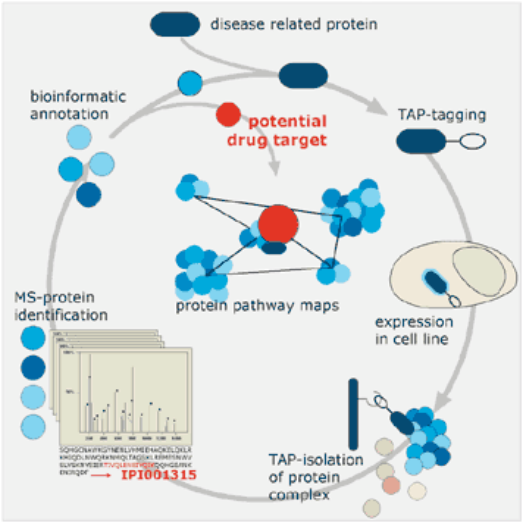
Softwarewerkzeuge der Bioinformatik

Inhalt dieser Veranstaltung: Softwarewerkzeuge kennenlernen für

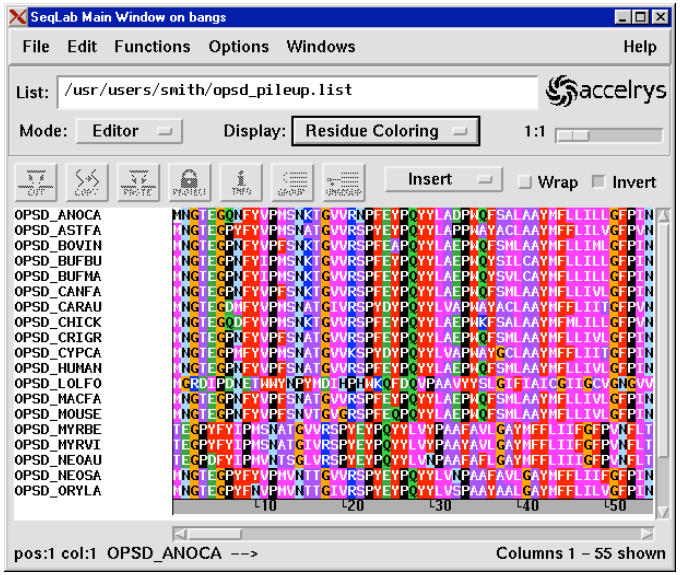
I Sequenzanalyse

II Analyse von Proteinstruktur und Ligandenbindung

III Zell- bzw. Netzwerksimulationen



www.cellzome.com



www.accelrys.com

Organisatorisches: Scheinvergabe
B.Sc. Bioinformatik und Biotechnologie M.Sc.

Voraussetzung für die Teilnahme an der Abschlussklausur ist das Erreichen von mindestens 50 % der maximalen Punkte aus den drei Praktikumsberichten.

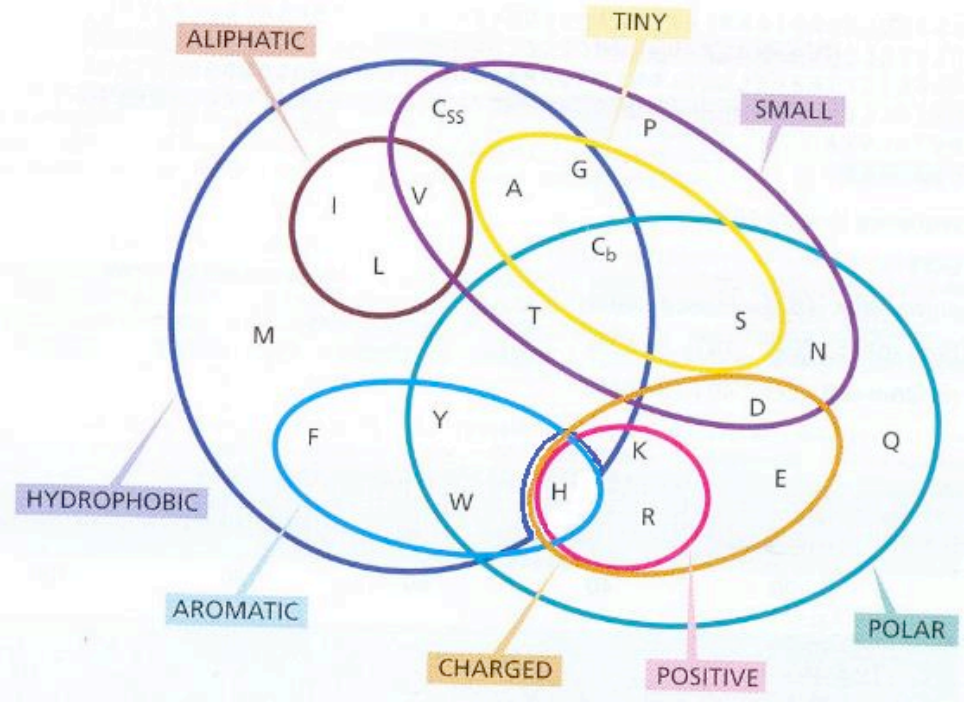
Die Veranstaltung gilt als bestanden, wenn in der abschließenden 120-minütigen Klausur über die Inhalte der Vorlesung, der Übungen und der Minipraktika mindestens die Note 4 erreicht wurde.

Für die Note des Scheins zählt das bessere Ergebnis entweder ausschließlich aus der abschließenden Klausur oder der Kombination des Durchschnitts der benoteten Praktika und der Note der Abschlussklausur, die jeweils zu 50 % gewichtet werden.

Bei Nichtbestehen der Klausur besteht die Möglichkeit einer schriftlichen oder mündlichen **Nachprüfung**. Diese findet im allgemeinen zu Beginn des darauffolgenden Semesters statt.

Eigenschaften der Aminosäuren

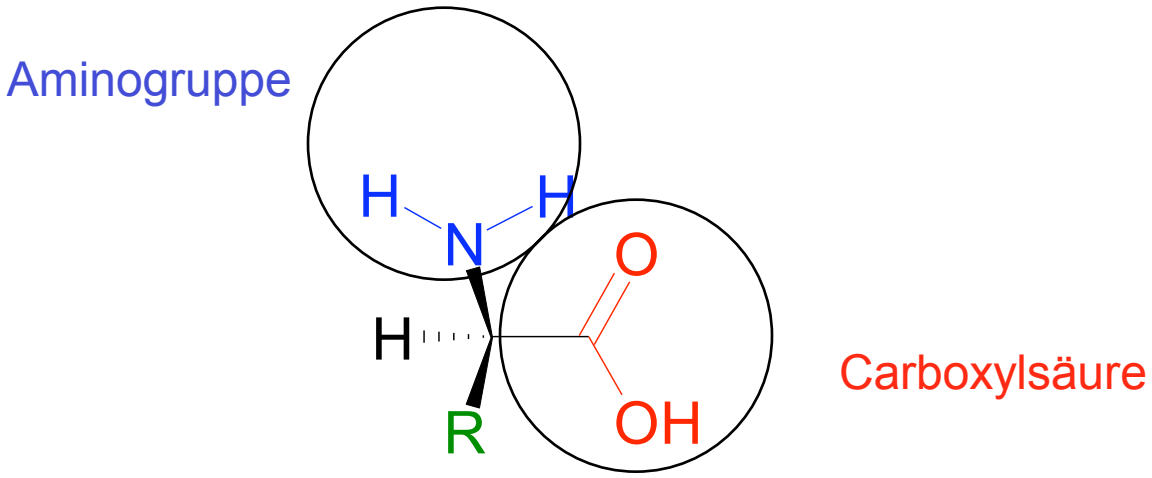
Aminosäuren unterscheiden sich in ihren physikochemischen Eigenschaften.



Q: müssen Bioinformatiker die Eigenschaften von Aminosäuren kennen?

Einleitung: Aminosäuren

Aminosäuren sind die **Bausteine** von Proteinen:



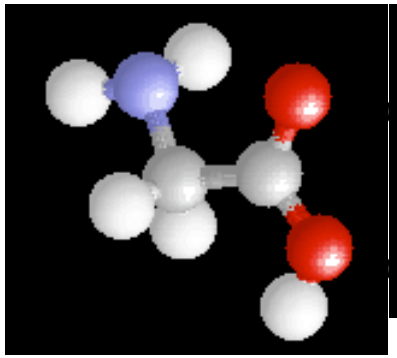
Aminosäuren unterscheiden sich hinsichtlich ihrer

- Größe
- elektrischen Ladung
- Polarität
- Form und Steifigkeit

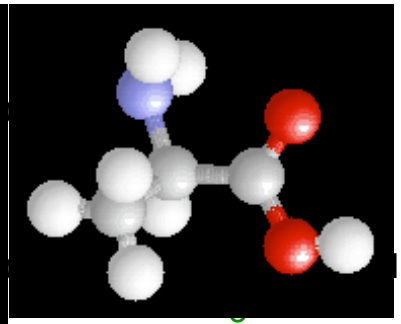
Einleitung: hydrophobe Aminosäuren

Proteine sind aus 20 verschiedenen natürlichen Aminosäuren aufgebaut

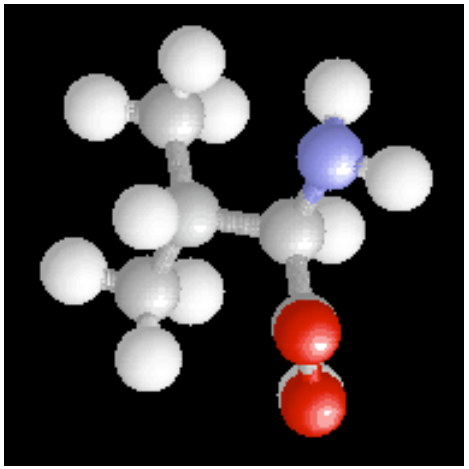
5 sind hydrophob.
Sie sind vor allem
Im Proteininneren.



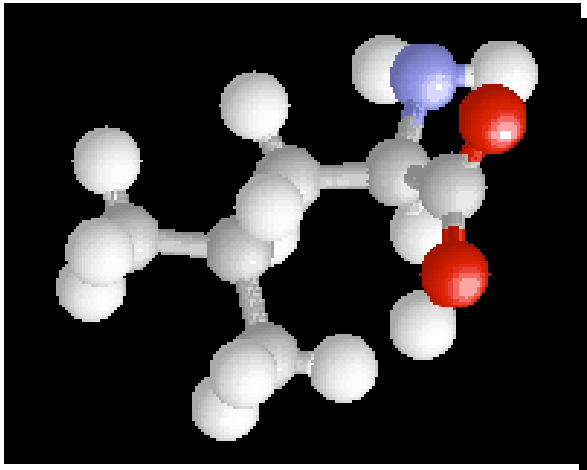
Glycine



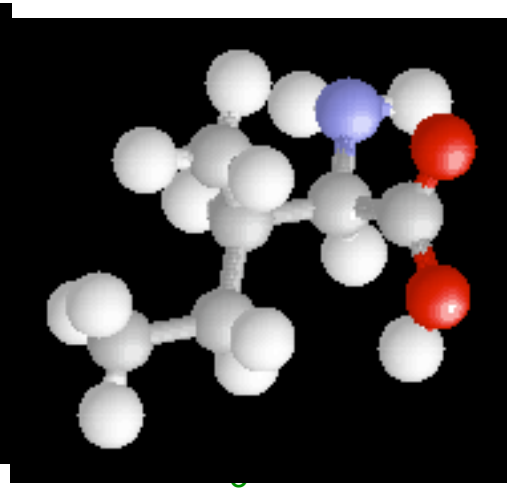
Alanine



Valine



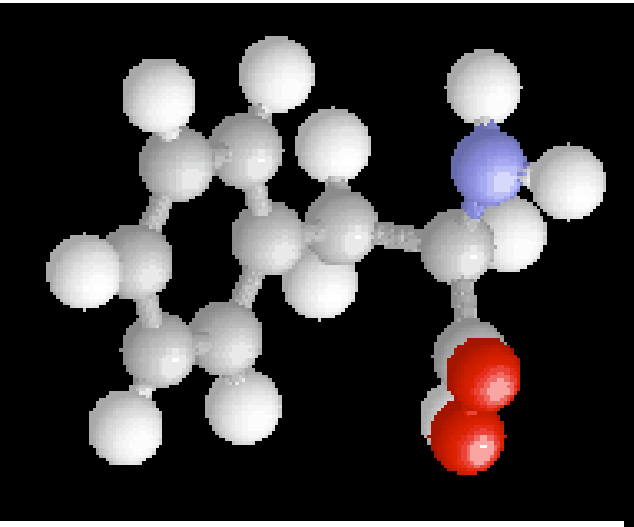
Leucine



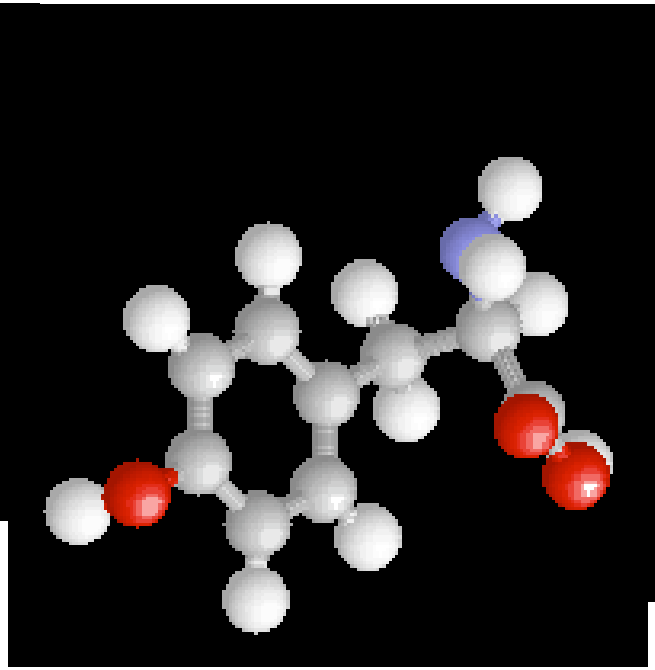
Isoleucine

Einleitung: aromatische Aminosäuren

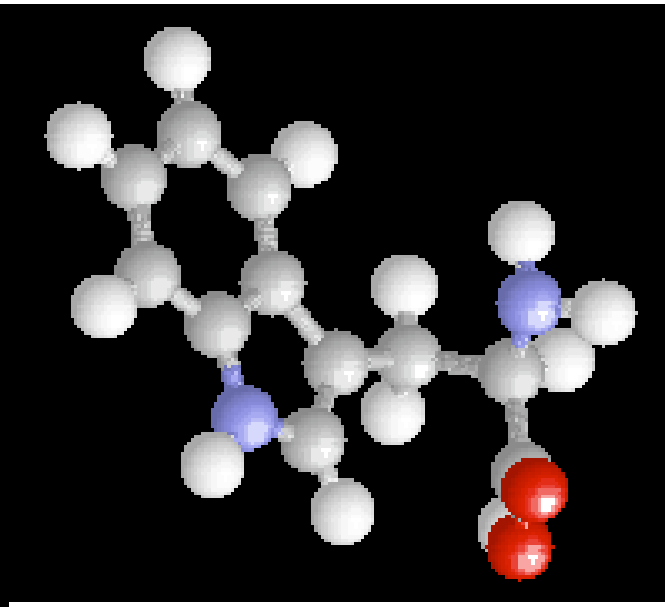
Es gibt drei voluminöse aromatische Aminosäuren. Tyrosin und Tryptophan liegen bei Membranproteinen vor allem in der Interface-region.



