Structural Bioinformatics
Lecture 5

Protein sequences: comparison and evolution

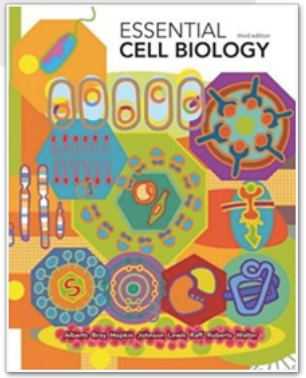
UNIVERSITÄT
DES
SAARLANDES

Outline

- Evolution of proteins
- Sequence alignment
  - Dynamic programming
  - Smith-Waterman and Needleman-Wunsch algorithms
- Sequence similarity search
  - BLAST

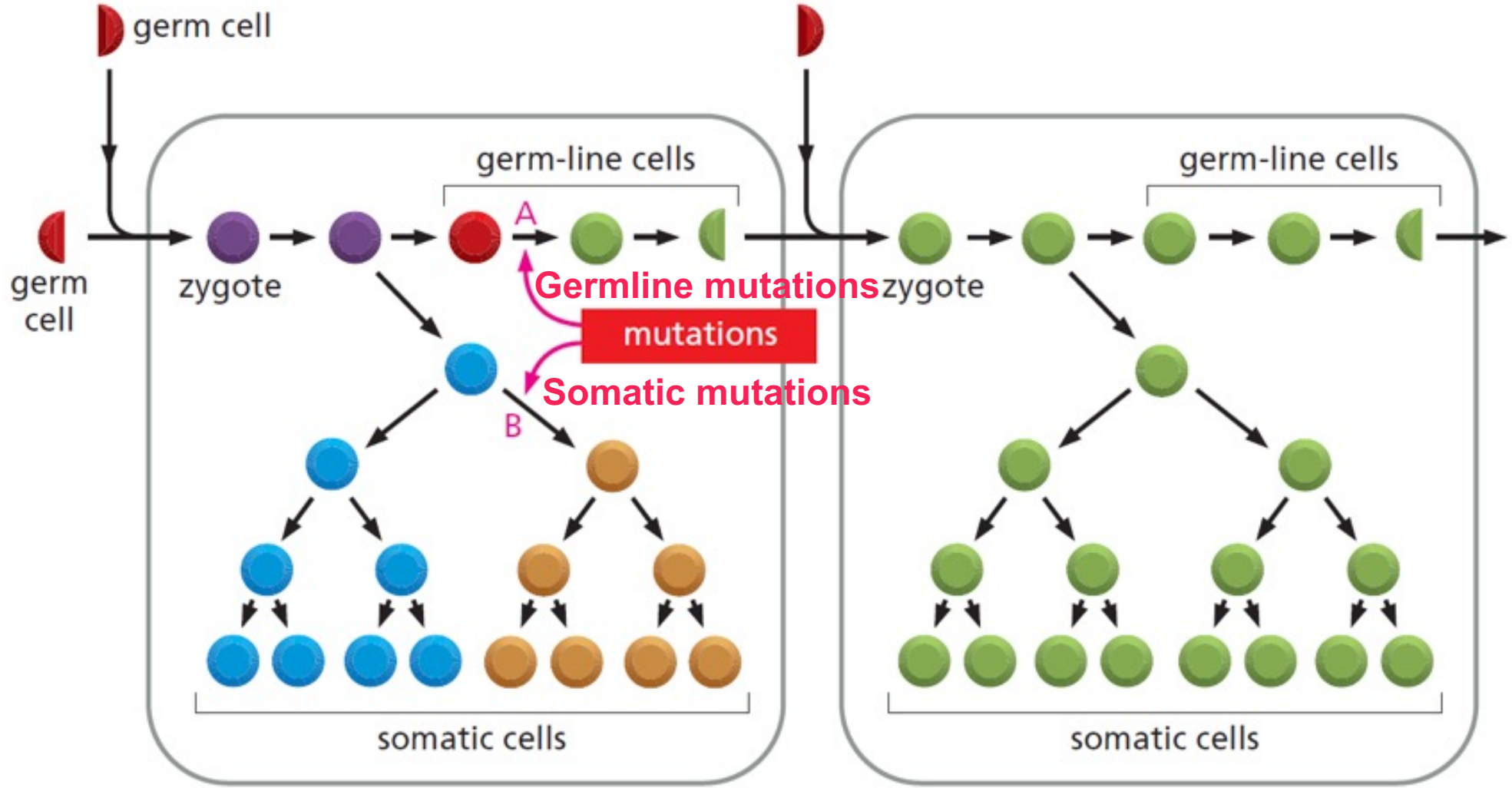
Errors are inevitable

US Postal Service on-time delivery of local first-class mail	13 late deliveries per 100 parcels
Airline luggage system	1 lost bag per 200
A professional typist typing at 120 words per minute	1 mistake per 250 characters
Driving a car in the United States	1 death per 10^4 people per year
DNA replication (without mismatch repair)	1 mistake per 10^7 nucleotides copied
DNA replication (including mismatch repair)	1 mistake per 10^9 nucleotides copied

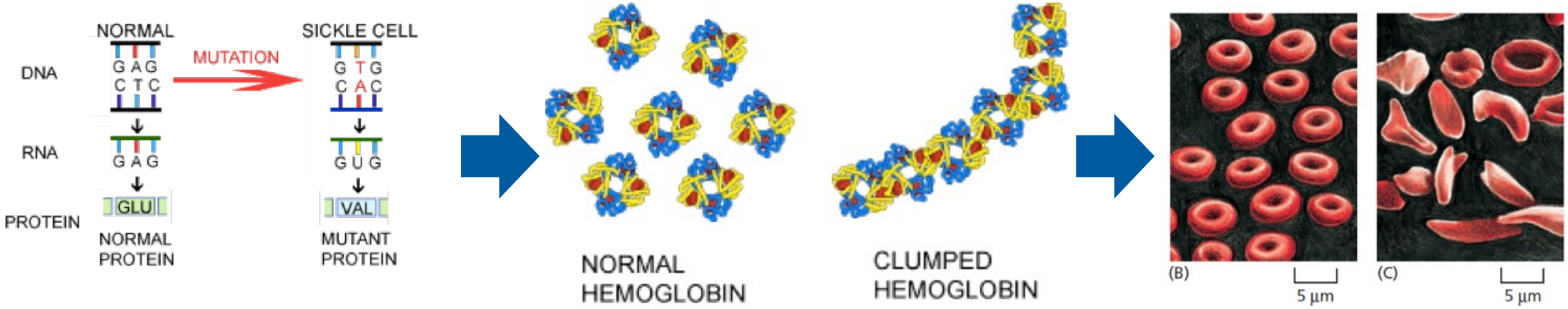


Essential Cell Biology, 3rd edition

Errors of replication: mutations



Individual mutations with high impact: Sickle cell anemia

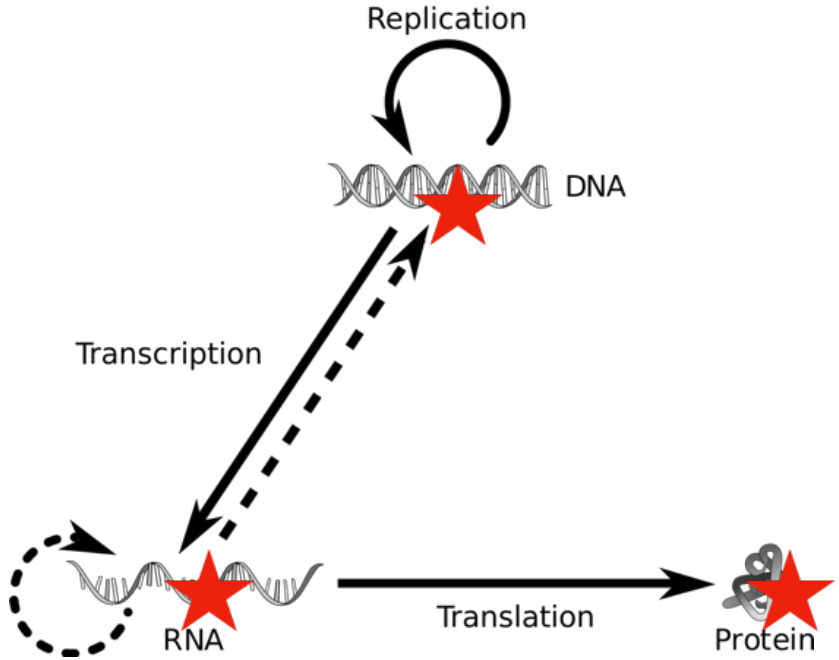


evolution.berkeley.edu

- **Negative effects at the whole organism level**: Pain and fatigue under conditions such as high elevation and intense exercise
- **Positive effects at the whole organism level**: Resistance to malaria: *Plasmodia* cannot live inside sickle-shaped blood cells

