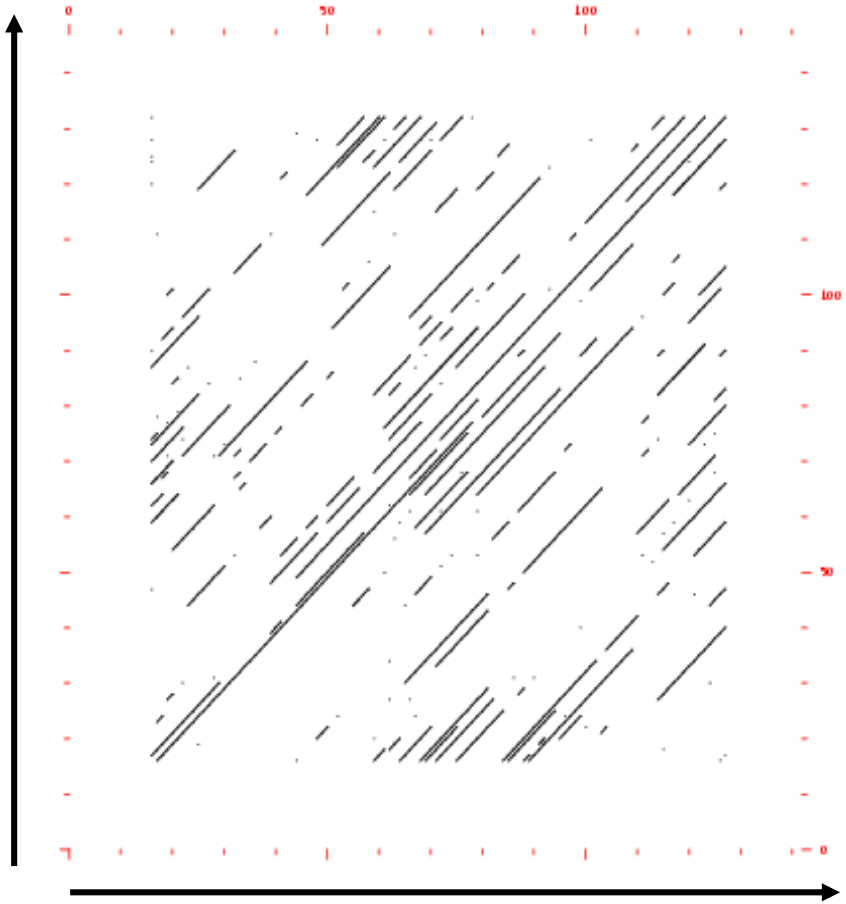
Matrix: PAM250

window size: 12

stringency: 9

Example Hemoglobin, Window Size 30 / Stringency = 9

Hemoglobin β-chain



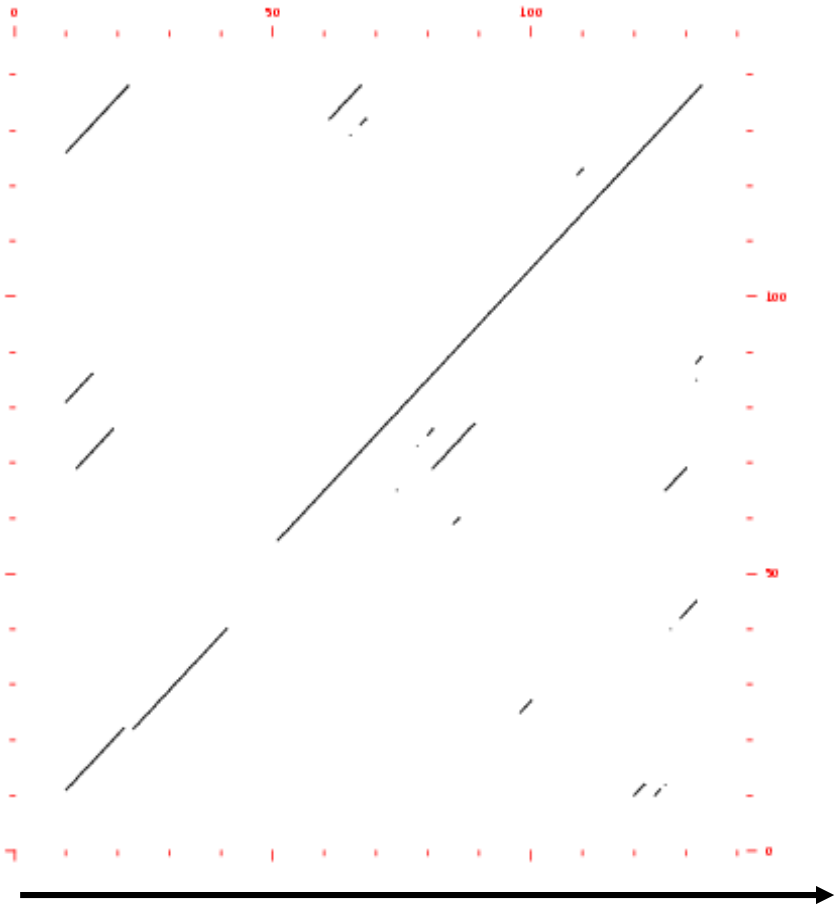
Hemoglobin α-chain

... Stringency to low ...