Phenylalanin



Tyrosin



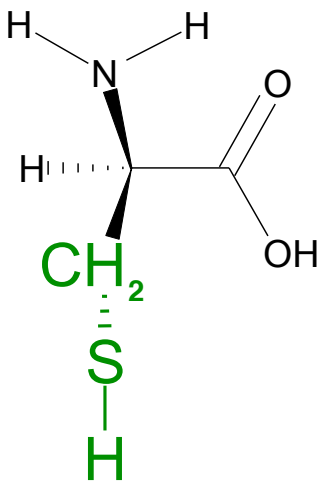
Tryptophan

Einleitung: Aminosäuren

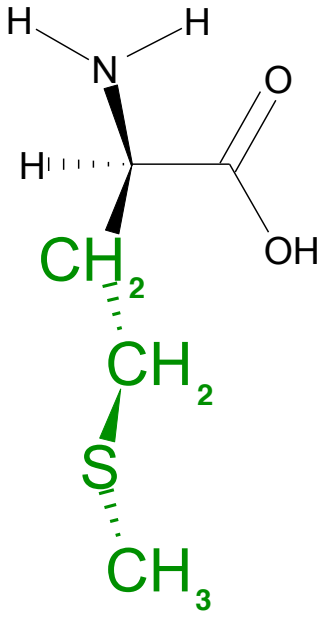
Es gibt 2 Schwefel enthaltende Aminosäuren und das ungewöhnliche Prolin.

Cysteine können Disulfidbrücken bilden.

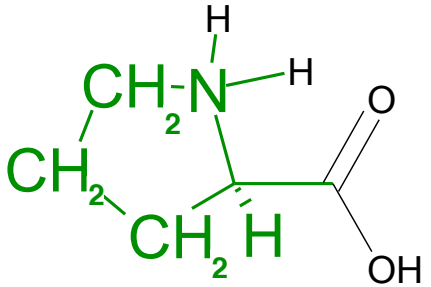
Prolin ist ein “Helixbrecher”.



Cystein



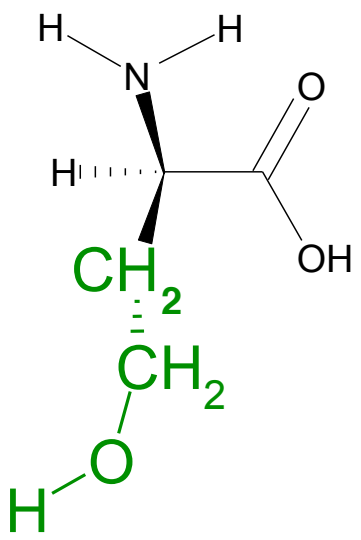
Methionin



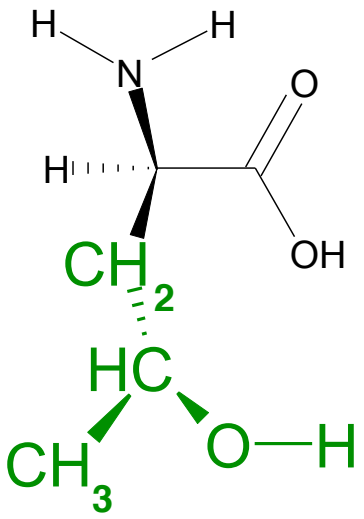
Prolin

Einleitung: Aminosäuren

Es gibt zwei Aminosäuren mit terminalen polaren Hydroxlgruppen:



Serin

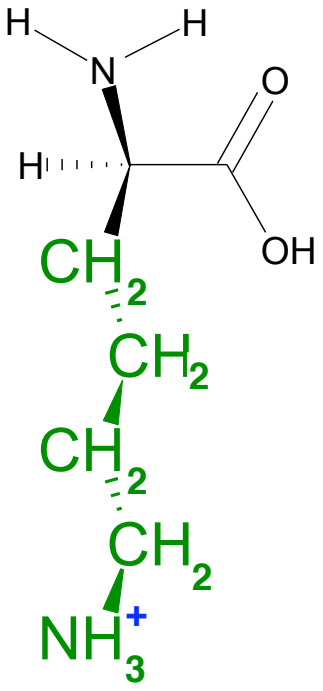


Threonin

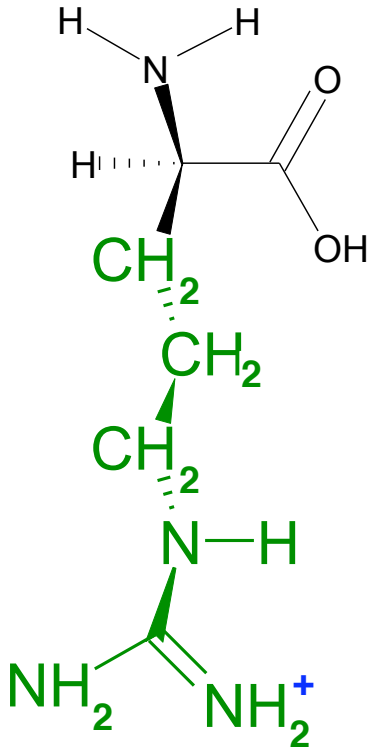
Einleitung: Aminosäuren

Es gibt 3 positiv geladene Aminosäuren. Sie liegen vor allem auf der Proteinoberflächen und in aktiven Zentren.

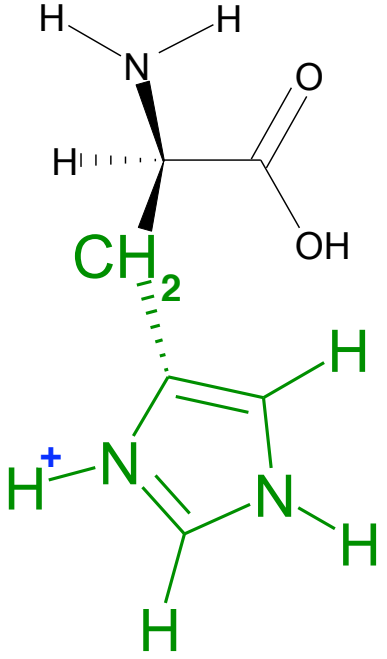
Thermophile Organismen besitzen besonders viele Ionenpaare auf den Proteinoberflächen.



Lysin



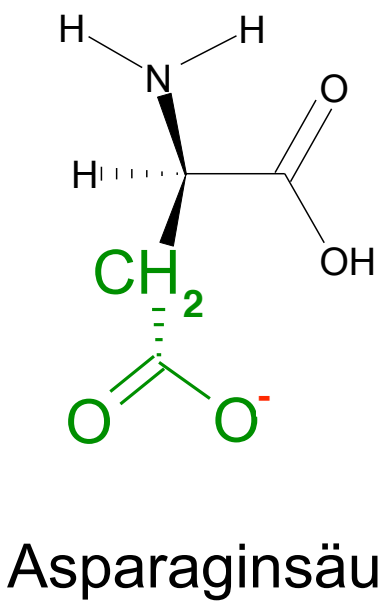
Arginin



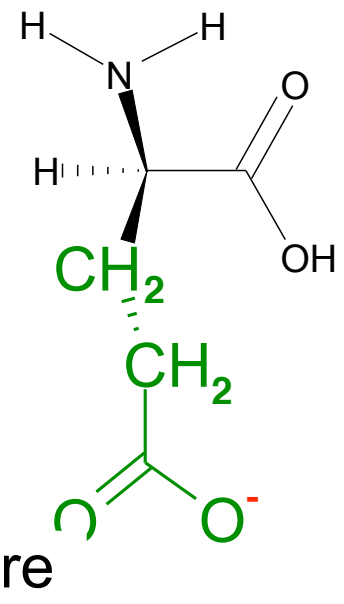
Histidin

Einleitung: Aminosäuren

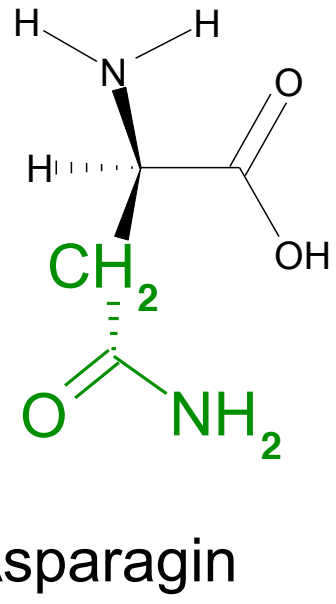
Es gibt 2 negativ geladene Aminosäuren und ihre zwei neutralen Analoga.
Asp und Glu haben pK_a Werte von 2.8. Das heisst, erst unterhalb von pH=2.8 werden ihre Carboxylgruppe protoniert.



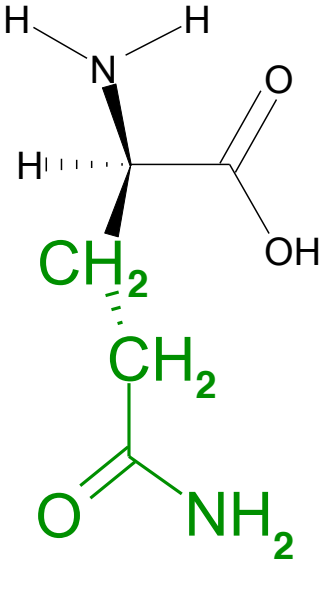
Asparaginsäure



Glutaminsäure



Asparagin



Glutamin

Buchstaben-Code der Aminosäuren

- **Ein**- und **Drei-Buchstaben-Codes** der Aminosäuren

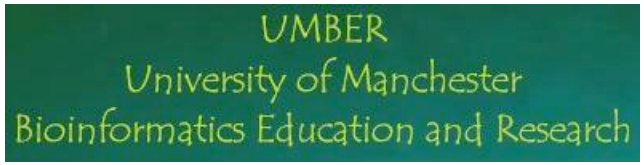
G	Glycin	**Gly**	**P**	Prolin	**Pro**	
A	Alanin	**Ala**	**V**	Valin	**Val**	
L	Leucin	**Leu**	**I**	Isoleucin	**Ile**	
M	Methionin	**Met**	**C**	Cystein	**Cys**	
F	Phenylalanin	**Phe**	**Y**	Tyrosin	**Tyr**	
W	Tryptophan	**Trp**	**H**	Histidin	**His**	
K	Lysin	**Lys**	**R**	Arginin	**Arg**	
Q	Glutamin	**Gln**	**N**	Asparagin	**Asn**	
E	Glutaminsäure	**Glu**	**D**	Asparaginsäure		**Asp**
S	Serin	**Ser**	**T**	Threonin	**Thr**	

Zusätzliche Codes

B	**Asn/Asp**	**Z**	**Gln/Glu**
X	Irgendeine Aminosäure		

Die Kenntnis dieser Abkürzungen ist essentiell für Sequenzalignments und für Proteinstrukturanalyse!

PRINTS



(http://bioinf.man.ac.uk/dbbrowser/PRINTS/)

- sekundäre Protein-Datenbank
- 2.156 Einträge und 12.444 Motive (in 2012)
- Fingerabdruck (*fingerprint*): Gruppe von konservierten Motiven
- mehrere funktionelle Bereiche (Faltung, Ligandenbindung, Komplexbildung, …) -> mehrere Sequenzmotive für ein Protein
- Motive aus kurzen lokalen Alignments

•Abstände zwischen Motiven und Reihenfolge spielen keine Rolle

•spezifisch für individuelle Proteine

•keine Zusammenfassung zu gemeinsamem Motiv

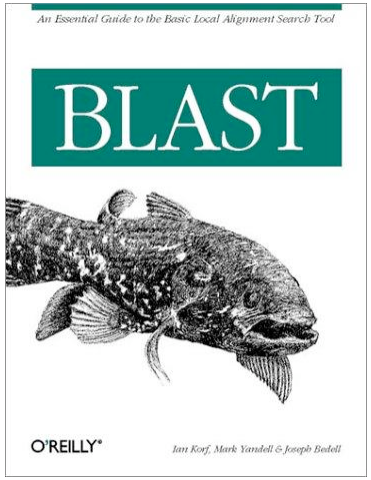
Pfam – Protein-Familien-Datenbank

(http://pfam.sanger.ac.uk/)

- sekundäre Protein-Datenbank
- 74% aller Proteinsequenzen haben mindestens einen Pfam-Eintrag
- Profile = funktionell interessante Domänen
- Profil: Auftrittswahrscheinlichkeiten bestimmter Aminosäuren an bestimmten Positionen in Form einer Matrix
- Pfam-A: genau untersuchte Profile aus multiplen Alignments, teilweise manuelle Alignments, >8000 Familien
- Pfam-B: automatisch generierte Profile: mehr Sequenzen, aber weniger präzise

V2 Paarweises Sequenzalignment

- Methoden des Sequenzalignments
- Austauschmatrizen
- Bedeutsamkeit von Alignments
- BLAST, Algorithmus – Parameter – Ausgabe http://www.ncbi.nih.gov



Diese Vorlesung lehnt sich eng an das BLAST Tutorial-Buch (links) an, Kapitel 3-9

Sequenz-Alignment

Wenn man 2 oder mehr Sequenzen vorliegen hat, möchte man zunächst einmal

- ihre Ähnlichkeiten quantitativ erfassen

  Die ähnlichen Regionen können hierbei die ganze Sequenz, oder Teile von ihr umfassen! Lokales Alignment $\leftrightarrow$ globales Alignment

- Entsprechungen zwischen einzelnen Bausteinen beider Sequenzen erfassen

- Gesetzmässigkeiten der Konservierung und Variabilität beobachten

- Rückschlüsse auf entwicklungsgeschichtliche **Verwandschaftsverhältnisse** ziehen

Wichtiges Ziel: **Annotation**, d.h. Zuordnung von strukturellen und funktionellen Merkmalen zu Gensequenzen.