Types of mutations

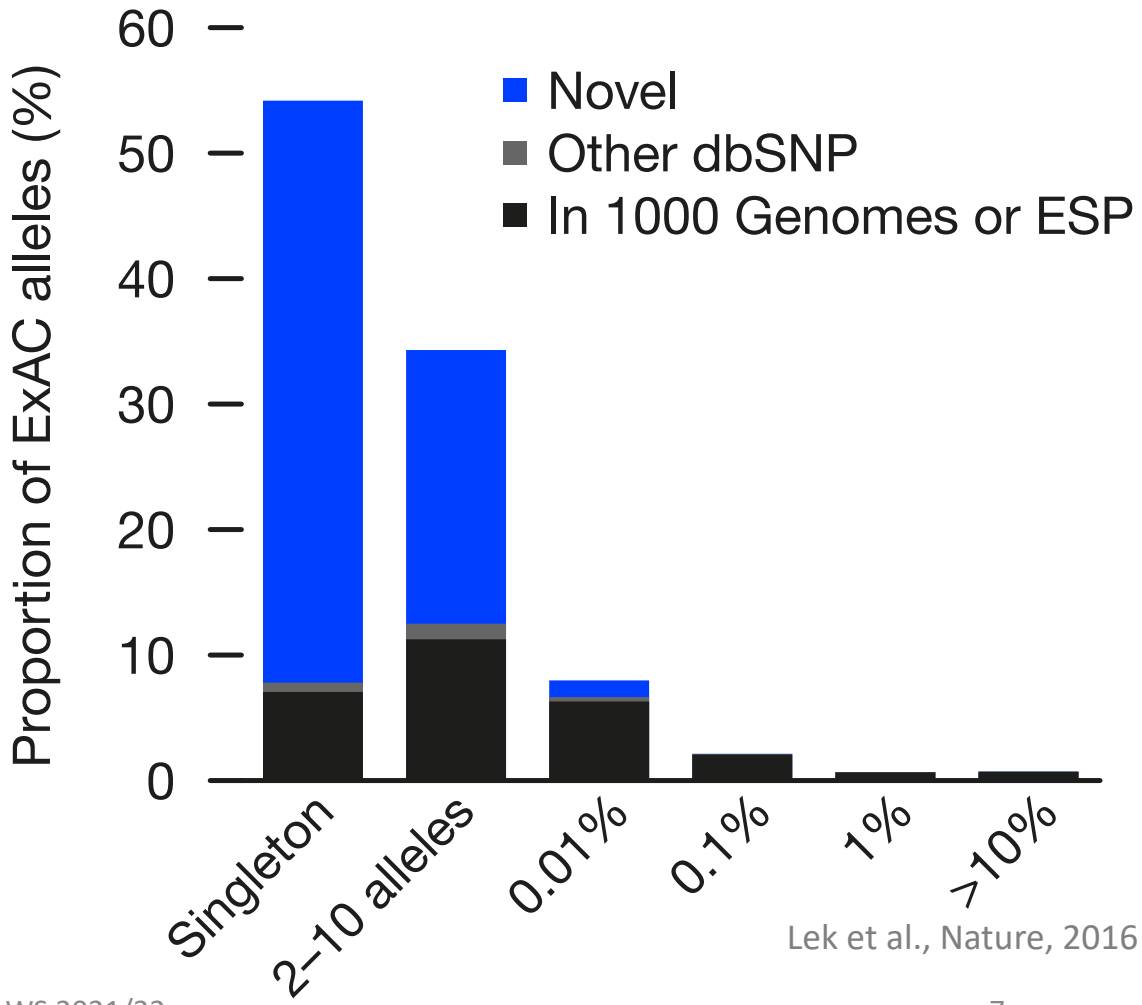
- Single-nucleotide variants (SNVs)
  - Non-coding
  - Coding
  - synonymous/silent
  - protein-truncating variants/ nonsense
  - **non-synonymous/missense**
- Short insertions and deletions/indels
- Genome duplications, structural variants
- Large-scale chromosomal rearrangements



	No mutation	Point mutations			
		Silent	Nonsense	Missense	
				conservative	non-conservative
DNA level	TTC	TTT	ATC	TCC	TGC
mRNA level	AAG	AAA	UAG	AGG	ACG
protein level	Lys	Lys	STOP	Arg	Thr

Mutations in the human genome

- **ExAC (Exome Aggregation Consortium)**: exome sequencing of 60,706 individuals
  - 7,404,909 high-quality sequence variants: **one per every 8 bp**
  - most of them are extremely rare and new (not in the databases)
  - ~12,000 missense and protein-truncating variants per individual
    - 99% of them only in that one individual
- **Most of private variants are unique**



Lek et al., Nature, 2016

Mutations in a single individual

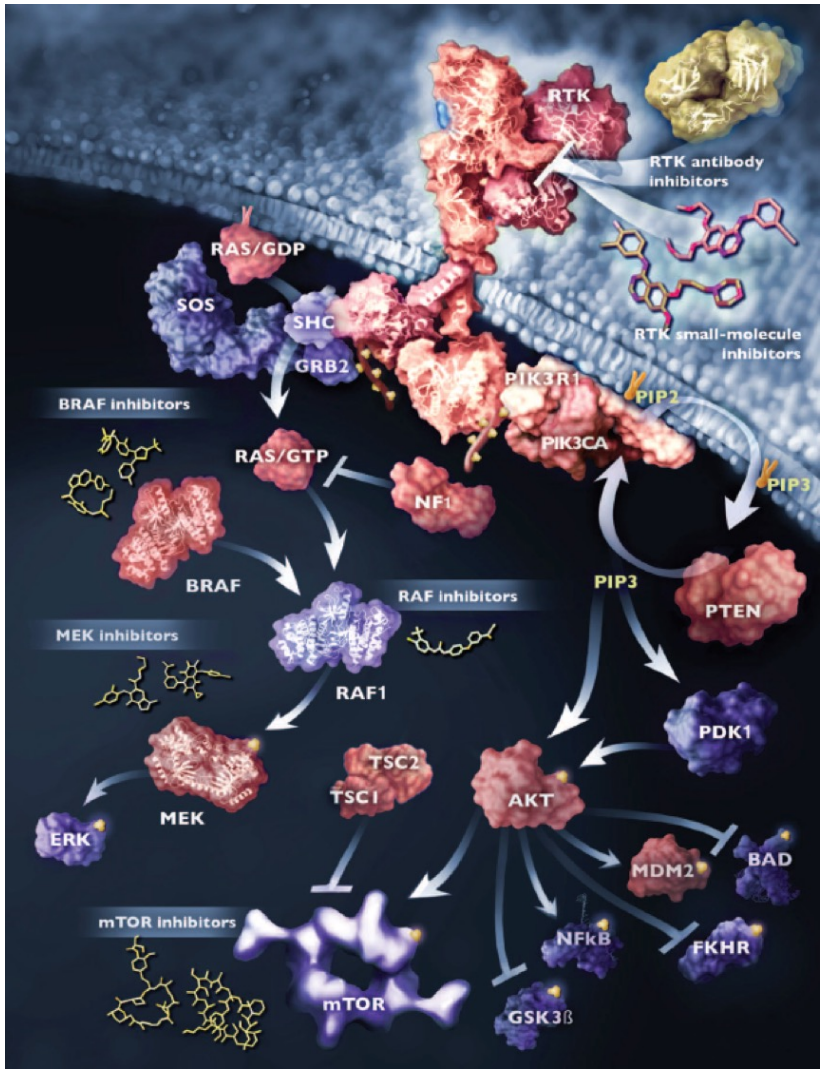
Table 1. Summary and Breakdown of DNA Variants

Type	Total Variants	Total High Confidence
Total SNVs	3,739,701	3,301,521
Total gene-associated SNVs	1,312,780	1,183,847
Total coding/UTR	49,017	44,542
Missense	10,592	9,683
Nonsense	83	73
Synonymous	11,459	10,864
5′UTR	4,085	2,978
3′UTR	22,798	20,944
Intron	1,263,763	1,139,305
Ts/Tv	—	2.14
dbSNP	3,493,748	3,167,180
Candidate private SNV	245,953	134,341
Indels (−107∼ +36 bp)	1,022,901	216,776
Coding	3,263	302
Structural variants (>50 bp)	44,781	2,566
In 1000G project[a]	4,434	1,967

Chen et al., Cell, 2012

Biological effect of mutations

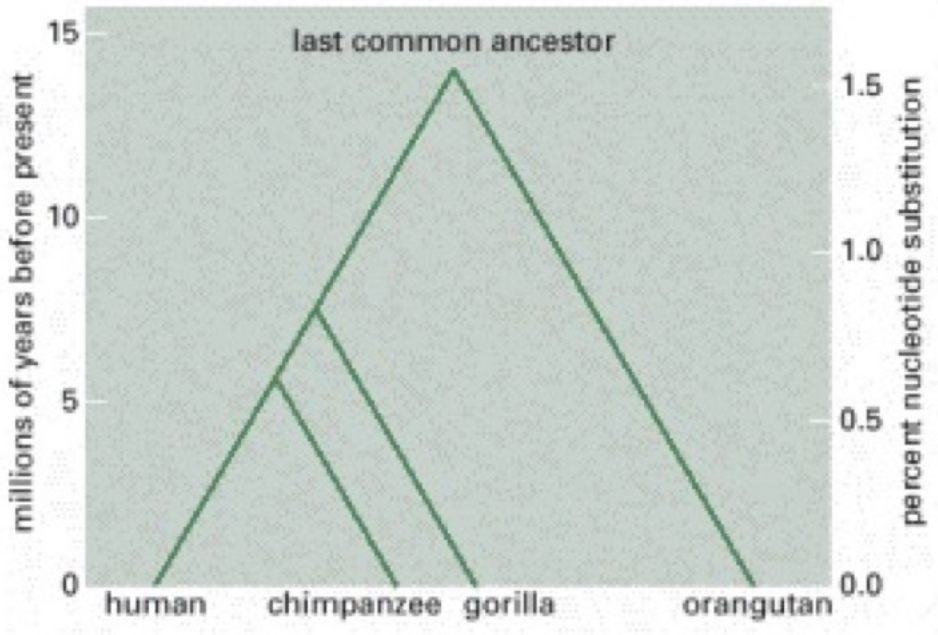
- Sources of mutations:
  - Spontaneous mistakes of DNA polymerase
  - Endogenous DNA damage
  - Exogenous DNA damage
- + repair mechanisms => 1 mutation in $\mathbf{10^{10}}$ nucleotides per cell division (cf. human genome size: $\mathbf{3 \times 10^9}$ bp)
- Mutations in coding regions affect protein sequence, **structure** and **interactions**



RAS and PI3K: pathways related to cell survival, often mutated in cancer (Vogelstein et al., Science, 2013)

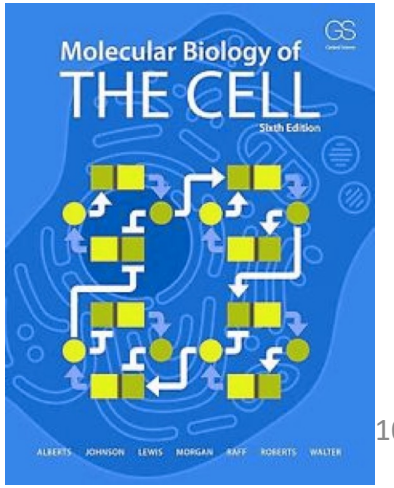
Evolution of genomes

- Genomes of closely related organisms are more similar than of distant ones
  - => sequence comparison can be used to infer evolution
- Similar function => similar sequence
- => protein function can be analyzed via sequence comparison