Graphics from *Compare* und *DotPlot2*

Example Hemoglobin, Window Size 18 / Stringency = 10

Hemoglobin β-chain



Hemoglobin α-chain

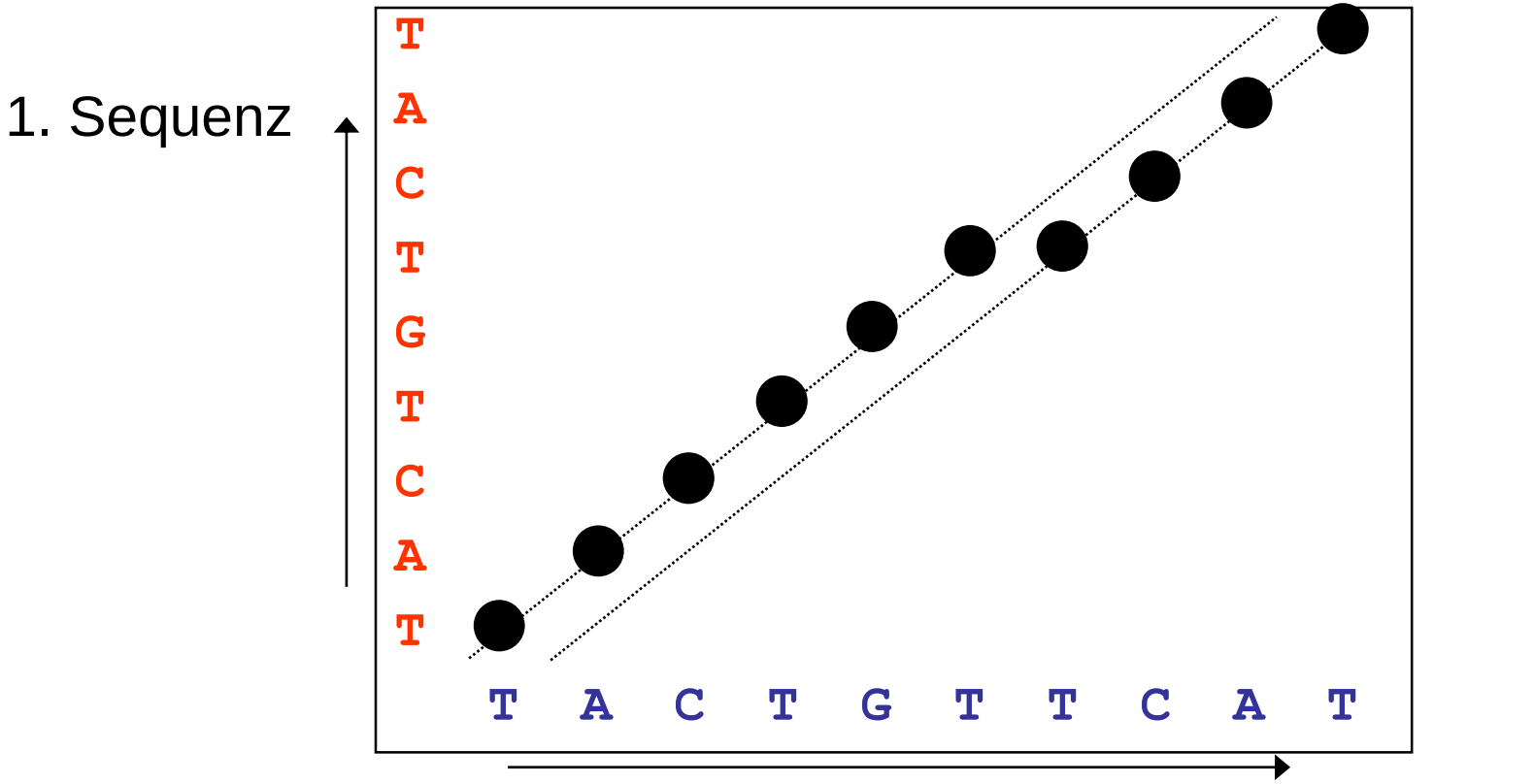
... stringency ok…

Graphics from *Compare* und *DotPlot2*

Comments

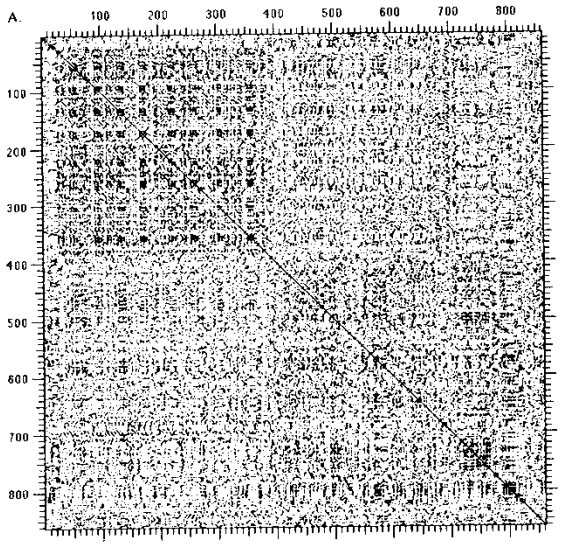
- Window/stringency method is more sensitive than pure word method
- the smaller the window, the more statistical weight mutations have => statistical fluctuations are also displayed
- but: large windows reduce the sensitivity for small sequences
- => optimal window stringency setting must be determined (by trial and error).
- Insertions, omissions are not displayed directly, but indirectly:

Indels in a Dotplot