Ähnlichkeit von Aminosäuren

Margaret Dayhoff
http://www.nlm.nih.gov/changingthefaceofmedicine/gallery/photo_76_7.html

Margaret Dayhoff stellte die Ähnlichkeit (beobachtete Austauschhäufigkeiten zwischen verwandten Sequenzen) zwischen Aminosäuren als $\log_2$ odds Verhältnis, oder *lod score* dar.

Lod score einer Aminosäure: nehme den Logarithmus zur Basis 2 ($\log_2$) von dem Verhältnis der beobachteten Häufigkeit für ein Paar durch die zufällig für das Paar erwartete Häufigkeit.

Lod score = 0 → beobachtete und erwartete Häufigkeiten sind gleich
> 0 → ein Austauschpaar tritt häufiger auf als zufällig erwartet
< 0 → unwahrscheinlicher Austausch

Allgemeine Formel für die Bewertung s_{ij} zweier Aminosäuren *i* und *j*.

$$s_{ij} = \log \frac{q_{ij}}{p_i p_j}$$

mit den individuellen Häufigkeiten p_i und p_j, und der Paarungsfrequenz q_{jj},

Ähnlichkeit der Aminosäuren

Beispiel: die relative Häufigkeiten von Methionin und Leucin seien 0.01 und 0.1.

Durch zufällige Paarung erwartet man 1/1000 Austauschpaare Met – Leu.

Wenn die beobachtete Paarungshäufigkeit 1/500 ist, ist das Verhältnis der Häufigkeiten 2/1.

Im Logarithmus zur Basis 2 ergibt sich ein *lod score* von +1 or 1 bit.

Wenn die Häufigkeit von Arginin 0.1 und die Paarung mit Leu die Häufigkeit 1/500 hat, dann ergibt sich ein *lod score* für ein Arg – Leu Paar von -2.322 bits.

Gewöhnlich berechnet man *nats*, multipliziert die Werte mit einem Skalierungsfaktur und rundet sie dann auf Integer Werte

→ **Austauschmatrizen** PAM und BLOSUM.

Diese ganzzahligen Werte (Integers) nennt man *raw scores.*

Bewertungs- oder Austausch-Matrizen

– dienen um die Qualität eines Alignments zu bewerten

– Für Protein/Protein Vergleiche:
eine *20 x 20* Matrix für die Wahrscheinlichkeit, mit der eine bestimmte Aminosäure gegen eine andere durch zufällige Mutationen ausgetauscht werden kann.

– Der Austausch von Aminosäuren ähnlichen Charakters (Ile, Leu) ist wahrscheinlicher (hat eine höhere Bewertung bzw. tritt häufiger in der Natur auf) als der von Aminosäuren unterschiedlichen Charakters (e.g. Ile, Asp).

– Matrizen werden als symmetrisch angenommen, besitzen also die Form einer Dreiecksmatrix.

Substitutions-Matrizen

Nicht alle Aminosäuren sind gleich

- Einige werden leichter ausgetauscht als andere
- Bestimmte Mutationen geschehen leichter als andere
- Einige Austausche bleiben länger erhalten als andere

Mutationen bevorzugen bestimmte Austausche

- Einige Aminosäuren besitzen ähnliche Codons (siehe Codon-Sonne)
- Diese werden eher durch Mutation der DNA mutiert

Selektion bevorzugt bestimmte Austausche

- Einige Aminosäuren besitzen ähnliche Eigenschaften und Struktur

PAM250 Matrix

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W
C	12																			
S	0	2																		
T	−2	1	3																	
P	−3	1	0	6																
A	−2	1	1	1	2															
G	−3	1	0	−1	1	5														
N	−4	1	0	−1	0	0	2													
D	−5	0	0	−1	0	1	2	4												
E	−5	0	0	−1	0	0	1	3	4											
Q	−5	−1	−1	0	0	−1	1	2	2	4										
H	−3	−1	−1	0	−1	−2	2	1	1	3	6									
R	−4	0	−1	0	−2	−3	0	−1	−1	1	2	6								
K	−5	0	0	−1	−1	−2	1	0	0	1	0	3	5							
M	−5	−2	−1	−2	−1	−3	−2	−3	−2	−1	−2	0	0	6						
I	−2	−1	0	−2	−1	−3	−2	−2	−2	−2	−2	−2	−2	2	5					
L	−6	−3	−2	−3	−2	−4	−3	−4	−3	−2	−2	−3	−3	4	2	6				
V	−2	−1	0	−1	0	−1	−2	−2	−2	−2	−2	−2	−2	2	4	2	4			
F	−4	−3	−3	−5	−4	−5	−4	−6	−5	−5	−2	−4	−5	0	1	2	−1	9		
Y	0	−3	−3	−5	−3	−5	−2	−4	−4	−4	0	−4	−4	−2	−1	−1	−2	7	10	
W	−8	−2	−5	−6	−6	−7	−4	−7	−7	−5	−3	2	−3	−4	−5	−2	−6	0	0	17

Beispiel für eine Bewertung

Wenn sich 2 Sequenzen in 2 (oder mehreren) Positionen unterscheiden,

möchte man die Wahrscheinlichkeit berechnen, daß Änderung A an Position 1 auftritt UND Änderung B an Position 2 (usw).

Man braucht also log (A × B), wobei das Malzeichen für die UND-Verknüpfung steht.

Es gilt allgemein log (A × B) = log A + log B

→ die Bewertung (Score) eines Alignments ist daher einfach die **Summe** aller Bewertungen für die Paare an Aminosäuren (Nukleinsäuren) des Alignments:

Sequenz 1: TCCPSIVARSN

Sequenz 2: SCCPSISARNT

1 12 12 6 2 5 -1 2 6 1 0 → Alignment Bewertung = 46

Dayhoff Matrix (1)

– wurde von Margaret.O. Dayhoff aufgestellt, die statistische Daten über die Austauschhäufigkeit von Aminosäuren in paarweisen Sequenzalignments sammelte

– Datensatz enthält eng verwandte Paare von Proteinsequenzen (> 85% Identität). Diese können nämlich zweifelsfrei aligniert werden.

– Aus der Frequenz, mit der Austausche auftreten, stellte sie die 20 x 20 Matrix für die Wahrscheinlichkeiten auf, mit der Mutationen eintreten.

– Diese Matrize heisst PAM 1. Ein evolutionärer Abstand von 1 PAM (point accepted mutation) bedeutet, dass es 1 Punktmutation pro 100 Residuen gibt, bzw. dass die beiden Sequenzen zu 99% identisch sind.

Dayhoff Matrix (2)

Aus PAM 1 kann man Matrizen für größere evolutionäre Entfernungen herstellen, indem man die Matrix **mehrfach mit sich selbst multipliziert.**

PAM250:

- 2,5 Mutationen pro Residue
- entspricht 20% Treffern zwischen zwei Sequenzen,
  d.h. man beobachtet Änderungen in 80% der Aminosäurepositionen.
- Dies ist die Default-Matrize in vielen Sequenzanalysepaketen.

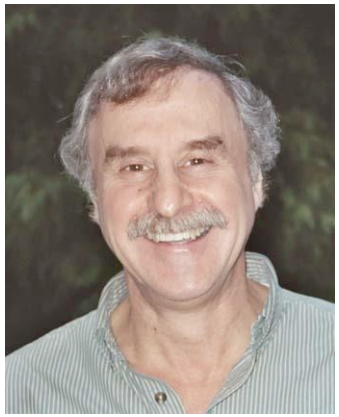
BLOSUM Matrix

Einschränkung der Dayhoff-Matrix:

Die Matrizen, die auf dem Dayhoff-Modell der evolutionären Raten basieren, sind von eingeschränktem Wert, da ihre Substitionsraten von Sequenzalignments abgeleitet wurden, die zu über 85% identisch sind.

S. Henikoff und J.G. Henikoff: verwendeten später lokale Multiple Alignments von entfernter verwandten Sequenzen

➔ **Blosum-Matrix**



Steven Henikoff

Dies war möglich, da es nun bereits mehr Sequenzen sowie Algorithmen für multiple Alignments gab.

Vorteile dieses Ansatzes:

- größere Datenmengen (es gibt mehr Sequenzen, die entfernt miteinander verwandt sind als nah verwandte)
- multiple Alignments sind robuster als paarweise Alignments

BLOSUM Matrix (2)

Die BLOSUM Matrizen (BLOcks SUbstitution Matrix) basieren auf der BLOCKS Datenbank.

Die BLOCKS Datenbank verwendet das Konzept von Blöcken (lückenlose Aminosäure-Signaturen), die charakteristisch für eine Proteinfamilie sind.

Aus den beobachteten Mutationen innerhalb dieser Blöcke wurden Austauschwahrscheinlichkeiten für alle Aminosäurepaare berechnet

und als Einträge für eine *log odds* BLOSUM matrix benutzt.

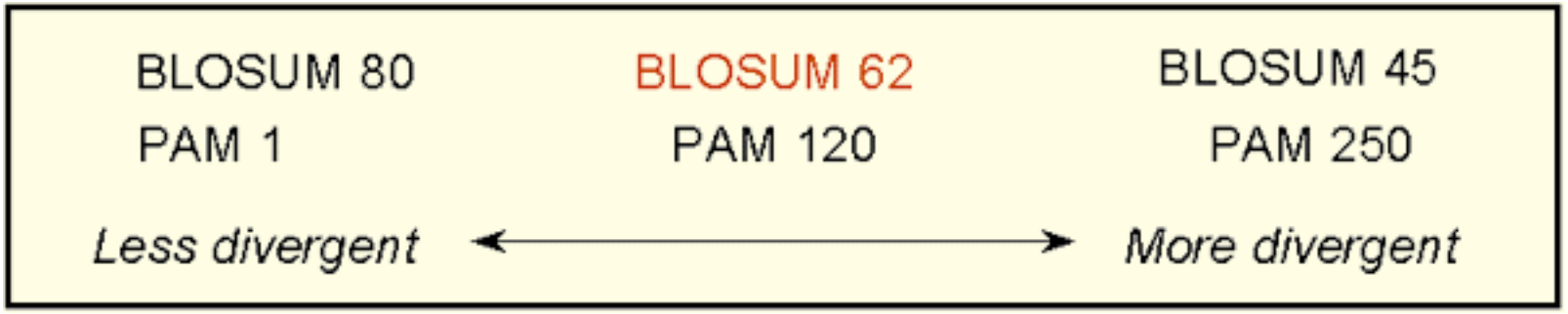
Man erhält unterschiedliche Matrizen indem man die untere Schranke des verlangten Grads an Identität variiert.

z.B. wurde die BLOSUM80 Matrix aus Blöcken mit > 80% Identität abgeleitet.

Welche Matrix soll man benutzen?

Enge Verwandtschaft (Niedrige PAM, hohe Blosum)

Entfernte Verwandtschaft (Hohe PAM, niedrige Blosum)



Vernünftige Default-Werte: PAM250, BLOSUM62

Gewichtung von Lücken (Gaps)

Neben der Substitutionsmatrix braucht man auch eine Methode zur Bewertung von Lücken.

Welche Bedeutung haben Insertionen und Deletionen im Verhältnis zu Substitutionen?

Unterscheide Einführung von Lücken:

```
aaagaaa
aaa-aaa
```

von der Erweiterung von Lücken:

```
aaaggggaaa
aaa----aaa
```

Verschiedene Programme (CLUSTAL-W, BLAST, FASTA) empfehlen unterschiedliche Default-Werte, die man wohl erst einmal verwenden sollte.

Needleman-Wunsch Algorithmus

- allgemeiner Algorithmus für Sequenzvergleiche

- maximiert eine Bewertung der Ähnlichkeit

- bester Match = größte Anzahl an Residuen einer Sequenz, die zu denen einer anderen Sequenz passen, wobei Deletionen erlaubt sind.

-Der Algorithmus findet durch dynamische Programmierung das bestmögliche GLOBALE Alignment zweier beliebiger Sequenzen

- NW beinhaltet eine iterative Matrizendarstellung

alle möglichen Residuenpaare (Basen oder Aminosäuren) – je eine von jeder Sequenz – werden in einem 2-dimensionalen Gitter dargestellt.

alle möglichen Alignments entsprechen Pfaden in diesem Gitter.

- Der Algorithmus hat 3 Schritte: 1 Initialisierung 2 Auffüllen 3 Trace-back

Needleman-Wunsch Algorithm: Initialisierung

Aufgabe: aligniere die Wörter “COELACANTH” und “PELICAN” der Länge $m = 10$ und $n = 7$. Konstruiere $(m + 1) \times (n + 1)$ Matrix.

Ordne den Elementen der ersten Zeile und Reihe die Werte $– m \times gap$ und $– n \times gap$ zu.

Die Pointer dieser Felder zeigen zurück zum Ursprung.

		C	O	E	L	A	C	A	N	T	H
	0	-1 ←	-2 ←	-3 ←	-4 ←	-5 ←	-6 ←	-7 ←	-8 ←	-9 ←	-10 ←
P	↑ -1										
E	↑ -2										
L	↑ -3										
I	↑ -4										
C	↑ -5										
A	↑ -6										
N	↑ -7										

Needleman-Wunsch Algorithm: Auffüllen

Fülle alle Matrizenfelder mit Werten und Zeigern mittels simpler Operationen, die die Werte der diagonalen, vertikalen, und horizontalen Nachbarzellen einschließen. Berechne

match score: Wert der Diagonalzelle links oben + Wert des Alignments (+1 oder -1)

horizontal gap score: Wert der linken Zelle + gap score (-1)

vertical gap score: Wert der oberen Zelle + gap score (-1).

Ordne der Zelle das Maximum dieser drei Werte zu. Der Pointer zeigt in Richtung des maximalen Werts.

		C	O	E	L	A	C	A	N	T	H
	0	-1 ←	-2 ←	-3 ←	-4 ←	-5 ←	-6 ←	-7 ←	-8 ←	-9 ←	-10 ←
P	↑ -1	↖ -1	↖ -2								

max(-1, -2, -2) = -1

max(-2, -2, -3) = -2

(Lege Konvention fest, damit Pointer bei gleichen Werten immer in eine bestimmte Richtung zeigen soll, z.B. entlang der Diagonalen.

Needleman-Wunsch Algorithmus: Trace-back

Trace-back ergibt das Alignment aus der Matrix.