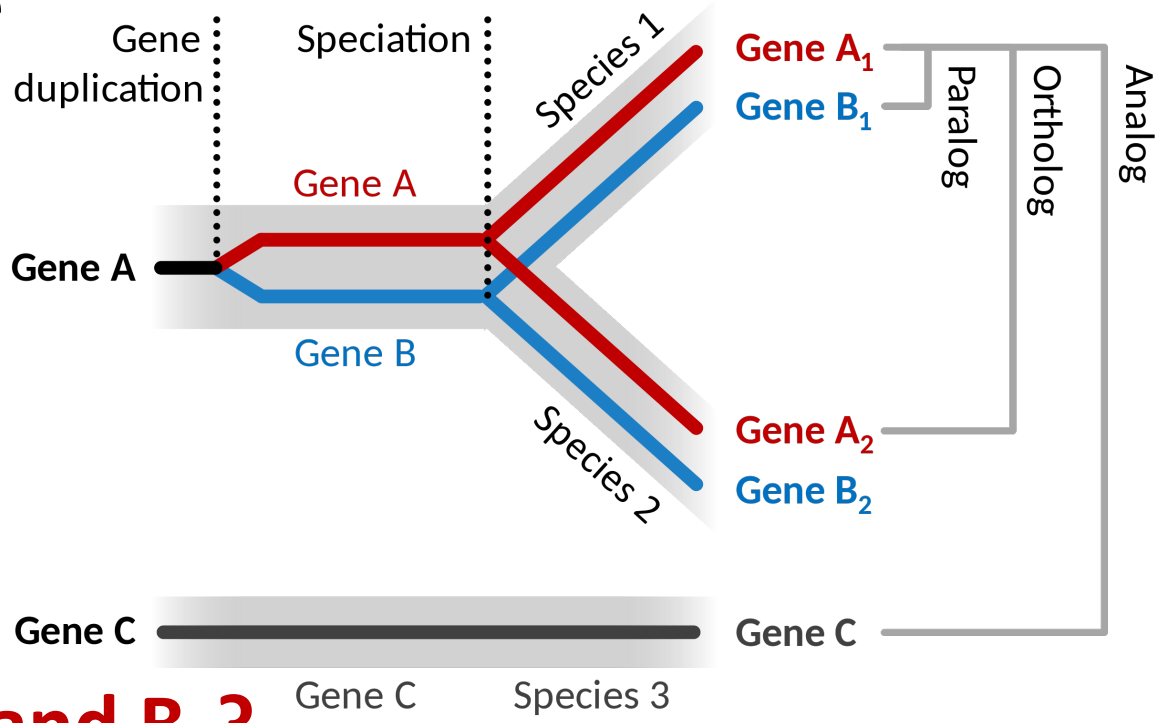
Alberts et al., "Molecular Biology of the Cell"

https://www.ncbi.nlm.nih.gov/books/NBK21054/ =>



Important concepts in sequence evolution

- **Homology**: shared ancestry in the evolutionary history of life
- **Orthologs**: descended from the same ancestral sequence separated by a *speciation* event
- **Paralogs**: related via *duplication* events in the last common ancestor
- **Analogs**: same/similar function, no(?) evolutionary relatedness



https://en.wikipedia.org/wiki/Sequence_homology

Trick question: what are genes A_1 and B_2?

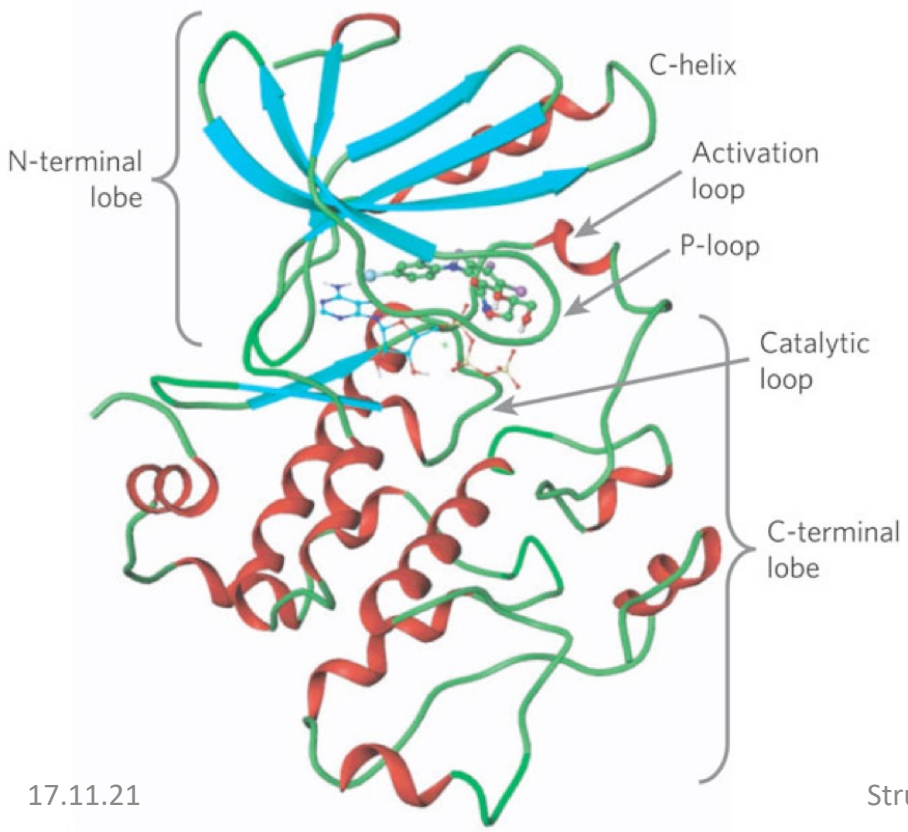
Evolution in real time: antibiotics resistance

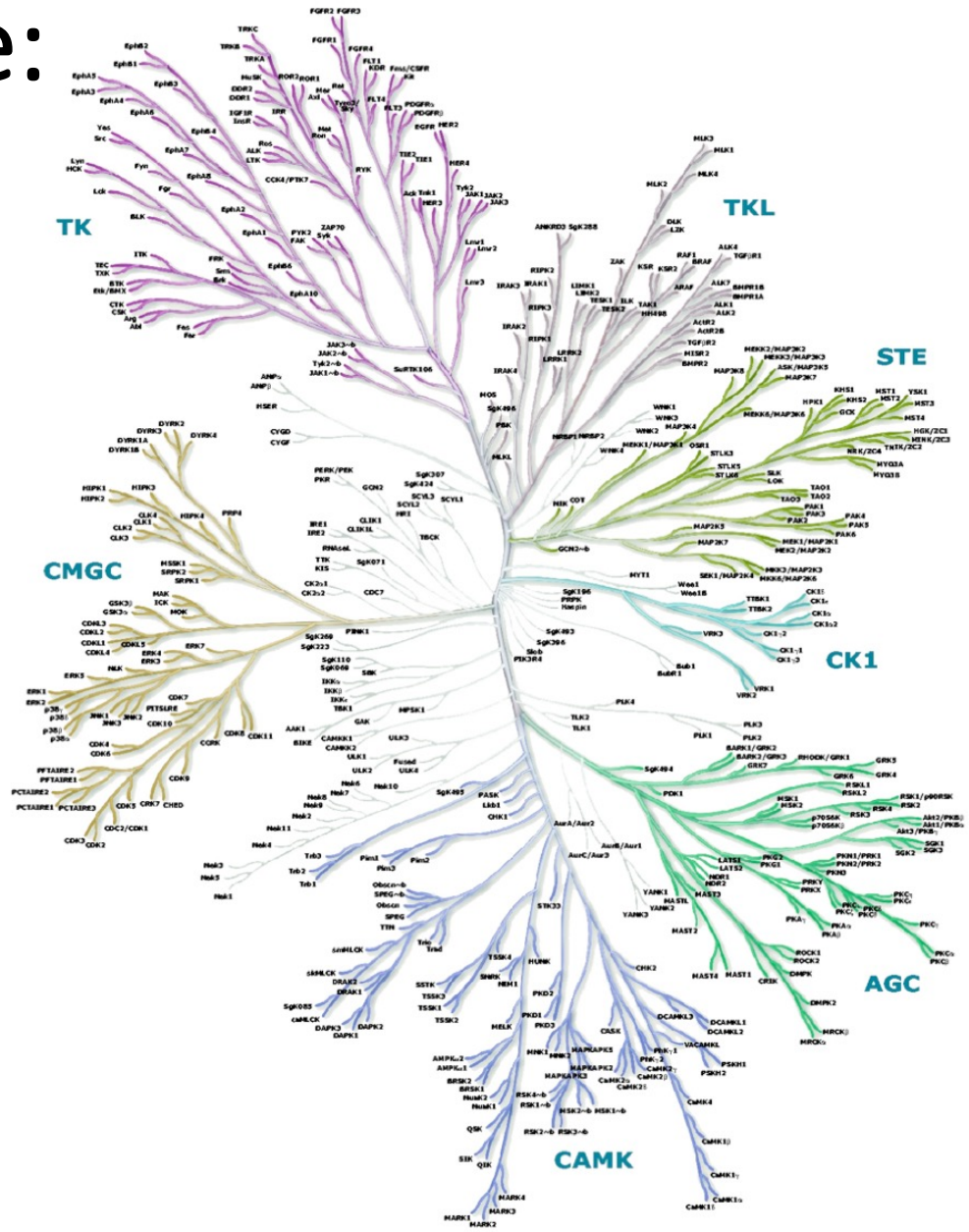
Baym et al., 2016

- *“Four-step trimethoprim MEGA-plate. The MEGA-plate with a trimethoprim gradient […] (0-3-30-300-3000-300-30-3-0). Movie was compiled from time-lapse imagery every 10 minutes for 11.7 days, and played at 30fps (18000X speed). Condensation on the lid is visible in the first several frames, and a single contaminating colony appears on the plate.“*