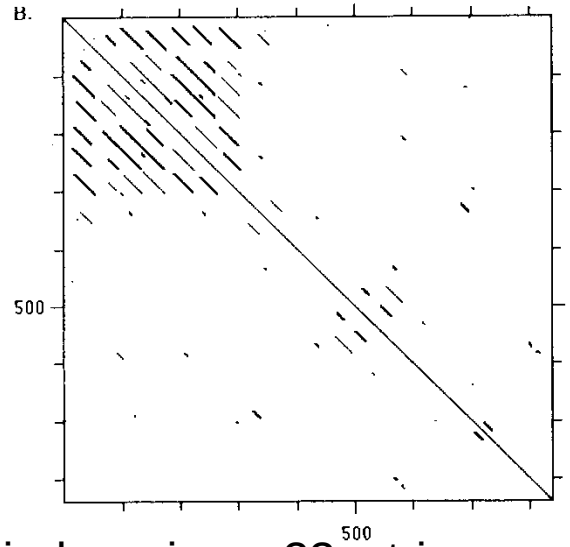



```
T A C T G - T C A T
| | | | |   | | | |
T A C T G T T C A T
```

Repeats



window size = 1, stringency = 1



window size = 23, stringency = 7

- sequence mapped onto itself
- plot is symmetrical
- repeats are smaller diagonals
- here: e.g. 6 repeats in the upper left corner
- repeats are not all the same length => contain mutations
- again: only observable with correct choice of window size and stringency

Dynamic Programming

Automated process, algorithm that finds the best alignment (with optimal score) depending on the given parameters (gap costs, score matrix)

- **Needleman - Wunsch Algorithm:**

  Global Alignment

- **Smith - Waterman Algorithm :**

  Local Alignment