Starte in Ecke rechts unten und folge den Pfeilen bis in die Ecke links oben.

```
COELACANTH
-PELICAN--
```

		C	O	E	L	A	C	A	N	T	H
	0	-1 ←	-2 ←	-3 ←	-4 ←	-5 ←	-6 ←	-7 ←	-8 ←	-9 ←	-10 ←
P	↑ -1	↖ -1	↖ -2	↖ -3	↖ -4	↖ -5	↖ -6	↖ -7	↖ -8	↖ -9	↖ -10
E	↑ -2	↖ -2	↖ -2	↖ -1	-2 ←	-3 ←	-4 ←	-5 ←	-6 ←	-7 ←	-8 ←
L	↑ -3	↖ -3	↖ -3	↑ -2	↖ 0	-1 ←	-2 ←	-3 ←	-4 ←	-5 ←	-6 ←
I	↑ -4	↖ -4	↑ -4	↑ -3	↑ -1	↖ -1	↖ -2	↖ -3	↖ -4	↖ -5	↖ -6
C	↑ -5	↖ -3	-4 ←	↑ -4	↑ -2	↖ -2	↖ 0	-1 ←	-2 ←	-3 ←	-4 ←
A	↑ -6	↑ -4	↖ -4	↖ -5	↑ -3	↖ -1	↑ -1	↖ 1	0 ←	-1 ←	-2 ←
N	↑ -7	↑ -5	↖ -5	↖ -5	↑ -4	↑ -2	↖ -2	↑ 0	↖ 2	1 ←	0 ←

BLAST – Basic Local Alignment Search Tool

• Findet das am besten bewertete lokale optimale Alignment einer Testsequenz mit allen Sequenzen einer Datenbank.

• Sehr schneller Algorithmus, 50 mal schneller als dynamische Programmierung.

• Kann verwendet werden um sehr große Datenbanken zu durchsuchen, da BLAST eine vor-indizierte Datenbank benutzt

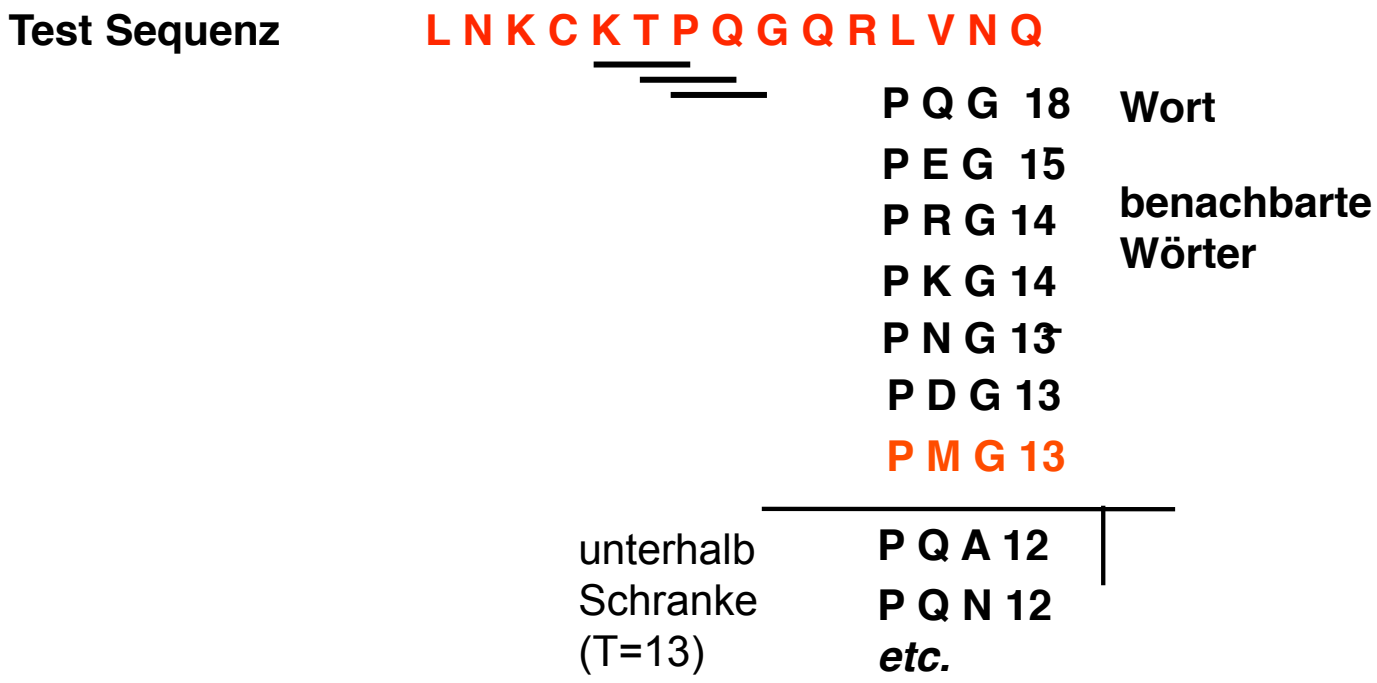
• Ist ausreichend sensititv und selektiv für die meisten Zwecke

• Ist robust – man kann üblicherweise die Default-Parameter verwenden

BLAST Algorithmus, Schritt 1

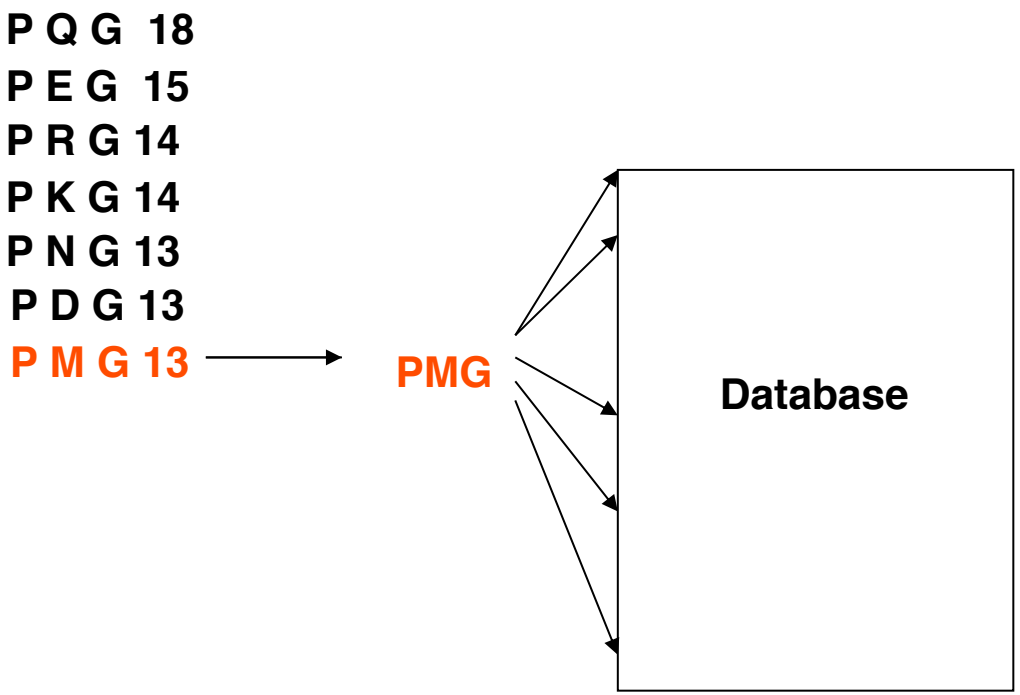
- Für ein gegebenes Wort der Länge *w* (gewöhnlich 3 für Proteine) und eine gegebene Bewertungs-Matrix

  erzeuge eine Liste aller Worte (*w*-mers), die eine Bewertung > *T* erhalten, wenn man sie mit dem *w*-mer der Eingabe vergleicht

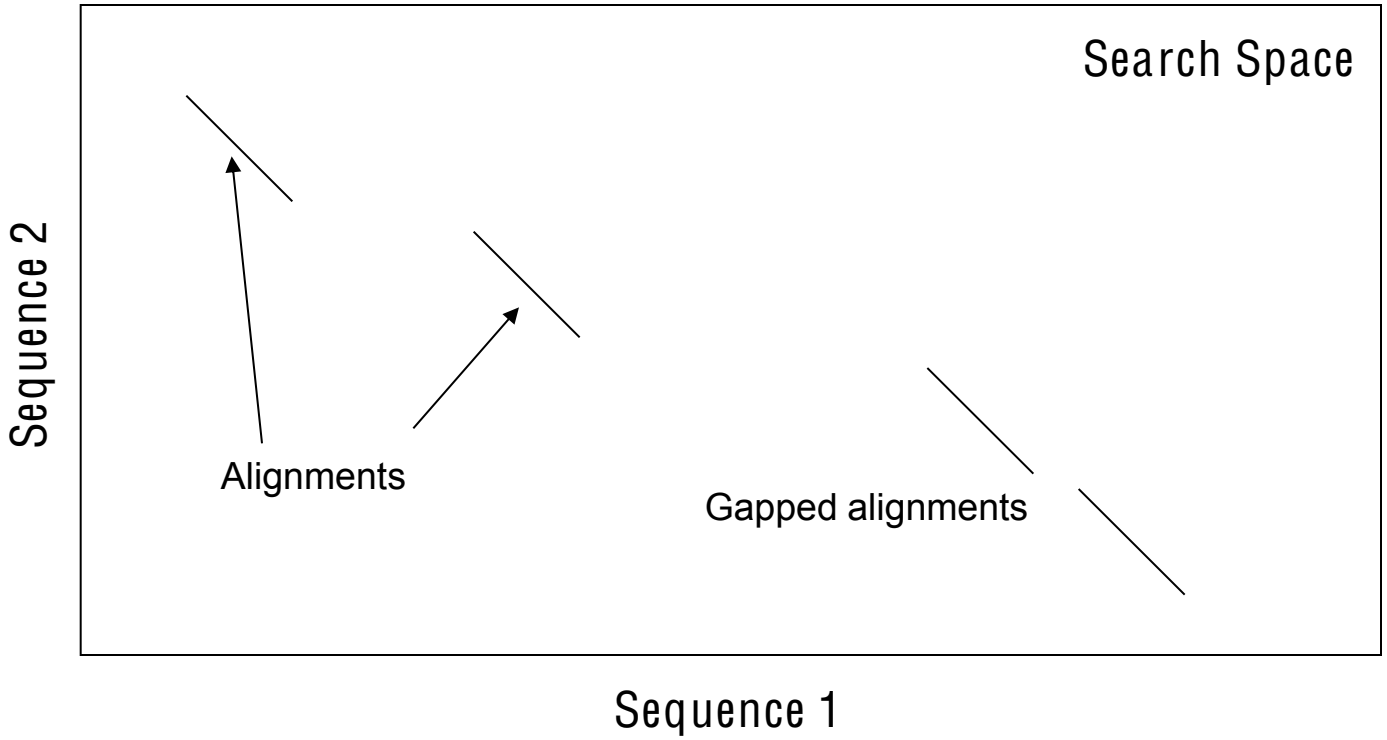


BLAST Algorithmus, Schritt 2

jedes benachbarte Wort ergibt alle Positionen in der Datenbank, in denen es gefunden wird (hit list).

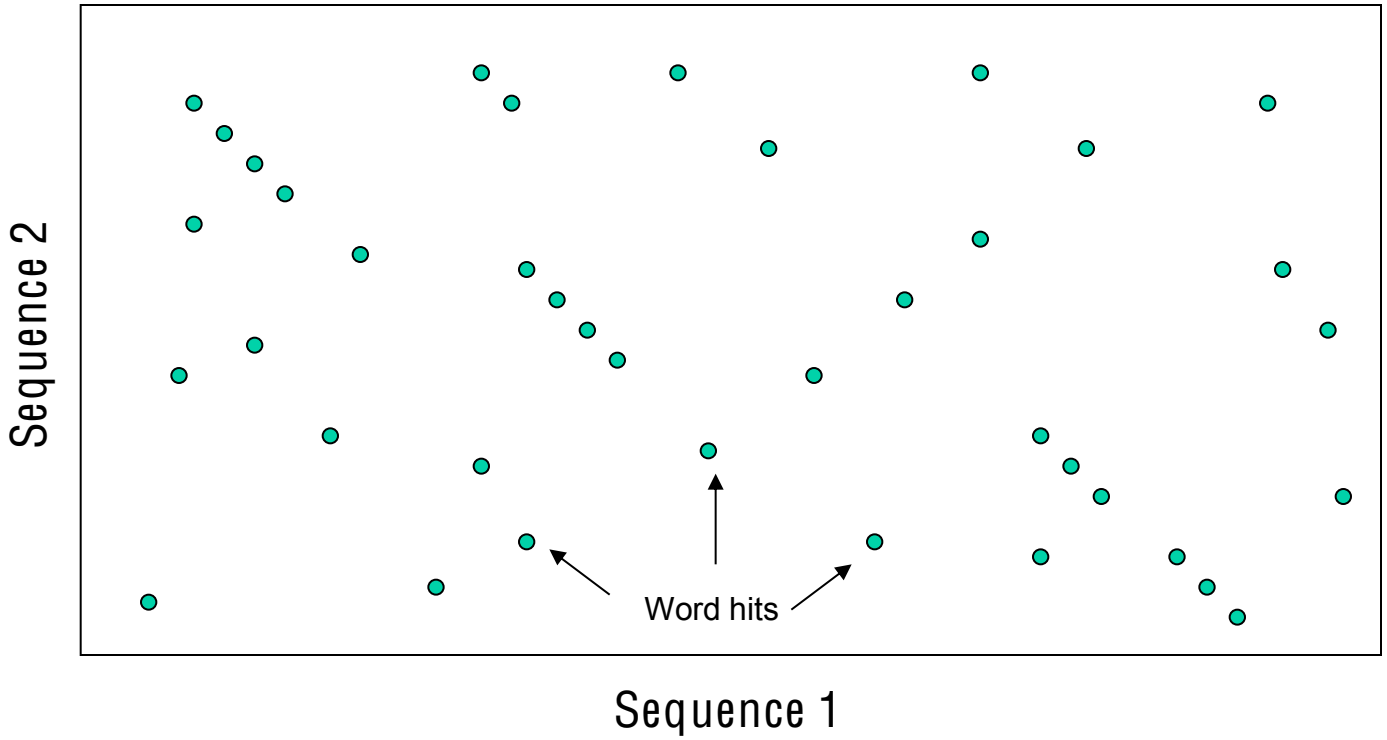


Was ist gesucht?