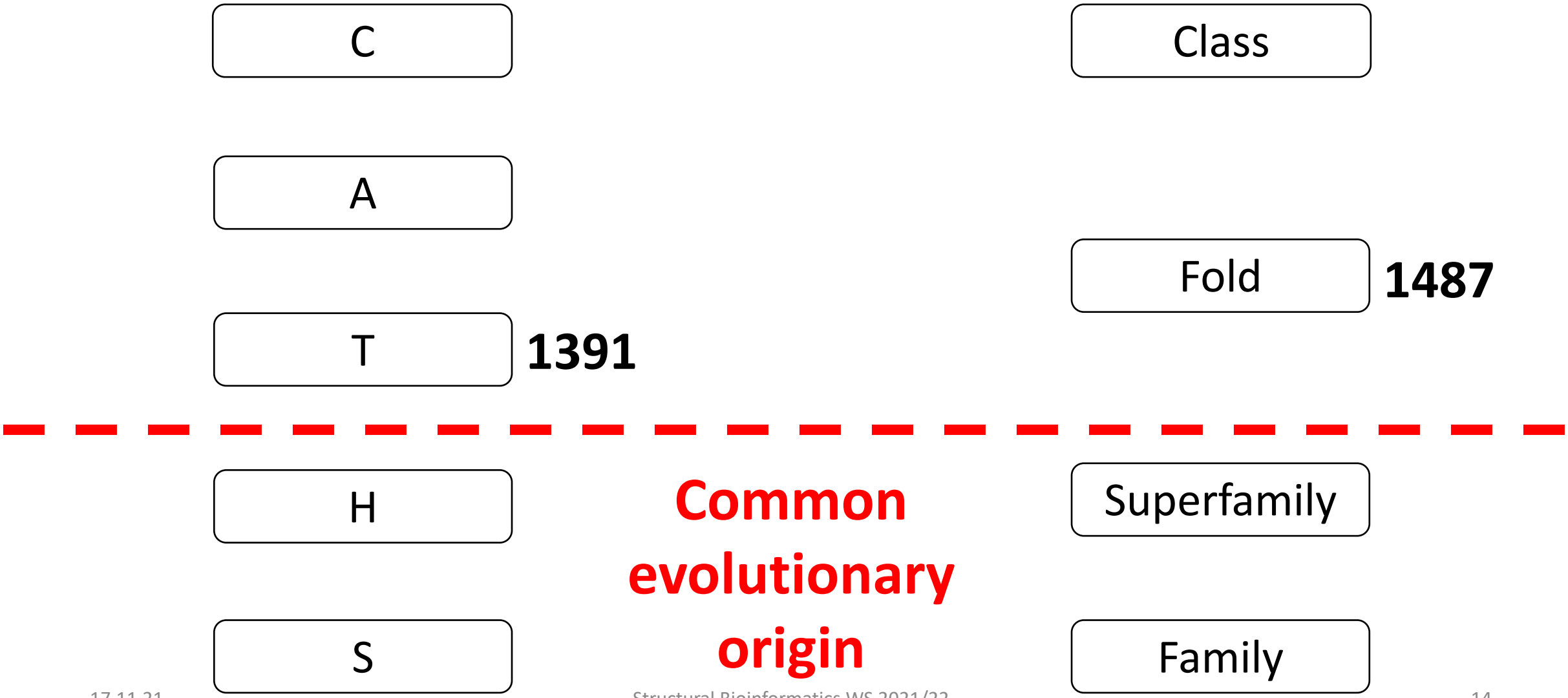
Evolution on a slow timescale: protein families

Some kinases have as little as 18% identity, yet same kinase fold





Evolutionary relations in CATH and SCOP

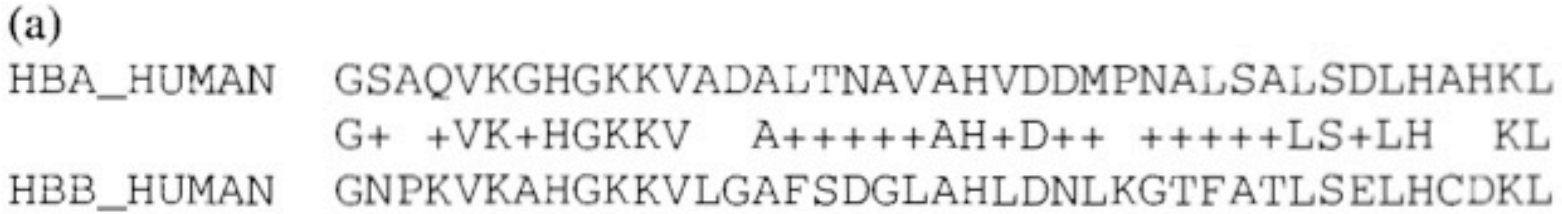


Homology and evolutionary relatedness

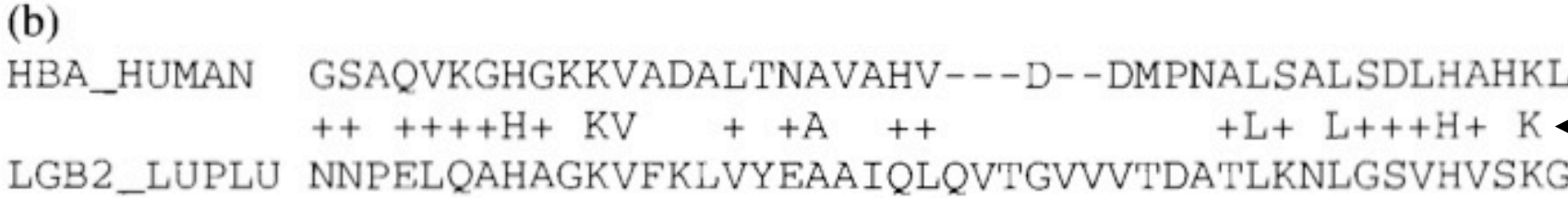
- Sequence similarity is a sign of common evolutionary origin
  - We can’t observe the past, so we don’t know for sure
  - A lot of indirect evidence, e.g. closely related species have more similar sequences for analogous proteins
- (Structural similarity — ???)
  - Proteins with similar sequences fold into similar structures
- To detect sequence similarity, we need to build a sequence **alignment**

Sequence alignment

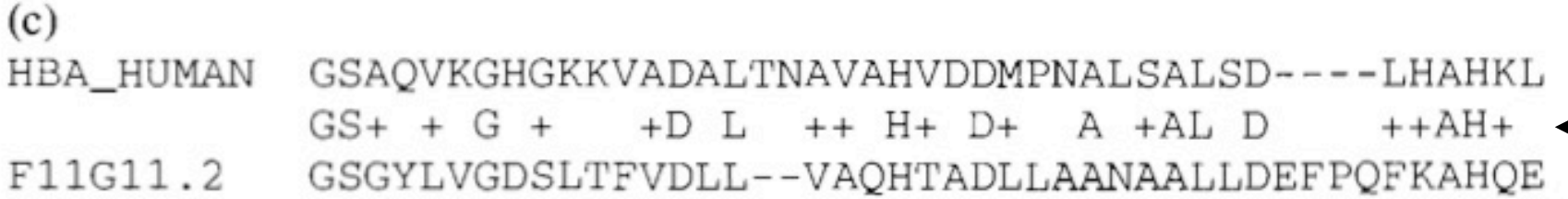
Sequence alignment: an example



Good alignment to an obviously related protein



Plausible alignment to a possibly related protein



Spurious alignment to an unrelated protein (can't tell that from it)

- **(Pairwise) sequence identity**: % identical letters in the aligned sequences
  - Already tells you something about evolutionary relatedness

Mutation model

```
HBA_HUMAN  GSAQVKGHGKKVADALTNAVAHV---D--DMPNALSALSDLHAHKL
           ++ ++++H+ KV    + +A   ++             +L+ L+++H+ K
LGB2_LUPLU NNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG
```

- **Substitutions** => changes a letter (amino acid)
- Insertions and deletions — **gaps** => introduces or removes letters (amino acids)
- Details: Durbin et al., “Biological Sequence Analysis”, Cambridge University Press, 1998

Alignment score

```
HBA_HUMAN  GSAQVKGHGKKVADALTNAVAHV---D--DMPNALSALSDLHAHKL
           ++ ++++H+ KV    + +A   ++           +L+ L+++H+ K
LGB2_LUPLU NNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG
```

- Sum of scores of all pairs of aligned letters (amino acids):

$$S = \sum_{i:\, not\ gaps} s(i) + \gamma$$

  - non-identical and identical pairs are scored: $s(i) = m(a_{1,i}, a_{2,i})$,
    i: alignment column, m: **substitution matrix**
    - 20x20 matrix that represents similarities of chemical properties of amino acids (e.g. BLOSUM, PAM, etc.)
  - gaps are penalized
    - linear gaps (of length g): $\gamma(g) = -dg$, d: **gap cost**
    - affine gaps (of length g): $\gamma(g) = -d\ -e(g-1)$,
      d: gap **opening** cost, e: gap **extension** cost

Substitution matrix: BLOSUM50

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
A	**5**	−2	−1	−2	−1	−1	−1	0	−2	−1	−2	−1	−1	−3	−1	1	0	−3	−2	0
R	−2	**7**	−1	−2	−4	1	0	−3	0	−4	−3	3	−2	−3	−3	−1	−1	−3	−1	−3
N	−1	−1	**7**	2	−2	0	0	0	1	−3	−4	0	−2	−4	−2	1	0	−4	−2	−3
D	−2	−2	2	**8**	−4	0	2	−1	−1	−4	−4	−1	−4	−5	−1	0	−1	−5	−3	−4
C	−1	−4	−2	−4	**13**	−3	−3	−3	−3	−2	−2	−3	−2	−2	−4	−1	−1	−5	−3	**−1**
Q	−1	1	0	0	−3	**7**	2	−2	1	−3	−2	2	0	−4	−1	0	−1	−1	−1	−3
E	−1	0	0	2	−3	2	**6**	−3	0	−4	−3	1	−2	−3	−1	−1	−1	−3	−2	−3
G	0	−3	0	−1	−3	−2	−3	**8**	−2	−4	−4	−2	−3	−4	−2	0	−2	−3	−3	−4
H	−2	0	1	−1	−3	1	0	−2	**10**	−4	−3	0	−1	−1	−2	−1	−2	−3	2	−4
I	−1	−4	−3	−4	−2	−3	−4	−4	−4	**5**	2	−3	2	0	−3	−3	−1	−3	−1	4
L	−2	−3	−4	−4	−2	−2	−3	−4	−3	2	**5**	−3	3	1	−4	−3	−1	−2	−1	1
K	−1	3	0	−1	−3	2	1	−2	0	−3	−3	**6**	−2	−4	−1	0	−1	−3	−2	−3
M	−1	−2	−2	−4	−2	0	−2	−3	−1	2	3	−2	**7**	0	−3	−2	−1	−1	0	1
F	−3	−3	−4	−5	−2	−4	−3	−4	−1	0	1	−4	0	**8**	−4	−3	−2	1	4	−1
P	−1	−3	−2	−1	−4	−1	−1	−2	−2	−3	−4	−1	−3	−4	**10**	−1	−1	−4	−3	−3
S	1	−1	1	0	−1	0	−1	0	−1	−3	−3	0	−2	−3	−1	**5**	2	−4	−2	−2
T	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	2	**5**	−3	−2	0
W	−3	−3	−4	−5	−5	−1	−3	−3	−3	−3	−2	−3	−1	1	−4	−4	−3	**15**	2	−3
Y	−2	−1	−2	−3	−3	−1	−2	−3	2	−1	−1	−2	0	4	−3	−2	−2	2	**8**	−1
V	0	−3	−3	−4	−1	−3	−3	−4	−4	4	1	−3	1	−1	−3	−2	0	−3	−1	**5**