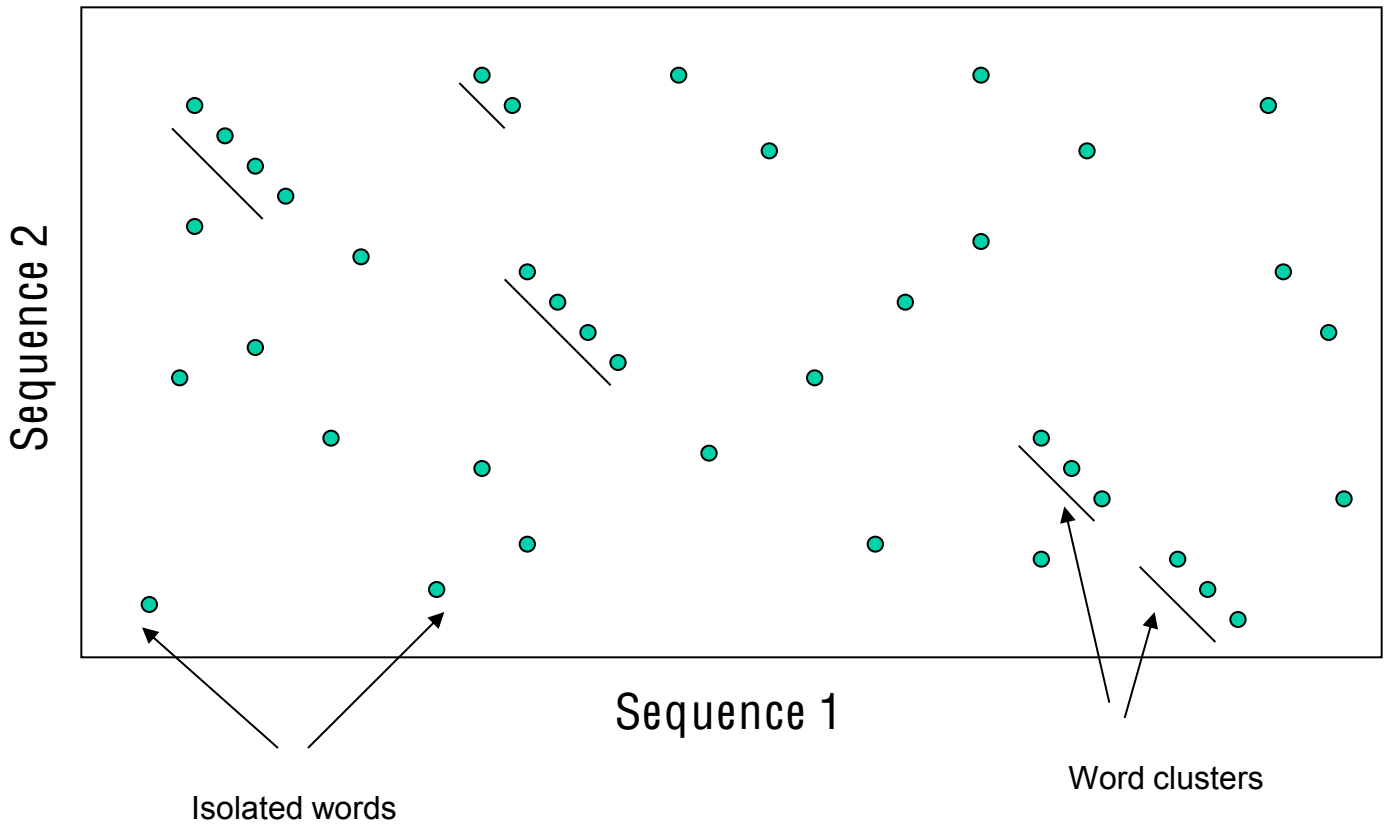
Das beste Mapping von Sequenz 1 auf Sequenz 2 entspricht einem unterbrochenen Pfad in dieser Diagonalmatrix.

Seeding



Können wir aus diesen „Word hits“ ein gutes Alignment konstruieren?

Seeding



BLAST Algorithmus: Erweiterungsschritt

- das Programm versucht, den Seed in beide Richtungen auszudehnen indem solange Residuenpaare hinzugefügt werden bis die zusätzliche Bewertung kleiner als ein Schrankenwert ist.

- Nachdem die Ausdehnung beendet wurde, wird das Alignment so “zurückbeschnitten” dass es die maximale Bewertung erhält.

High-scoring Segment Pair (HSP)

Nachbarschaft für 3-Buchstaben-Worte

BLOSUM62		PAM200	
Wort	Bewertung	Wort	Bewertung
RGD	17	RGD	18
KGD	14	RGE	17
QGD	13	RGN	16
RGE	13	KGD	15
EGD	12	RGQ	15
HGD	12	KGE	14
NGD	12	HGD	13
RGN	12	KGN	13
AGD	11	RAD	13
MGD	11	RGA	13
RAD	11	RGG	13
RGQ	11	RGH	13
RGS	11	RGK	13
RND	11	RGS	13
RSD	11	RGT	13
SGD	11	RSD	13
TGD	11	WGD	13

Kommentar:
Sowohl die Auswahl der Austauschmatrix wie die Wahl des Cut-offs T wird den Seeding-Schritt beeinflussen.

PSI-BLAST

“Position-Specific Iterated BLAST”

– Entfernte Verwandtschaften lassen sich besser durch Motiv- oder Profil-Suchen entdecken als durch paarweise Vergleiche

– PSI-BLAST führt zunächst eine BLAST-Suche mit Gaps durch.

– Das PSI-BLAST Programm verwendet die Information jedes signifikanten Alignments um eine positionsspezifische Substitionsmatrix zu konstruieren, die an Stelle der Eingabesequenz in der nächsten Runde der Datenbank-Suche verwendet wird.

– PSI-BLAST kann iterativ verwendet werden bis keine neuen signifikanten Alignments mehr gefunden werden.

BLAST Ausgabe (2)

Kleine Wahrscheinlichkeit deutet an, dass der Treffer wohl nicht zufällig zustande kam.

Sequences producing significant alignments:	Score (bits)	E Value
swissprot:CTRB_HUMAN Chymotrypsinogen B precursor (EC 3.4.21.1).	433	e-121
swissprot:CTR2_CANFA Chymotrypsinogen 2 precursor (EC 3.4.21.1).	386	e-107
swissprot:CTRB_RAT Chymotrypsinogen B precursor (EC 3.4.21.1).	383	e-106
swissprot:CTRB_BOVIN Chymotrypsinogen B (EC 3.4.21.1).	348	4e-96
swissprot:CTRA_BOVIN Chymotrypsinogen A (EC 3.4.21.1).	330	1e-90
swissprot:CTRA_GADMO Chymotrypsin A precursor (EC 3.4.21.1).	286	2e-77

BLAST Ausgabe (3)

swissprot:CO2_HUMAN Complement C2 precursor (EC 3.4.21.43) (C3/C...	55	1e-07
swissprot:CO2_MOUSE Complement C2 precursor (EC 3.4.21.43) (C3/C...	53	3e-07
swissprot:ACH2_LONAC Achelase II protease (EC 3.4.21.-).	52	1e-06
swissprot:GD_DROME Serine protease gd precursor (EC 3.4.21.-) (G...	46	4e-05
swissprot:ACRO_CAPHI Acrosin (EC 3.4.21.10) (Fragment).	39	0.009
swissprot:CTRP_PENMO Chymotrypsin (EC 3.4.21.1) (Fragment).	36	0.047
swissprot:VSPA_CERCE Cerastotin (EC 3.4.21.-) (Fragments).	35	0.098
swissprot:EL2B_HORSE Neutrophil elastase 2B (EC 3.4.21.-) (Prote...	35	0.13
swissprot:CERC_SCHMA Cercarial protease precursor (EC 3.4.21.-) ...	34	0.26
swissprot:EL2A_HORSE Neutrophil elastase 2A (EC 3.4.21.-) (Prote...	33	0.42
swissprot:HPT_RABIT Haptoglobin beta chain (Fragment).	31	1.4
swissprot:NMT1_ASPPA NMT1 protein homolog.	30	4.8

Niedrige Bewertungen mit hohen Wahrscheinlickeiten deuten an, dass dies wohl keine guten Treffer sind.

Karlin-Altschul Statistik: E-value

Karlin und Altschul leiteten die Bewertung der Signifikanz eines Alignments ab (hier ohne Herleitung):

$$E = kmne^{-\lambda S}$$

Die Anzahl an Alignments (*E*), die man während einer Suche in einer Sequenzdatenbank mit *n* Sequenzen mit einer *m* Buchstaben langen Suchsequenz zufällig erhält, ist eine Funktion der Größe des Suchraums *(m × n)*, der normalisierten Austauschbewertungen *(λS),* und einer Konstanten *(k)*.

Bedeutung des Alignments in BLAST

E-Wert (Erwartungswert)

- E = P × Anzahl der Sequenzen in Datenbank
- E entspricht der Anzahl an Alignments einer bestimmten Bewertung, die man zufällig in einer Sequenz-Datenbank dieser Grösse erwartet

  (wird z.B. für ein Sequenzalignment E=10 angegeben, erwartet man 10 zufällige Treffer mit der gleichen Bewertung).

  Dieses Alignment ist also nicht signifikant.
- Treffer werden in BLAST nur ausgegeben, wenn der E-Wert kleiner als eine vorgewählte Schranke ist.

Grobe Anhaltspunkte

E-Wert (Erwartungswert)

$E \leq 0{,}0001$	genaue Übereinstimmung
$0{,}0001 \leq E \leq 0{,}02$	Sequenzen vermutlich homolog
$0{,}02 \leq E \leq 1$	Homologie ist nicht auszuschließen
$E \geq 1$	man muss damit rechnen, dass diese gute Übereinstimmung Zufall ist.

BLAST Ausgabe (5)

```
>swissprot:VSP5_TRIMU Mucrofibrase 5 precursor (EC 3.4.21.-).
          Length = 257

 Score =  103 bits (280), Expect = 3e-22
 Identities = 74/232 (31%), Positives = 110/232 (46%), Gaps = 10/232 (4%)

Query: 34   IVNGEEAVPGTWPWQVTLQDRSGFHFCGGSLISEDWVVTAAHCGVRTSEILIAGEFDQGS 93
            I+ G+E       P+ V +       + CGG+LI+E+WV+TAAHC      EI +     +
Sbjct: 25   IIGGDECNINEHPFLVLVYYDD--YQCGGTLINEEWVLTAAHCNGENMEIYLGMHSKKVP 82

Query: 94   DEDNIQVLRIAKVFKQPKYSILTVNNDITLLKLASPARYSQTISAVCLPSVDDDAGSLCA 153
            ++D   + +    K F      +    N DI L++L   P R S   I+ + LPS      GS+C
Sbjct: 83   NKDRRRRVPKEKFFCDSSKNYTKWNKDIMLIRLNRPVRKSAHIAPLSLPSSPPSVGSVCR 142

Query: 154  TTGWGRTKYNANKSPDKLERAALPLLTNAECKRSW-GRRLTDVMICGA--ASGVSSCMGD 210
               GWG          PD     A + LL     C+ ++ G    T    +C       G  SC GD
Sbjct: 143  IMGWGTISPTKVTLPDVPRCANINLLDYEVCRAAYAGLPATSRTLCAGILEGGKDSCGGD 202

Query: 211  SGGPLVCQKDGAYTLVAIVSWASDTCS-ASSGGVYAKVTKIIPWVQKILSSN 261
            SGGPL+C   +G +     IVSW   D C+      G+Y   V    + W++ I++ N
Sbjct: 203  SGGPLIC--NGQFQ--GIVSWGGDPCAQPHEPGLYTNVFDHLDWIKGIIAGN 250
```

Zusammenfassung

Paarweises Sequenzalignment ist heute Routine, aber nicht trivial.

Mit **dynamischer Programmierung** (z.B. Smith-Waterman) findet man garantiert das Alignment mit optimaler Bewertung.
Vorsicht: die Bewertungsfunktion ist nur ein Modell der biologischen Evolution.

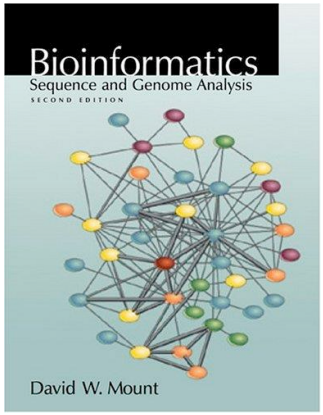
Die schnellste Alignmentmethode ist BLAST und seine Derivate wie BLAT.
Es ergibt sehr robuste und brauchbare Ergebnisse für Proteinsequenzen.

Multiple Sequenzalignments sind in der Lage, entferntere Ähnlichkeiten aufzuspüren und bieten ein besseres funktionelles Verständnis von Sequenzen und ihren Beziehungen

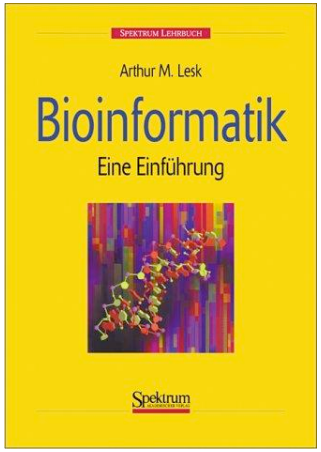
Kommt nächste Woche dran.

V3 - Multiples Sequenz Alignment und Phylogenie

Literatur: Kapitel 4 in Buch von David Mount



Thioredoxin-Beispiel heute aus Buch von Arthur Lesk



Leitfragen für V3

Frage1: Können wir aus dem Vergleich von Protein- (bzw. DNA-) Sequenzen etwas über evolutionäre Prozesse lernen?

Ansatz 1: vergleiche die Aminosäuresequenzen von homologen Proteinen aus verschiedenen Organismen und leite daraus phylogenetische Stammbäume ab (zweiter Teil der Vorlesung heute).

Methode: (1) suche homologe Proteine in verschiedenen Organismen (BLAST bzw. Psiblast) (2) führe multiples Sequenzalignment durch (erster Teil)

Ansatz 2: vergleiche die kompletten Genomsequenzen verschiedener Organismen (Breakpoint-Analyse) und leite daraus phylogenetische Stammbäume ab (wird hier nicht behandelt).

Leitfragen für V3

Frage2: Können wir aus den evolutionären Veränderungen in einer Proteinsequenz etwas über die Struktur und Funktion des Proteins lernen? (erster Teil)

Ansatz : führe multiples Sequenzalignment durch (erster Teil der Vorlesung)

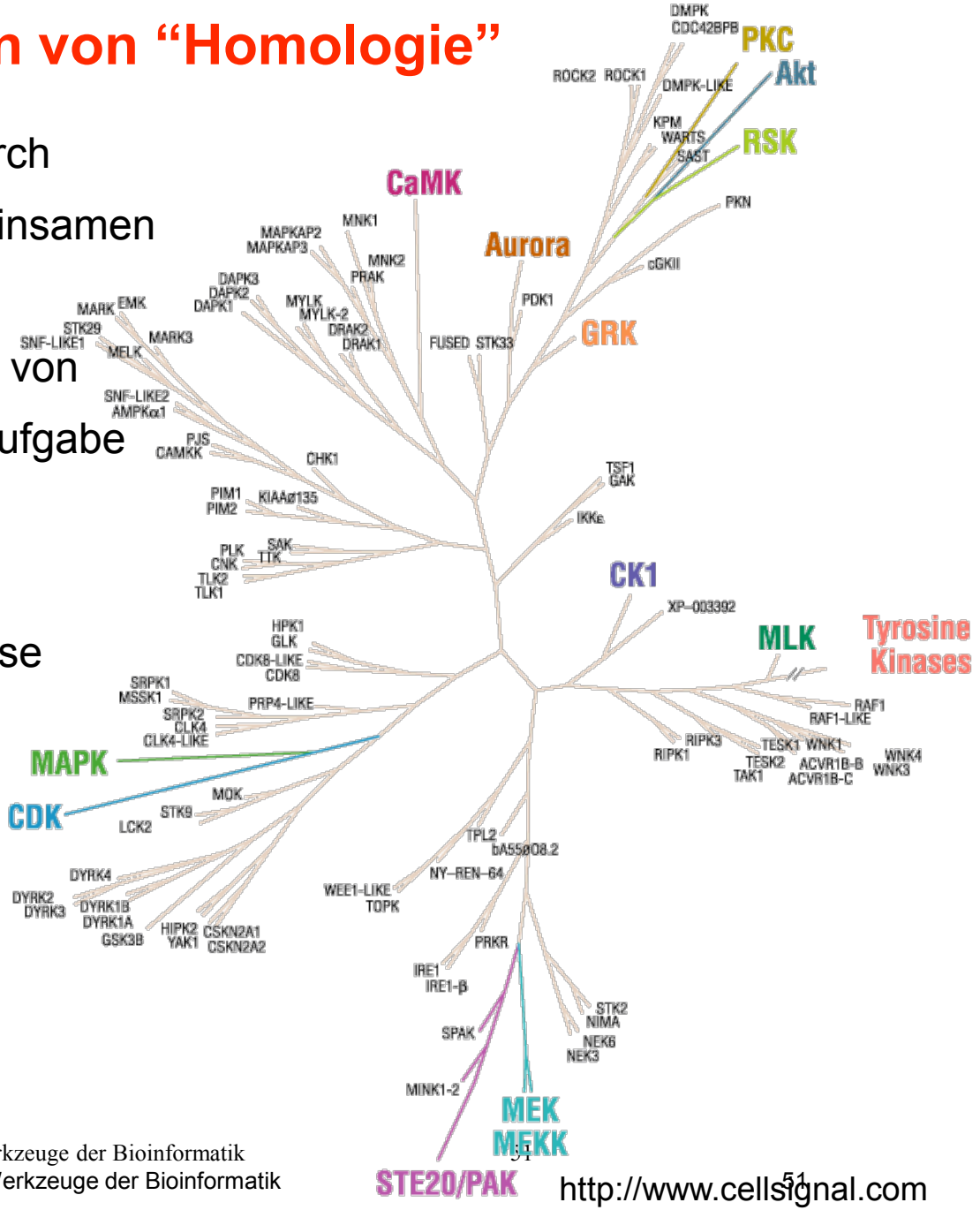
Exkurs: Evolution von Autos

- Welche Teile entsprechen dem aktiven Zentrum eines Proteins?
- Wird auch die Karosserie von Autos an Umgebungsbedingungen angepasst? (wo in Europa gibt es am meisten Cabrios?)
- Was entspricht dem Prozess der Proteinfaltung?
- Welchem Teil des Proteins entsprechen die Autotüren?

Definition von “Homologie”

- Homologie: Ähnlichkeit, die durch Abstammung von einem gemeinsamen Ursprungsgen herrührt – die Identifizierung und Analyse von Homologien ist eine zentrale Aufgabe der Phylogenie.

- Ein Alignment ist eine Hypothese für die positionelle Homologie zwischen Basenpaaren bzw. Aminosäuren.



Alignments können einfach oder schwer sein

```
GCGGCCCA TCAGGTACTT GGTGG
GCGGCCCA TCAGGTAGTT GGTGG
GCGTTCCA TCAGCTGGTT GGTGG
GCGTCCCA TCAGCTAGTT GGTGG
GCGGCGCA TTAGCTAGTT GGTGA
******** ********** *****
```

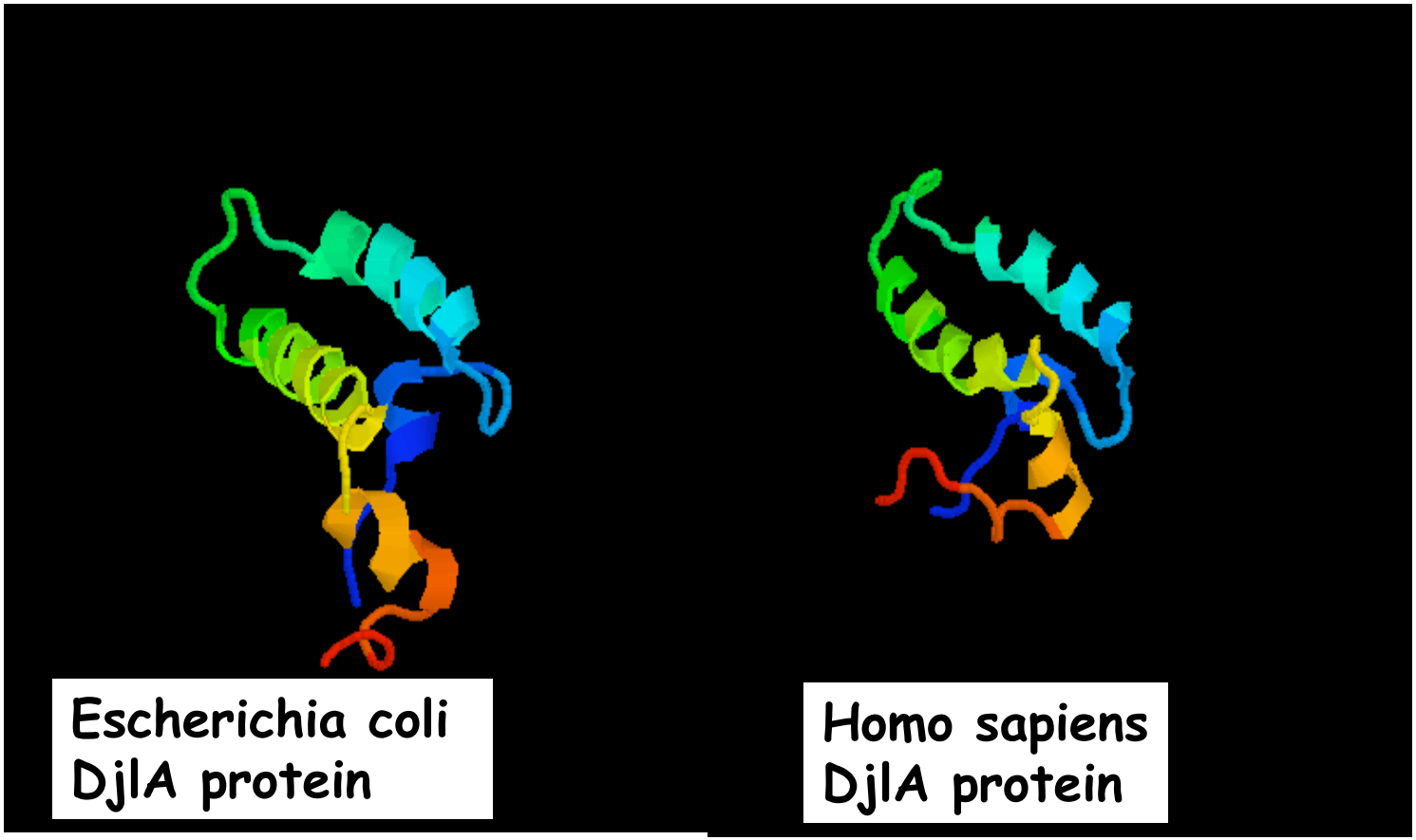
Einfach

```
TTGACATG CCGGGG---A AACCG
TTGACATG CCGGTG--GT AAGCC
TTGACATG -CTAGG---A ACGCG
TTGACATG -CTAGGGAAC ACGCG
TTGACATC -CTCTG---A ACGCG
******** ?????????? *****
```

Schwierig wegen Insertionen und Deletionen (indels)

Kann man beweisen, dass ein Alignment korrekt ist?

Protein-Alignment kann durch tertiäre Strukturinformationen geführt werden

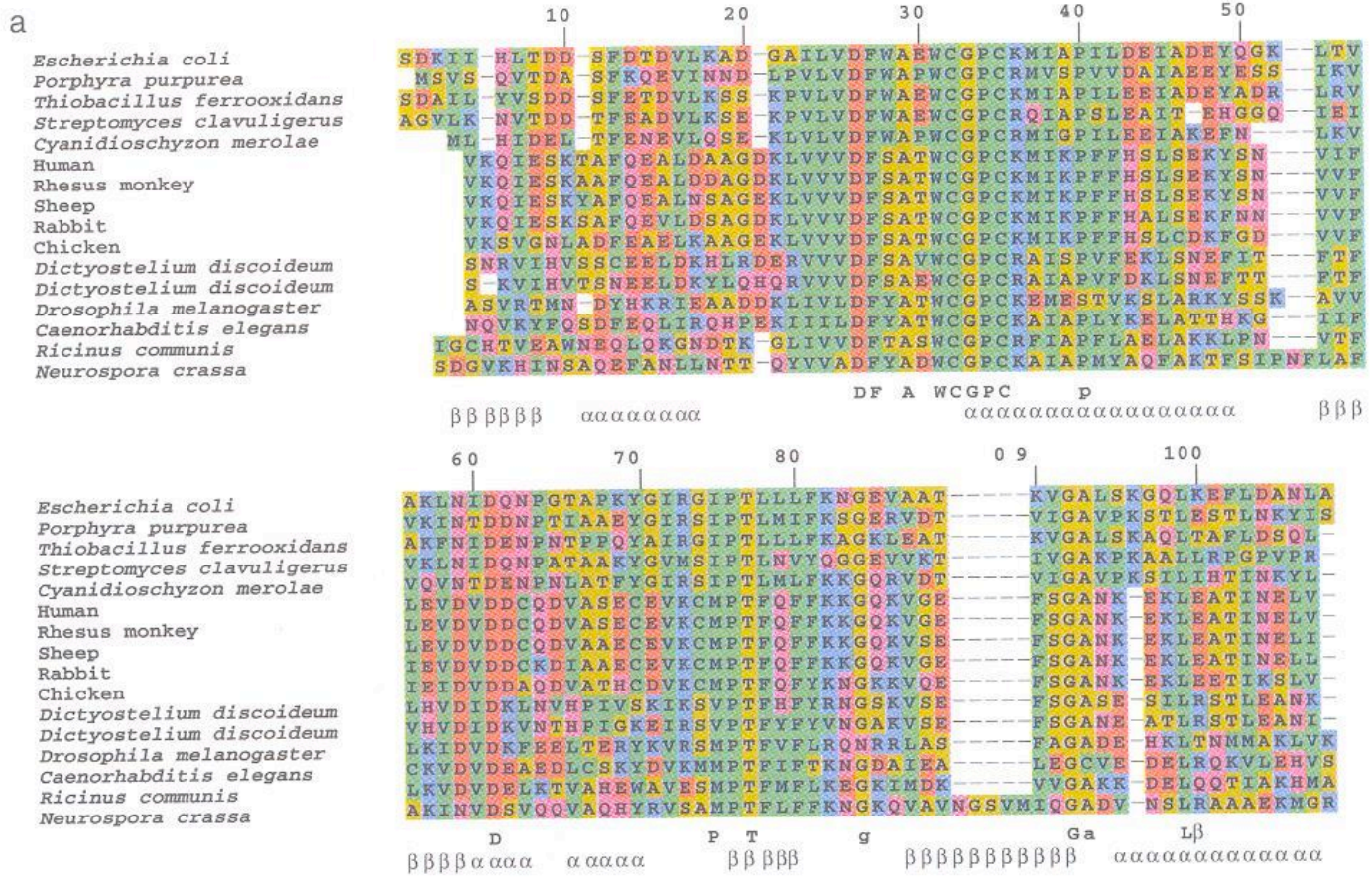


Gaps eines Alignments sollten vorwiegend in Loops liegen, nicht in Sekundär-struktur-elementen.

nur so kann man letztlich bewerten, ob ein Sequenzalignment korrekt ist. Beweisen im strikten Sinne kann man dies nie.

MSA für Thioredoxin-Familie

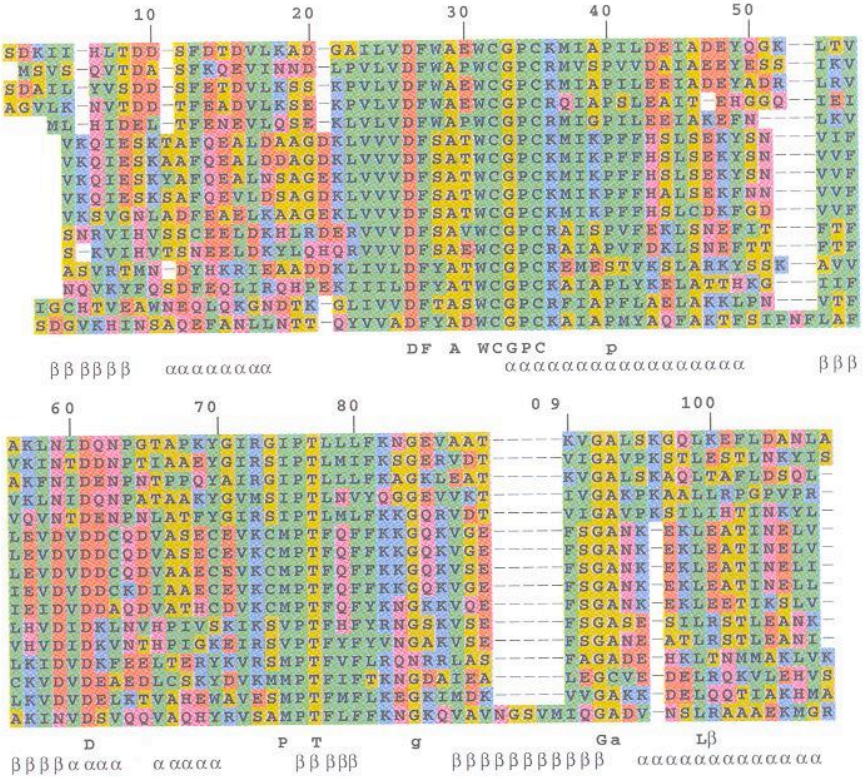
Farbe	Aminosäuretyp	Aminosäuren
gelb	klein, wenig polar	Gly, Ala, Ser, Thr
grün	hydrophob	Cys, Val, Ile, Leu
		Pro, Phe, Tyr, Met, Trp
violett	polar	Asn, Gln, His
rot	negativ geladen	Asp, Glu
blau	positiv geladen	Lys, Arg

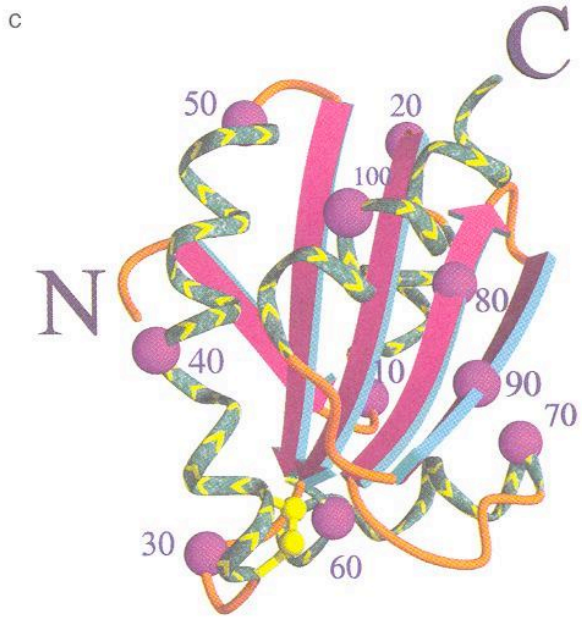


Infos aus MSA von Thioredoxin-Familie

Thioredoxin: aus 5 beta-Strängen bestehendes beta-Faltblatt,
das auf beiden Seiten von alpha-Helices flankiert ist.