Matrix of local scores

- Score of matching any two characters

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

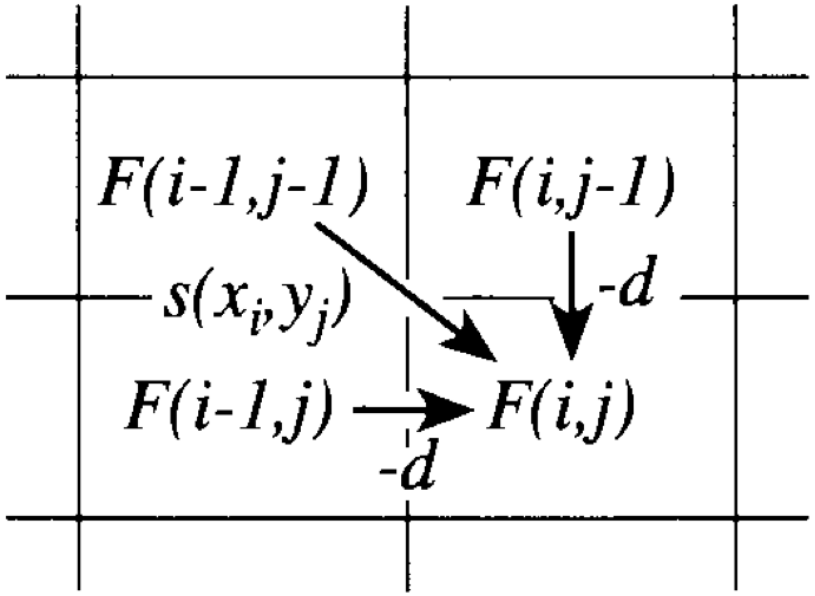
Alignment algorithm: dynamic programming

- Number of possible global alignments of two sequences of length n: $\binom{2n}{n} = \frac{(2n)!}{(n!)^2}$
- Dynamic programming solves the problem in polynomial time using **dynamic programming matrix**
- Global, local, semi-global, end-gap free alignments

Dynamic programming matrix

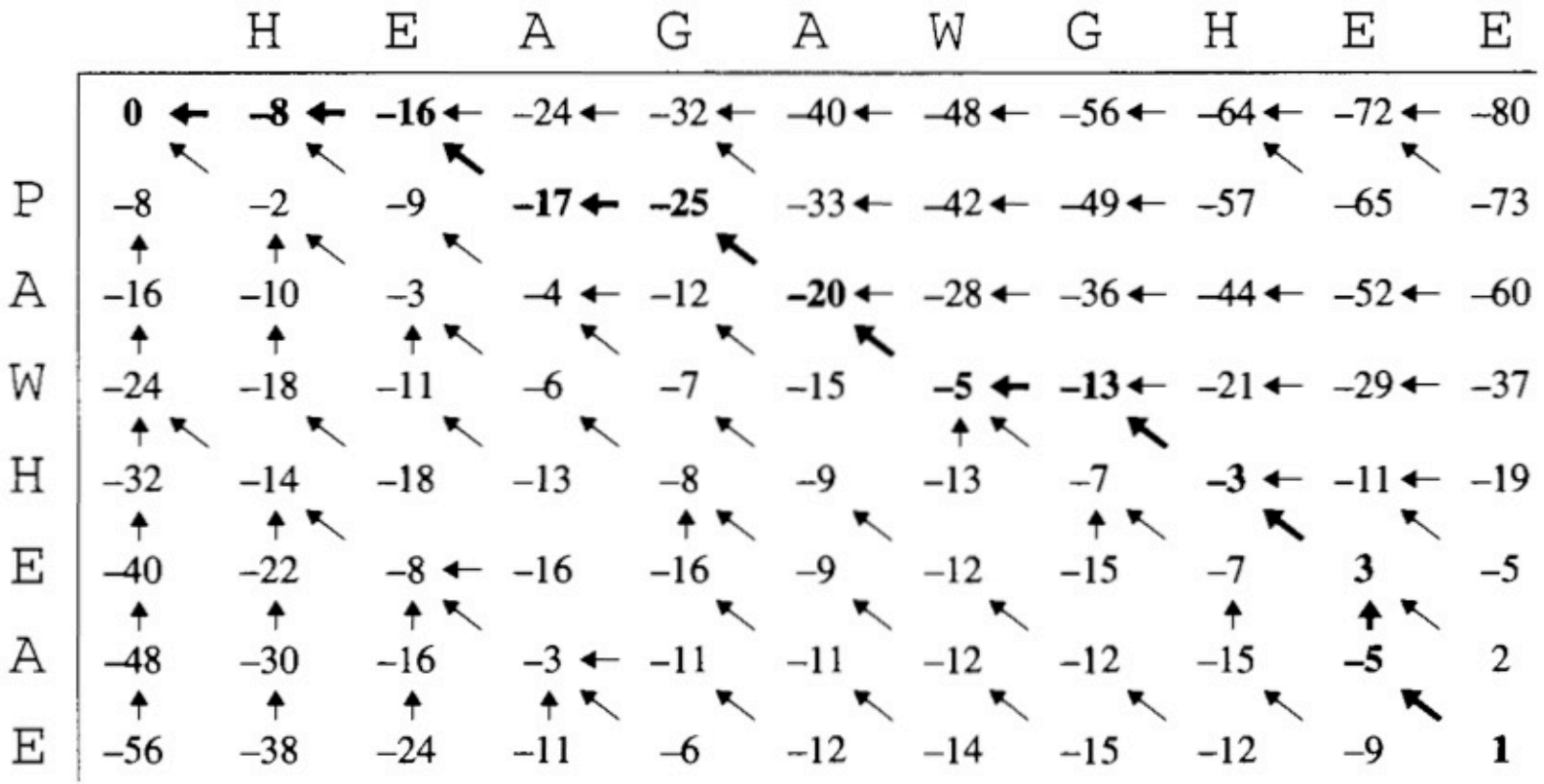
- Idea: construct longer alignment from a shorter segment

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



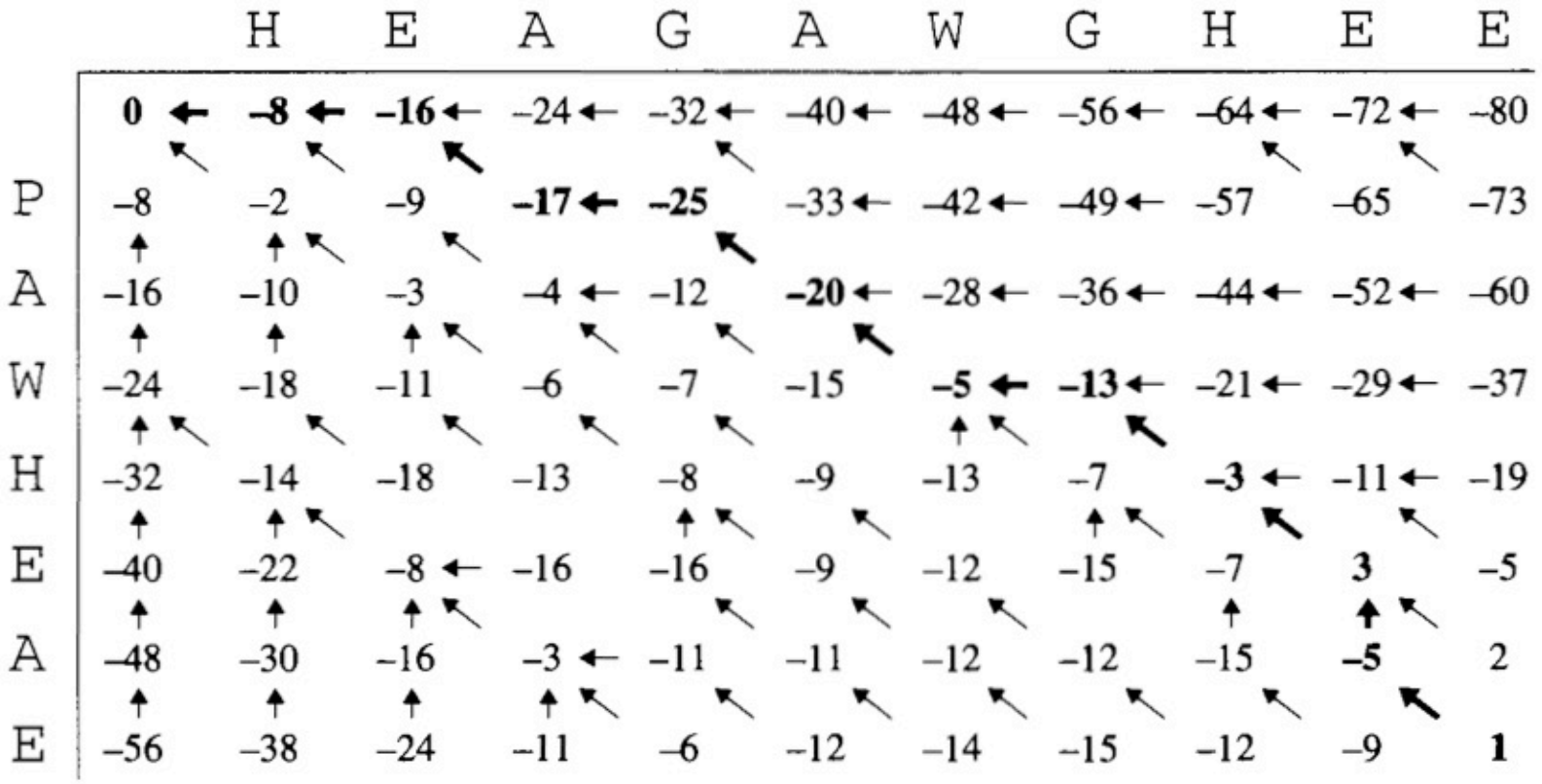
Dynamic programming matrix: example

- (Mis-)matches: see matrix of local scores
- Linear gap penalties, $d = -8$



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

Dynamic programming matrix: example



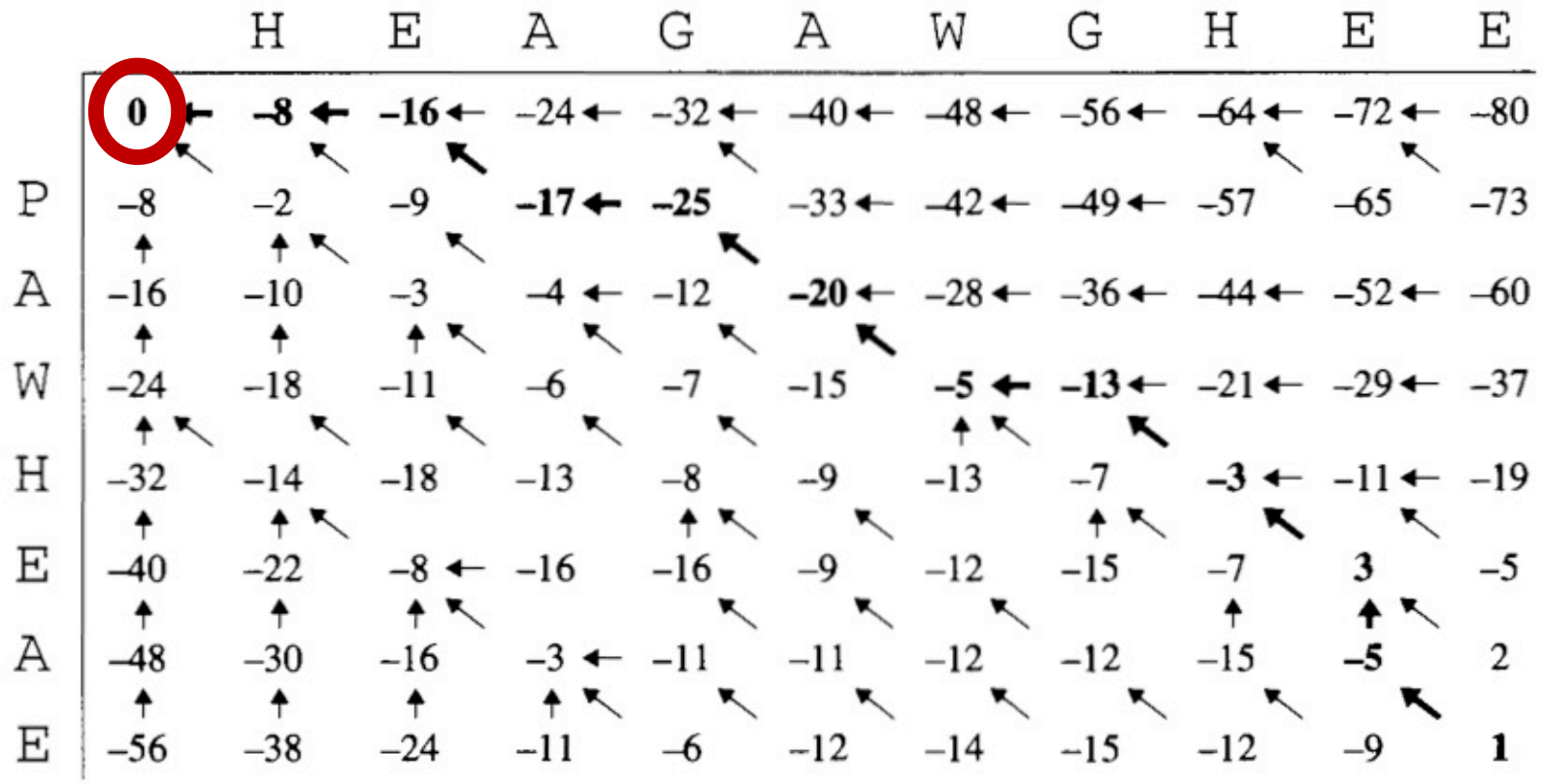
		H	E	A	G	A	W	G	H	E	E
	0	**–8**	**–16**	–24	–32	–40	–48	–56	–64	–72	–80
P	–8	–2	–9	**–17**	**–25**	–33	–42	–49	–57	–65	–73
A	–16	–10	–3	–4	–12	**–20**	–28	–36	–44	–52	–60
W	–24	–18	–11	–6	–7	–15	**–5**	**–13**	–21	–29	–37
H	–32	–14	–18	–13	–8	–9	–13	–7	**–3**	–11	–19
E	–40	–22	–8	–16	–16	–9	–12	–15	–7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

- Back-tracing yields **alignment**:

```
HEAGAWGHE-E
--P-AW-HEAE
```

Needleman-Wunsch algorithm (1970)

- **Global** pairwise sequence alignment
- Dynamic programming matrix is initialised with 0 in the top-left corner



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

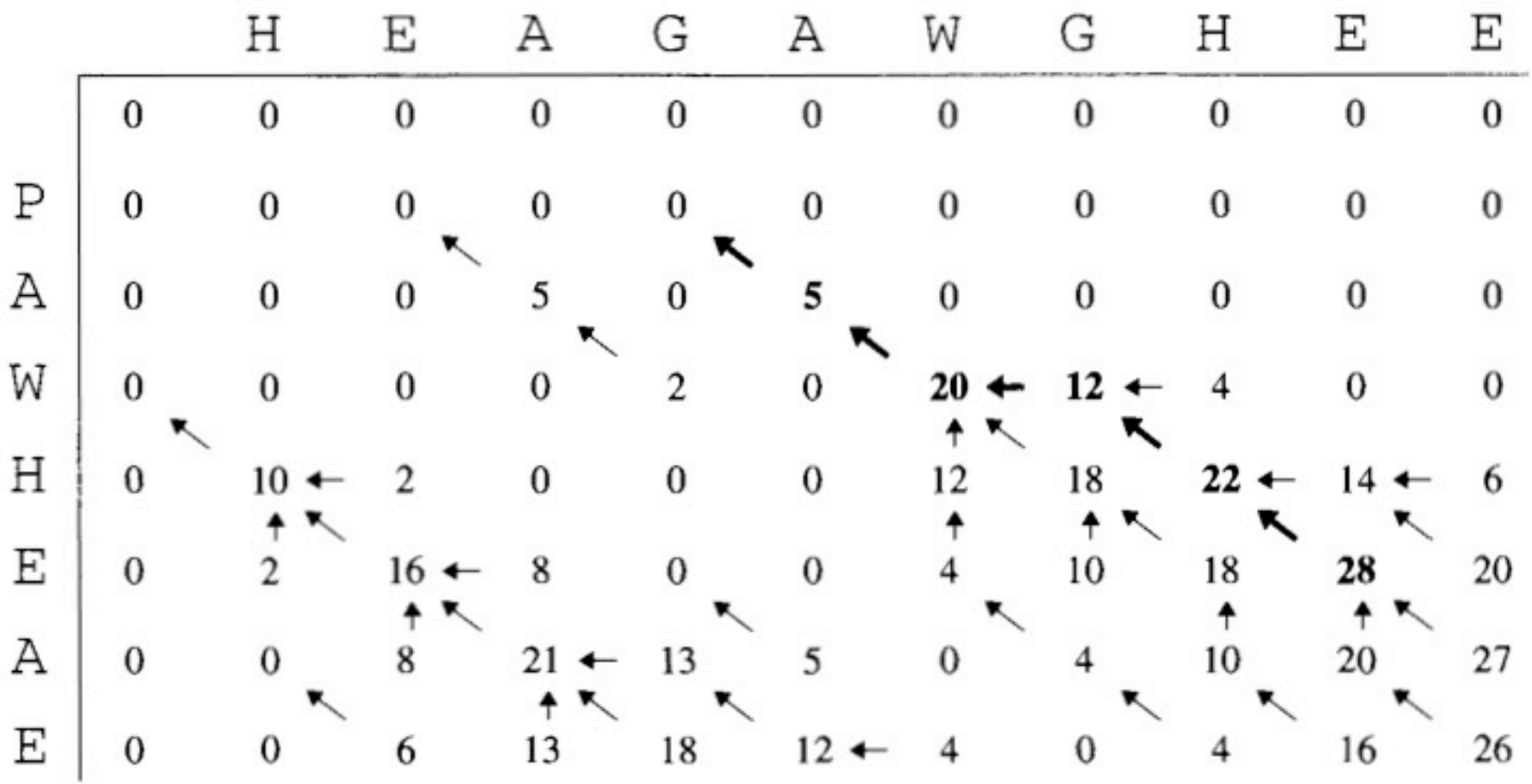
Smith-Waterman algorithm (1981)

- **Local** pairwise sequence alignment
  - Find the best alignable subsequences in two sequences
- Extended definition of $F(i, j)$:

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

- Trace-back from the maximum value in the matrix

Smith-Waterman algorithm: example



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

AWGHE

AW-HE

Local alignment: one important requirement

- Score of a random alignment must be **negative**!
  - Ungapped case: $\sum_{a,b} q_a q_b s(a,b) < 0$, a, b: all amino acids, q_a, q_b: global frequencies
  - No theoretical estimated for the gapped case

Dynamic programming: summary

- Filling the matrix
  - $F(i,j) = \max \begin{cases} F(i-1,j-1) + s(x_i, y_j), \\ F(i-1,j) - d, \\ F(i,j-1) - d, \\ [0 \text{ for local alignment}] \end{cases}$
  - a pointer to the cell, from which the value *F*(*i*, *j*) was derived
- Trace-back
  - reconstruction of the path using the pointers yields the alignment

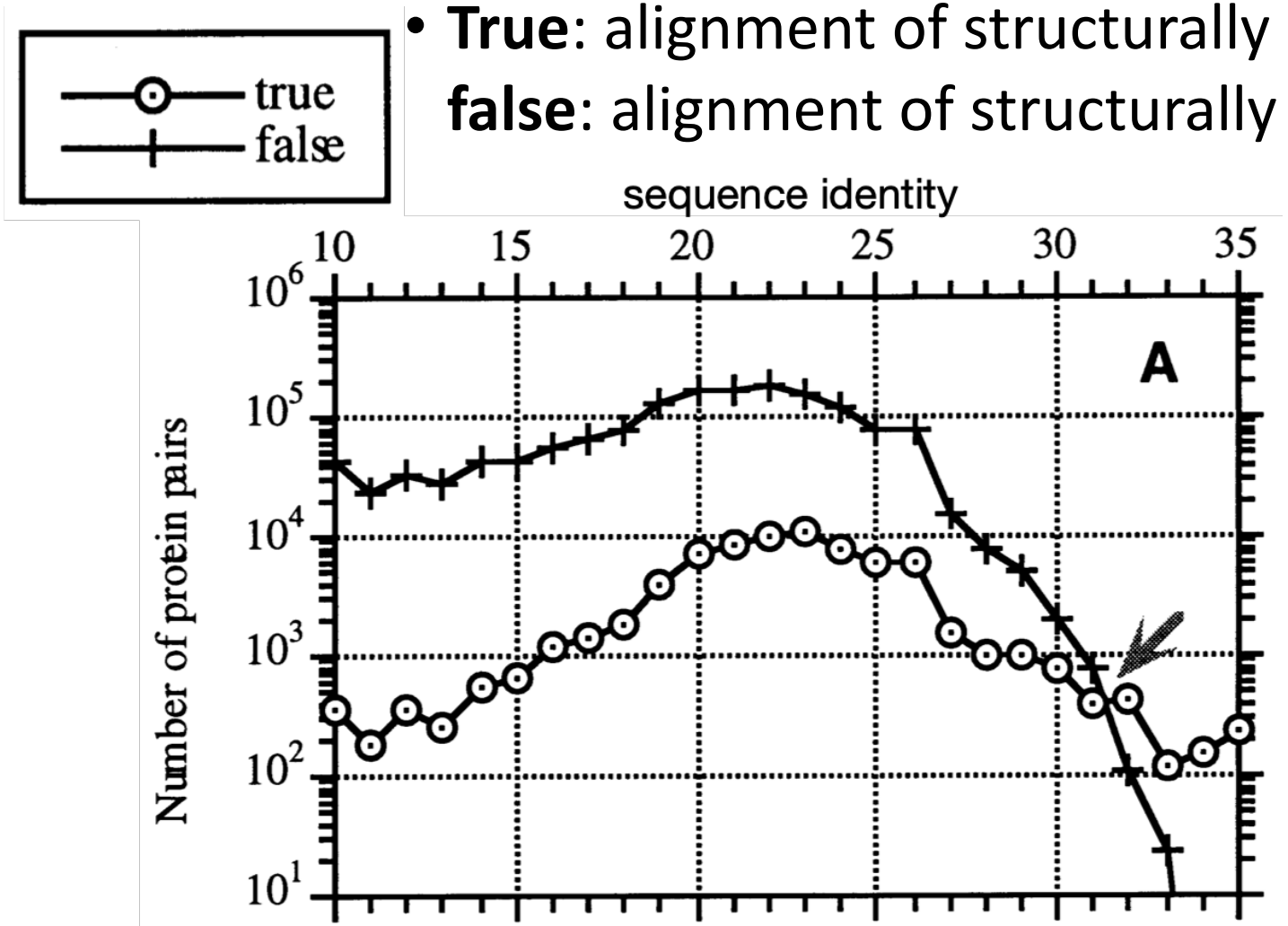
Alignments can be built with other models