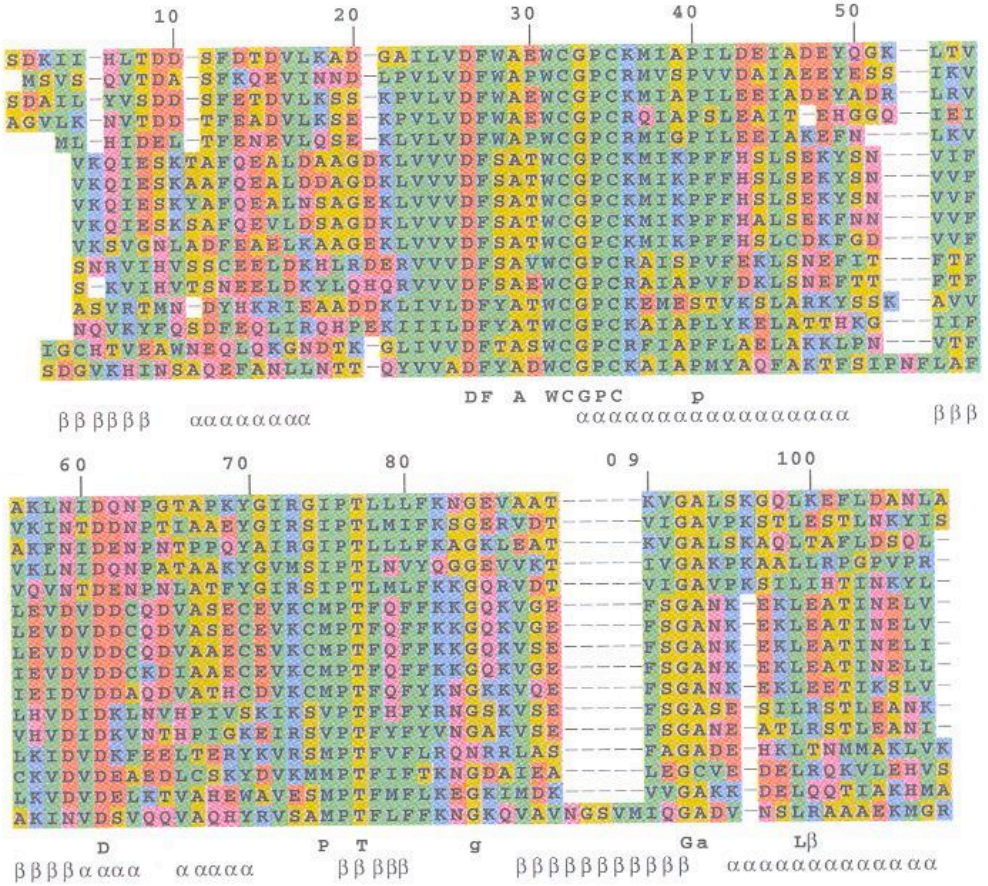
gemeinsamer Mechanismus: Reduktion von Disulfidbrücken in Proteinen

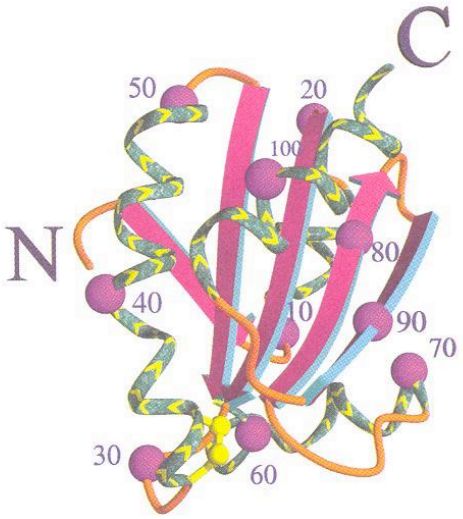




Infos aus MSA von Thioredoxin-Familie

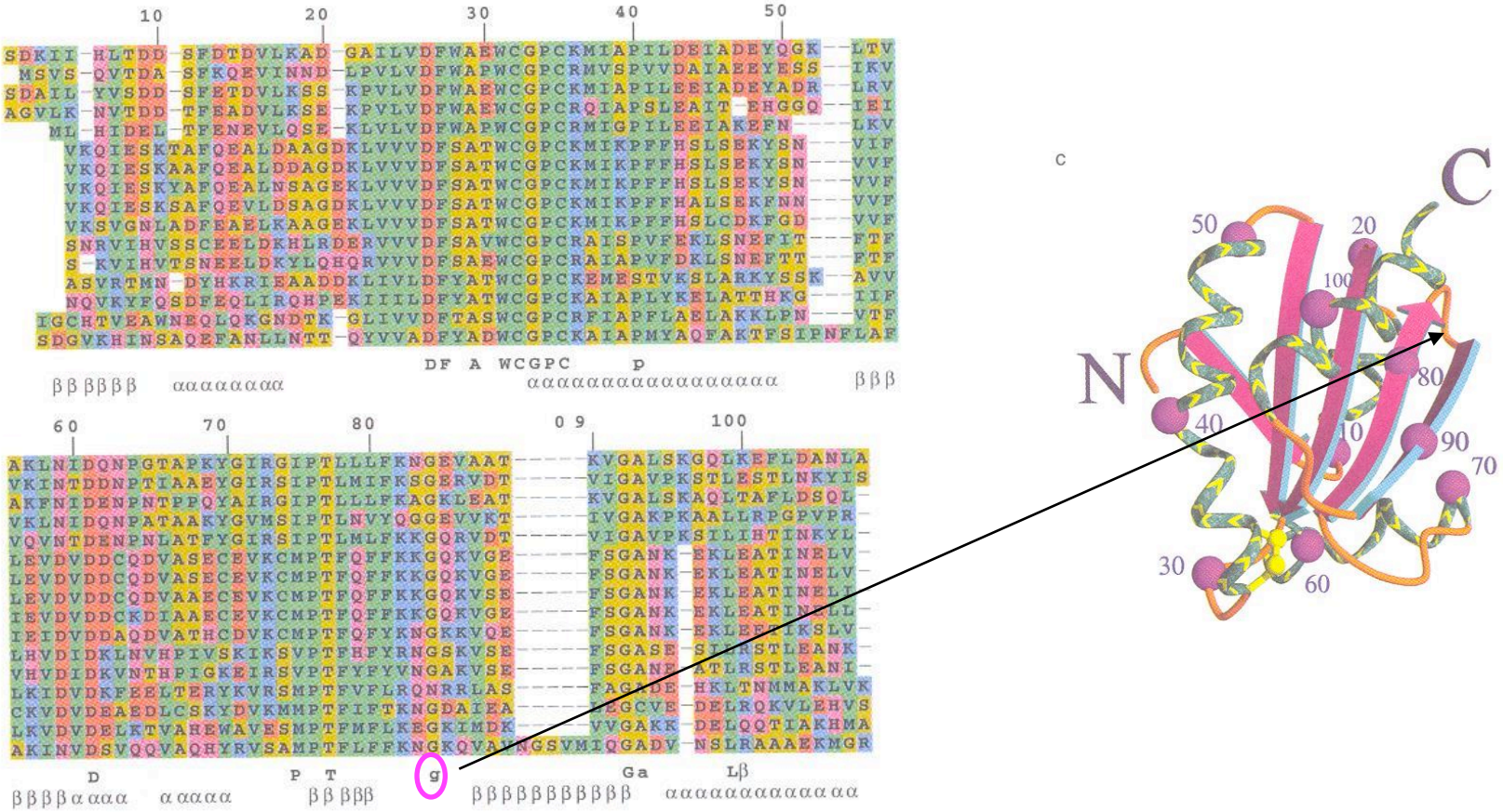
1) Die am stärksten konservierten Abschnitte entsprechen wahrscheinlich dem aktiven Zentrum. Disulfidbrücke zwischen Cys32 und Cys35 gehört zu dem konservierten WCGPC[K oder R] Motiv. Andere konservierte Sequenzabschnitte, z.B. Pro76Thr77 und Gly92Gly93 sind an der Substratbindung beteiligt.





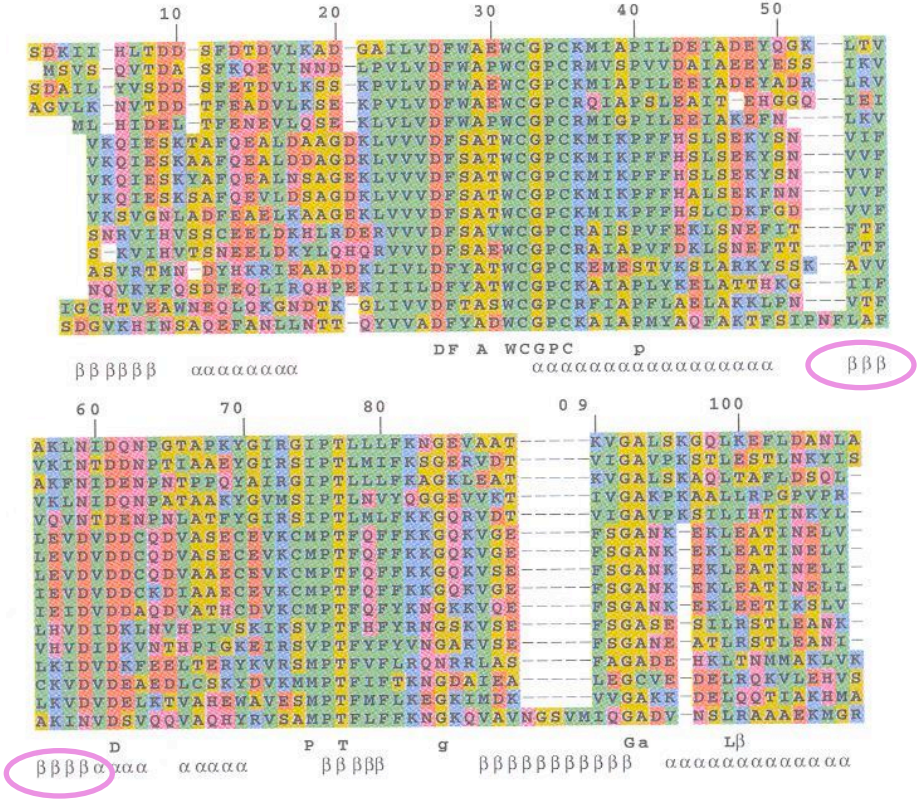
Infos aus MSA von Thioredoxin-Familie

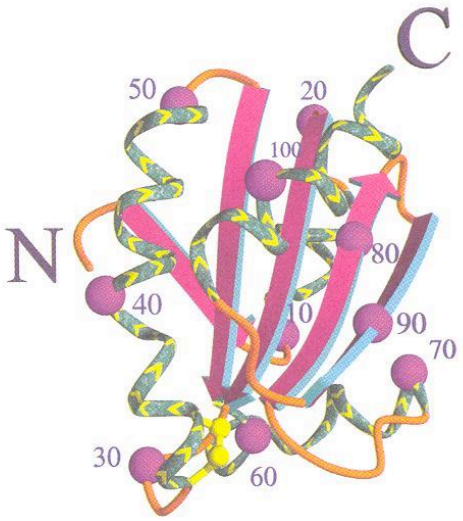
2) Abschnitte mit vielen Insertionen und Deletionen entsprechen vermutlich Schleifen an der Oberfläche. Eine Position mit einem konservierten Gly oder Pro lässt auf eine Wendung der Kette (‚turn') schließen.



Infos aus MSA von Thioredoxin-Familie

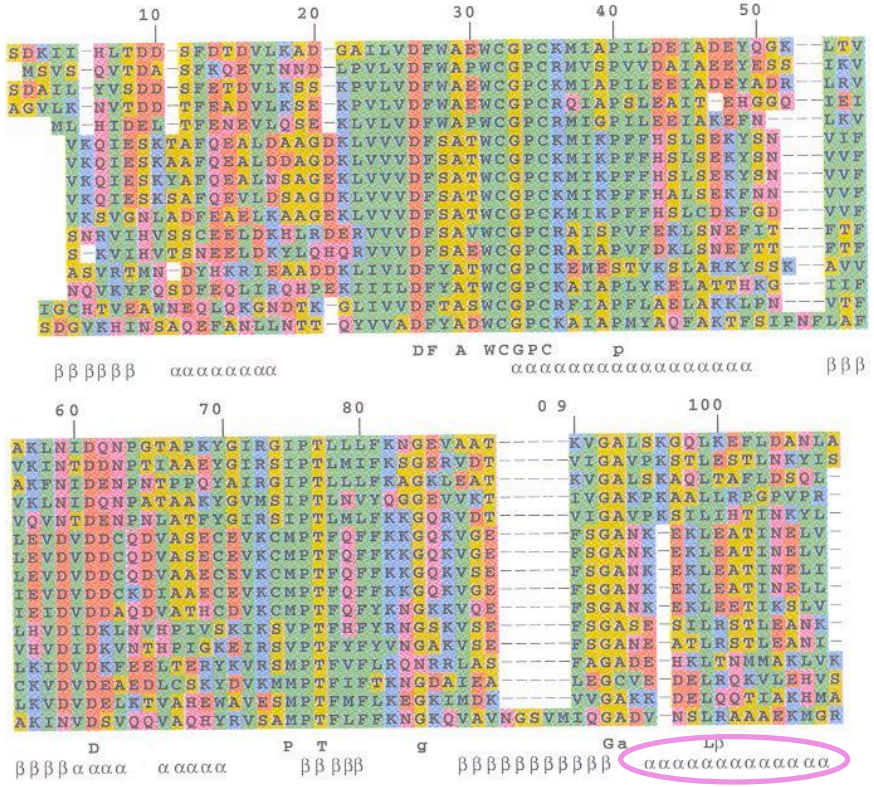
3) Ein konserviertes Muster hydrophober Bausteine mit dem Abstand 2 (d.h., an jeder zweiten Position), bei dem die dazwischen liegenden Bausteine vielfältiger sind und auch hydrophil sein können, lässt auf ein β-Faltblatt an der Moleküloberfläche schließen.