- Finite state automata
- Hidden Markov models
- …

Twilight zone of sequence alignments (Rost, 1999)

- **Q**: How are sequence identity and structural similarity related?
- Align all proteins with known 3D structures (as of 1997)
  - 792 structurally non-redundant protein chains aligned vs. 5646 other protein chains
  - **True**: alignment of structurally similar proteins;
    false: alignment of structurally dissimilar proteins

Twilight zone of sequence alignments (Rost, 1999)

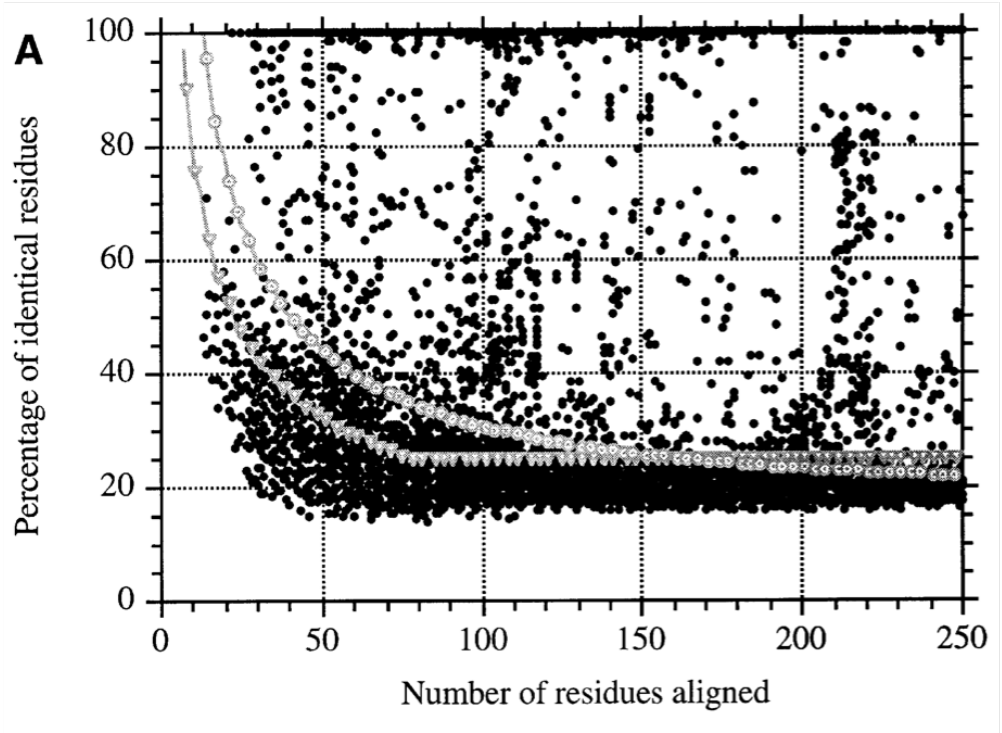


- **True**: alignment of structurally similar proteins;
  false: alignment of structurally dissimilar proteins

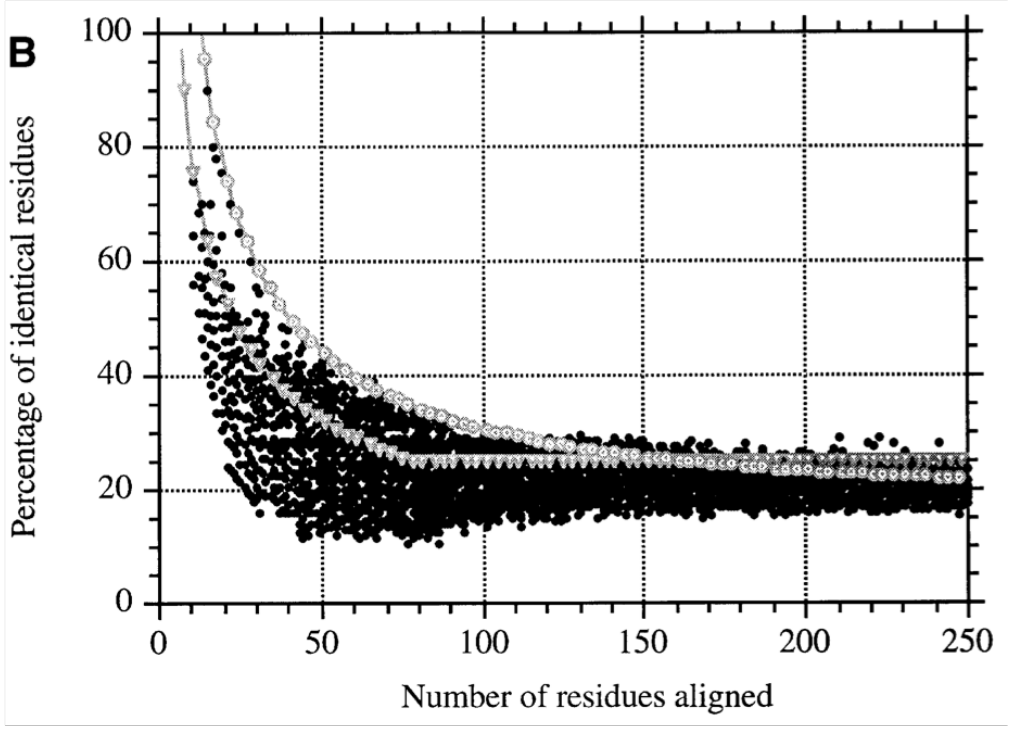
Twilight zone of sequence alignments (Rost, 1999)

- **True**: alignment of structurally similar proteins;
  false: alignment of structurally dissimilar proteins

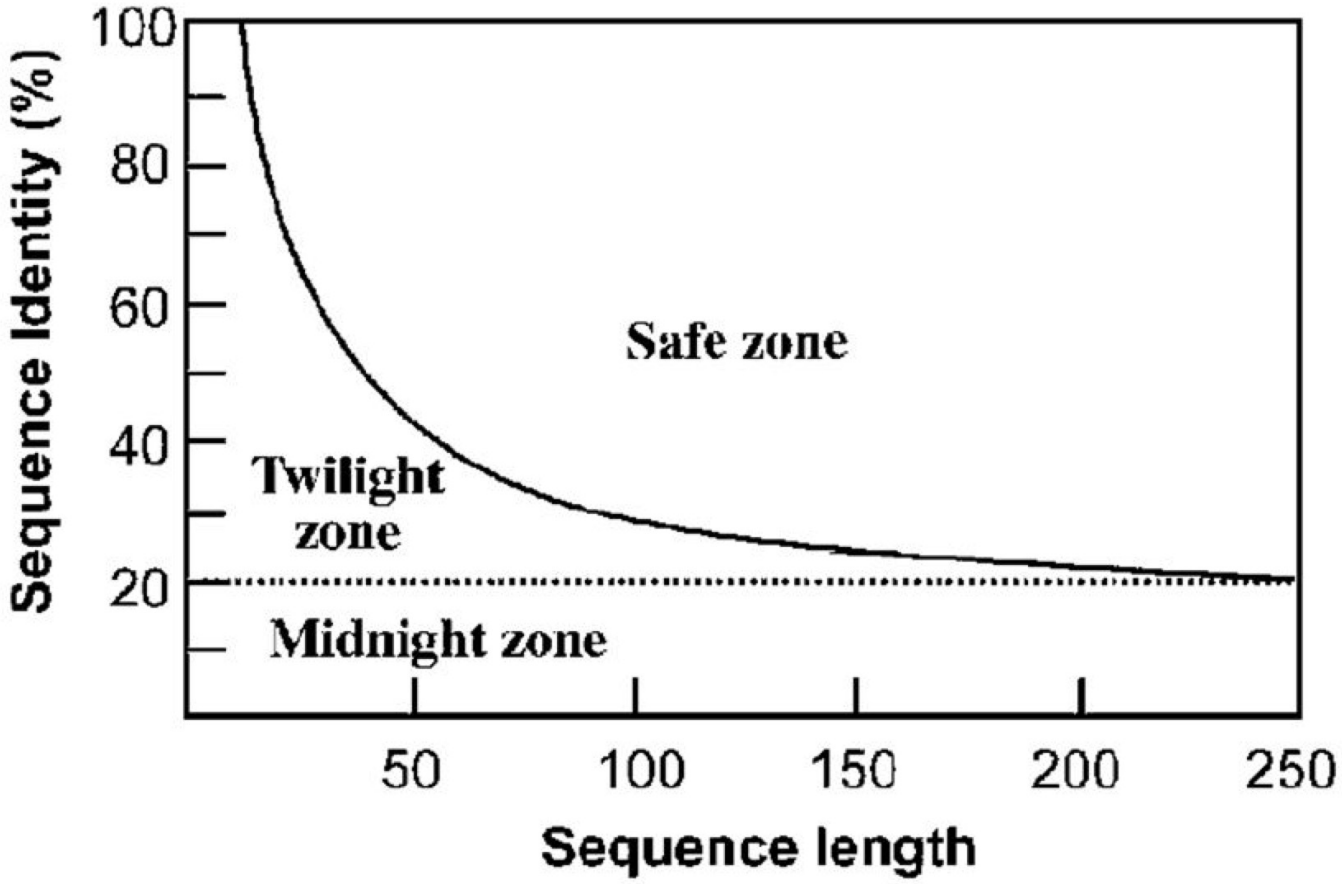
True



False



Twilight zone of sequence alignments (Rost, 1999)



Sequence similarity search

Sequence similarity search vs. alignment

- Not the same thing!
- **Similarity search**: detecting similar proteins in a database
- **Alignment**: mapping between amino acid residues
- Alignment ⇒ similarity (if %identity is sufficient),
  Similarity ⇏ alignment

Sequence similarity search: why bother?

- Alignment = dynamic programming: $O(n^2)$ in time and memory
- GenBank: 219,055,207 + 1,432,874,252 (WGS) sequences (Oct 2020)
  - WGS doubles since last year
- UniProt: 563,972 (SwissProt) + 209,157,139 (TrEMBL) sequences
- Length of a single protein ≈ 400 amino acids => size of an all-against-all dynamic programming matrix:
  $\approx ((563{,}972 + 209{,}157{,}139) * 400)^2 \approx (200{,}000{,}000 * 400)^2 \approx 80{,}000{,}000{,}000^2 \approx 10^{22}$
- => we need something faster than $O(n^2)$

BLAST: Basic Local Alignment Search Tool

- Altschul, S.F., Gish, W., Miller, W., Myers, E.W. & Lipman, D.J. (1990) "Basic local alignment search tool." J. Mol. Biol. 215:403-410.
- Most cited bioinformatics paper: 88,439 citations
- https://blast.ncbi.nlm.nih.gov/Blast.cgi

BLAST is a heuristic method

- **Correct** method: guaranteed to find the optimal solution (alignment)
  - Dynamic programming-based methods are correct
- **Heuristic**: does not guarantee the best solution, just a good one
  - BLAST is a heuristic method

Maximal Segment Pair (MSP)

- The **highest-scoring** pair of segments of **identical length** from the two sequences that are being aligned
- Statistical significance of the score for an MSP can be estimated (Karlin and Altschul, 1990)

The Annals of Statistics
1990, Vol. 18, No. 2, 571–581

STATISTICAL COMPOSITION OF HIGH-SCORING SEGMENTS FROM MOLECULAR SEQUENCES

BY SAMUEL KARLIN[1], AMIR DEMBO AND TSUTOMU KAWABATA

Proc. Natl. Acad. Sci. USA
Vol. 87, pp. 2264–2268, March 1990
Evolution

Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes

(sequence alignment/protein sequence features)

SAMUEL KARLIN† AND STEPHEN F. ALTSCHUL‡§

Sequence random model

- Sequence = random process
- Next letter appears independently from the previous ones
  - a_i appears with a probability p_i (e.g. from all available sequence data)
- Given an alphabet $a = (a_1, a_2, \ldots, a_r)$ and a scoring scheme $s = (s_1, s_2, \ldots, s_r)$, what is a chance that a random sequence will have a score per character greater than a pre-defined?
  - E.g. hydrophobicity profile

Probability of a high-scoring sequence segment

- Given an alphabet $a = (a_1, a_2, \ldots, a_r)$ and a scoring scheme $s = (s_1, s_2, \ldots, s_r)$, what is a chance that a random sequence will have a score per character greater than a pre-defined q?

THEOREM 1 [Iglehart (1972) and Karlin, Dembo and Kawabata (1990)].

$$\lim_{n \to \infty} \Pr\left\{ M(n) - \frac{\ln n}{\theta^*} \leq x \right\} = \exp\{ -K^* e^{-\theta^* x} \}, \tag{4}$$

Probability of a high-scoring sequence segment

THEOREM 1 [Iglehart (1972) and Karlin, Dembo and Kawabata (1990)].

$$\lim_{n\to\infty} \Pr\left\{ M(n) - \frac{\ln n}{\theta^*} \le x \right\} = \exp\{-K^* e^{-\theta^* x}\}, \quad (4)$$

- $X_1, X_2, \ldots, X_n, \ldots$: a sequence of independent identically distributed (i.i.d.) random variables, such that $Prob\{X = s_i\} = p_i$
- $S_0 = 0$, S_k: partial sum process
- $M(n) = sup_{0 \le k \le l \le n}(S_l - S_k)$: maximal score of a segment from a sequence of length *n*
- Expected score is negative ($E[X] = \sum p_i s_i < 0$)
- ϑ^*: unique positive root of equation $E\left[e^{\vartheta X}\right] = \sum p_i e^{\vartheta s_i} = 1$, *K** can also be extpressed via *S.* and ϑ^*

Probability of a high-scoring sequence segment

THEOREM 1 [Iglehart (1972) and Karlin, Dembo and Kawabata (1990)].

$$\lim_{n\to\infty} \Pr\left\{ M(n) - \frac{\ln n}{\theta^*} \le x \right\} = \exp\{-K^* e^{-\theta^* x}\}, \tag{4}$$

Probability of a high-scoring segment in a sequence of length *n* is exponentially distributed

Probability of a high-scoring MSP

- Comparison of two sequences, **no gaps!**
- Two independent sequences of lengths *n* and *m* from an alphabet *a* with r characters with probabilities $p = (p_1, p_2, \dots, p_r)$ and $p' = (p'_1, p'_2, \dots, p'_r)$ => a pair of letters a_i in seq1 and a_j in seq2 occurs with probability $p_i p'_j$
- Then probability of an MSP with a score *M* is distributed as

$$Prob\left\{M - \frac{\ln mn}{\lambda^*} > x\right\} \leq K^* e^{-\lambda^* x}$$

λ^* is the unique positive root of $\sum p_i p'_j e^{\lambda s_{ij}} = 1$, s_{ij} is the similarity matrix, expected pair score $\sum p_i p'_j s_{ij} < 0$, there exist i, j: $s_{ij} > 0$

Probability of a high-scoring MSP

$$Prob\left\{M - \frac{\ln mn}{\lambda^*} > x\right\} \leq K^* e^{-\lambda^* x}$$

Probability of a high-scoring MSP in a pair of sequences of lengths *n* and *m* is exponentially distributed

Defining a good substitution matrix

- Suppose we have a set of MSPs, q_{ij} are the observed frequencies of letters *i* and *j*
- Expected score of a random MSP: $\sum p_i p_j s_{ij}$ (probabilities in both sequences are identically distributed),
  score of observed MSPs: $\sum q_{ij} s_{ij}$
- Odd ratios $q_{ij}/p_i p_j$ represent how much good MSPs are better than random
  => $s_{ij} = \frac{q_{ij}}{p_i p_j}$ is a good substitution matrix

E-value

$$Prob\left\{M - \frac{\ln mn}{\lambda^*} > x\right\} \leq K^* e^{-\lambda^* x}$$

- Expected number of MSPs with a score larger than ***S*** when comparing two sequences of lengths *n* and *m*: **E-value $\boldsymbol{E = mnK^* e^{-\lambda^* S}}$**
- When searching against a database, the second sequence can be interpreted as all sequences in the database concatenated
- The key parameter reflecting significance of your hits!
  - Can be adjusted in the search, typically set to 1e-5 (10^{-5})

Improved traversal of the search space

- Only consider sequences that are likely to provide MSP with a score > *S*: score of a random match
- For this, define a **word pair**: a segment pair of length *w*
  - For MSP, the length is not defined in advance
- Seek sequences that contain words, such that the score of the word pair > *T*: some threshold (TBD)
- Extend the word pairs to see where the corresponding MSP has a score > *S*
  - The lower *T*, the more sensitive the algorithm, but the longer the execution time

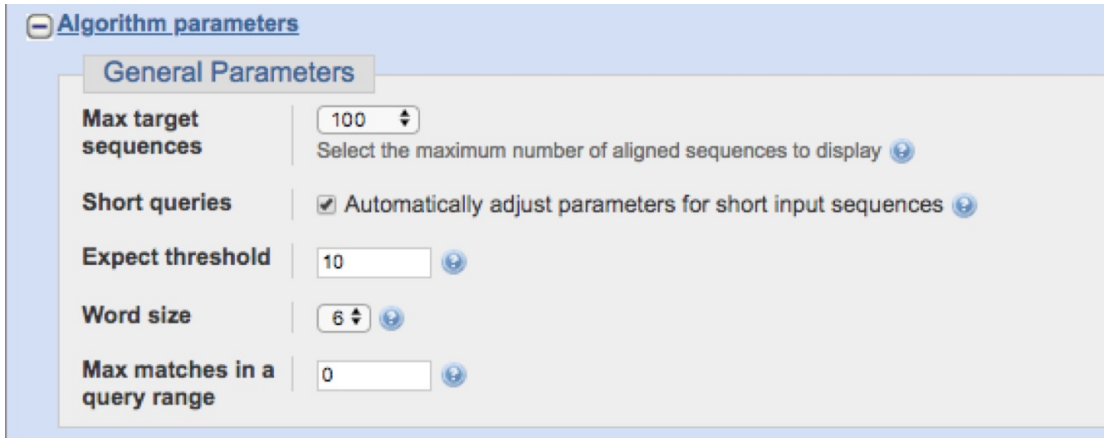
BLAST: three steps of the algorithm

1. Compile a list of high-scoring words
2. Scan the database for potential hits (containing the high-scoring words)
3. Extend the hits

- Implemented differently for protein and nucleic acid sequences

Step 1: Compile a list of all high-scoring words

- Split the query sequence in *w*-mers, find all possible *w*-mers that score at least *T* compared to the query *w*-mers
  - (for nucleic acids, only the query *w*-mers are considered)
- *w*, *T* are chosen based on the estimated number of MSPs that would be missed
  - Smaller *w*: more precise search
  - Larger *w*: more time to build the list of high-scoring words
  - Recommended size: *w* = 4 (Altschul et al., 1990)
  - Can be adjusted in NCBI‘s BLAST web interface
- *w* = 3, *T* = 17: ca. 50 words per position of the query sequence



Step 2: Scan the database

- Build an index of the database: words → all its occurrences
- Scan using the index

Step 3: Extending word pairs to find MSP

- Extend until the score drops below *S* or by a certain margin compared to the score of a shorter sequence pair
  - => not guaranteed to find the highest-scoring MSP
  - but the probability of that is low
- Use dynamic programming to extend hits

Implementation

- Web-based: https://blast.ncbi.nlm.nih.gov/Blast.cgi
- Command-line tool

Summary and possible exam questions

- Relationship between protein evolution and sequence similarity
- Relationship between protein evolution and similarity of 3D structures
- Alignment twilight, midnight zones
- Dynamic programming: filling the matrix and trace-back
- Needleman-Wunsch and Smith-Waterman algorithms
- Difference between sequence alignment and sequence similarity search algorithms
- BLAST: distribution of the probability of a hit, E-value
- The three steps of the BLAST algorithm