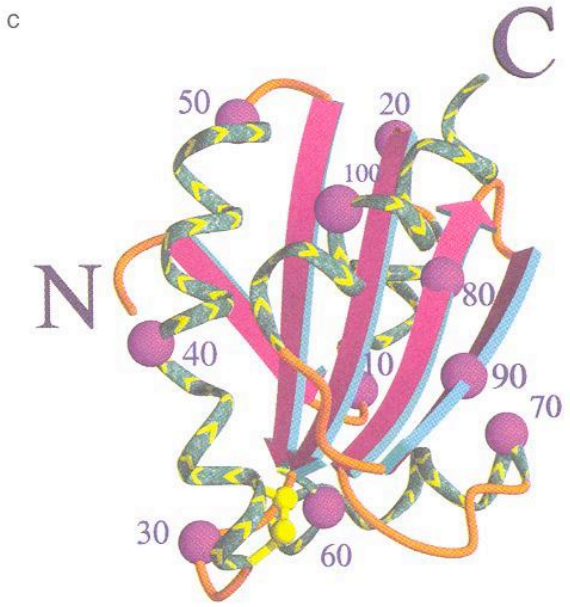




Infos aus MSA von Thioredoxin-Familie

4) Ein konserviertes Muster hydrophober Aminosäurereste mit dem Abstand von ungefähr 4 lässt auf eine α-Helix schließen.





Progressives Alignment

- wurde von Feng & Doolittle 1987 vorgestellt
- ist eine heuristische Methode.

  Daher ist nicht garantiert, das “optimale” Alignment zu finden.

- benötigt (*n-1) + (n-2) + (n-3) ... (n-n+1)* paarweise Sequenzalignments als Ausgangspunkt.

- weitverbreitete Implementation in Clustal (Des Higgins)

- ClustalW ist eine neuere Version, in der Gewichte (weights) verwendet werden.

ClustalW- Paarweise Alignments

- Berechne alle möglichen paarweisen Alignments von Sequenzpaaren.
  Es gibt (*n-1)+(n-2)...(n-n+1)* Möglichkeiten.

- Berechne aus diesen isolierten paarweisen Alignments den “Abstand” zwischen jedem Sequenzpaar.

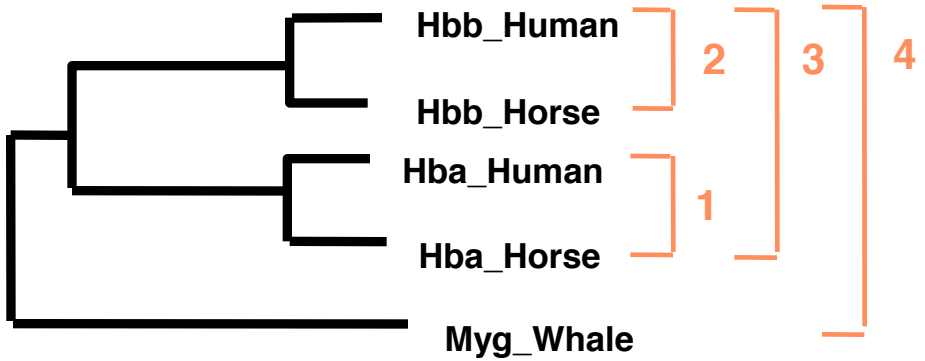
- Erstelle eine Abstandsmatrix.

- aus den paarweisen Distanzen wird ein Nachbarschafts-Baum erstellt

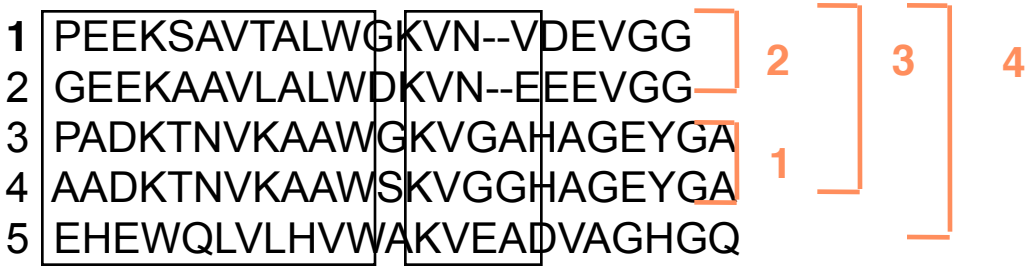
- Dieser Baum gibt die Reihenfolge an, in der das progressive Alignment ausgeführt werden wird.

Überblick der ClustalW Prozedur

Hbb_Human 1 -
Hbb_Horse 2 .17 -
Hba_Human 3 .59 .60 -
Hba_Horse 4 .59 .59 .13 -
Myg_Whale 5 .77 .77 .75 .75 -



alpha-helices



CLUSTAL W

↓

Schnelle paarweise Alignments:
berechne Matrix der Abstände

↓

Nachbar-Verbindungs-
Baumdiagramm

↓

progressive Alignments
entsprechend dem
Baumdiagramm

ClustalW- Vor- und Nachteile

Vorteil:

- Geschwindigkeit.

Nachteile:

- keine objektive Funktion.
- Keine Möglichkeit zu quantifizieren, ob Alignment gut oder schlecht ist (vgl. E-value für BLAST)
- Keine Möglichkeit festzustellen, ob das Alignment “korrekt” ist

Mögliche Probleme:

- Prozedur kann in ein lokales Minimum geraten.

  D.h. falls zu einem frühen Zeitpunkt ein Fehler im Alignment eingebaut wird, kann dieser später nicht mehr korrigiert werden, da die bereits alignierten Sequenzen fest bleiben.
- Zufälliges Alignment.

MSA mit MAFFT-Programm

Ziel: entdecke **lokale Verwandtschaft** zwischen zwei Sequenzen (homologe Segmente) durch Analyse der **Korrelation**.

Dies geht mit der **Fast Fourier Transformation** sehr schnell.

Allerdings braucht man dazu eine **numerische Darstellung** der beiden Sequenzen.

Annahme: evolutionär besonders wichtig sind das **Volumen** und die **Polarität** jeder Aminosäure.

Bilde daher zwei Vektoren der Länge *n*, die die Volumina und Polaritäten aller *n* Aminosäuren *k* enthalten.

MSA mit MAFFT-Programm

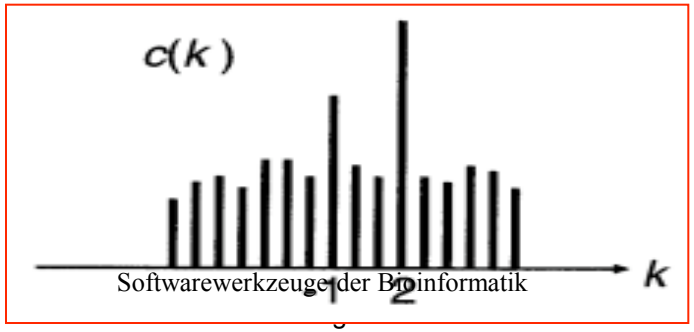
Berechne die Korrelation der beiden Vektoren v_1 , v_2 mit den Aminosäure-Volumina für jede mögliche Verschiebung k:

$$c_v(k) = \sum_{1 \le n \le N, 1 \le n+k \le M} \hat{v}_1(n)\hat{v}_2(n+k).$$

und analog die Korrelation der Vektoren mit den Aminosäure-Polaritäten.

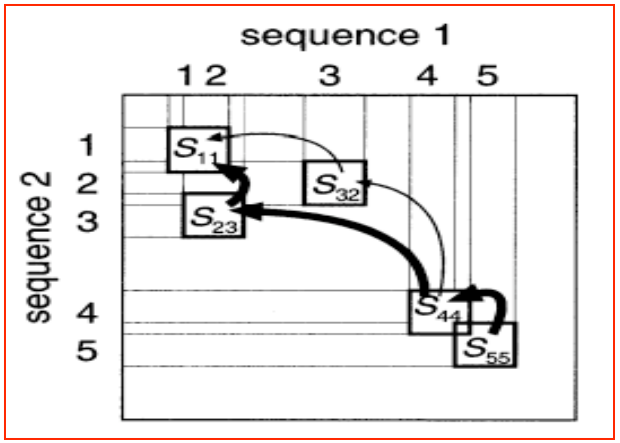
Bilde dann die Summe der beiden Korrelationen: $c(k) = c_v(k) + c_p(k).$

Schritt 1: Finde passende (d.h. möglicherweise homologe) Segmente mit maximaler Korrelation



MSA mit MAFFT-Programm

Schritt 2: Bilde paarweise Alignments mit eingeschränkter globaler dynamischer Programmierung:



Schritt 3: erstelle progressiv multiples Alignment:

- Schnelle Berechnung einer Distanzmatrix:

  → gruppiere 20 Aminosäuren in 6 physikochemische Gruppen

  → zähle 6-Tuples, die beide Sequenzen gemeinsam haben (vgl. Blast)
- konstruiere Baum mit UPGMA-Methode
- Baue multiples Alignment analog auf

Schritt 4: verfeinere MSA interativ durch Aufteilen des MSAs in 2 Bereiche und Re-Alignierung

Alignment von Protein-kodierenden DNS-Sequenzen

- Es macht wenig Sinn, proteinkodierende DNS-Abschnitte zu alignieren!

```
ATGCTGTTAGGG          ATGCT-GTTAGGG
ATGCTCGTAGGG    ⟶     ATGCTCGT-AGGG
```

Das Ergebnis kann sehr unplausibel sein und entspricht eventuell nicht dem biologischen Prozess.
Es ist viel sinnvoller, die Sequenzen in die entsprechenden Proteinsequenzen zu übersetzen, diese zu alignieren und dann in den DNS-Sequenzen an den Stellen Gaps einzufügen, an denen sie im Aminosäure-Alignment zu finden sind.

Zusammenfassung

Progressive Alignments sind die am weitesten verbreitete Methode für multiple Sequenzalignments.

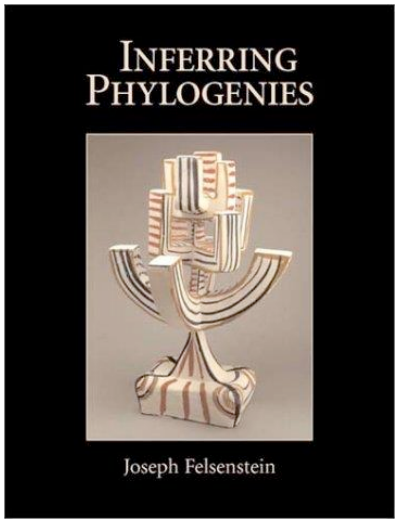
Sehr sensitive Methode ebenfalls: Hidden Markov Modelle (HMMer)

Multiples Sequenzalignment ist nicht trivial. Manuelle Nacharbeit kann in Einzelfällen das Alignment verbessern.

Multiples Sequenzalignment erlaubt Denken in Proteinfamilien und –funktionen.

Rekonstruiere Phylogenien aus einzelnen Gensequenzen

Material dieser Vorlesung aus
- Kapitel 6, DW Mount „Bioinformatics“
und aus Buch von Julian Felsenstein.



Eine **phylogenetische Analyse** einer Familie verwandter Nukleinsäure- oder Proteinsequenzen bestimmt, wie sich diese Familie durch Evolution entwickelt haben könnte.
Die evolutionären Beziehungen der Sequenzen können durch Darstellung als Blätter auf einem Baum veranschaulicht werden.

Phylogenien, oder evolutionäre Bäume, sind die Grundlage um Unterschiede zwischen Arten zu beschreiben und statistisch zu analysieren.
Es gibt sie seit über 140 Jahren und seit etwa 40 Jahren mit Hilfe von statistischen, algorithmischen und numerischen Verfahren.

3 Hauptansätze für Phylogenien einzelner Gene

- **maximale Parsimonie**

- **Distanzmatrix**

- **maximum likelihood** (wird hier nicht behandelt)

Häufig verwendete **Programme**:
PHYLIP (phylogenetic inference package – J Felsenstein)
PAUP (phylogenetic analysis using parsimony – Sinauer Assoc

Parsimonie Methoden

Edwards & Cavalli-Sforza (1963):
derjenige evolutionäre Baum ist zu bevorzugen,
der „den minimalen Anteil an Evolution“ enthält.

Luca Cavalli-Sforza

→ suche Phylogenien, die gerade so viele Zustandsänderungen beinhalten, wenn wir mit ihnen die evolutionären Vorgänge rekonstruieren, die zu den vorhandenen Daten (Sequenzen) führen.

(1) Für jede vorgeschlagene Phylogenie müssen wir in der Lage sein, die Vorgänge zu rekonstruieren, die am wenigsten Zustandsänderungen benötigen.

(2) Wir müssen unter allen möglichen Phylogenien nach denen suchen können, die eine minimale Anzahl an Zustandsänderungen beinhalten.

Finde den besten Baum durch heuristische Suche

Die naheliegende Methode, den Baum höchster Parsimonie zu finden ist,
ALLE möglichen Bäume zu betrachten und einzeln zu bewerten.

Leider ist die Anzahl an möglichen Bäumen üblicherweise zu groß.

→ verwende heuristische Suchmethoden, die versuchen,
die besten Bäume zu finden ohne alle möglichen Bäume zu betrachten.

(1) Konstruiere eine erste Abschätzung des Baums
und verfeinere diesen durch kleine Änderungen
= finde „benachbarte“ Bäume.

(2) Wenn irgendwelche dieser Nachbarn besser sind,
verwende diese und setze die Suche fort.

Zähle evolutionäre Zustandsänderungen als Modell für evolutionäre Kosten eines gegebenen Evolutionsbaums

Hierfür existieren zwei verwandte Algorithmen, die *dynamische Programmierung verwenden*: Fitch (1971) und Sankoff (1975)

- bewerte eine Phylogenie Buchstabe für Buchstabe

- betrachte jeden Buchstaben als Baum mit Wurzel an einem geeigneten Platz.

- propagiere eine Information nach unten durch den Baum;
beim Erreichen der Blätter ist die Anzahl der Zustandsänderungen bekannt.

Dabei werden die Zustandsänderungen oder internen Zuständen an den Knoten des Baums nicht konstruiert.

Sankoff Algorithmus

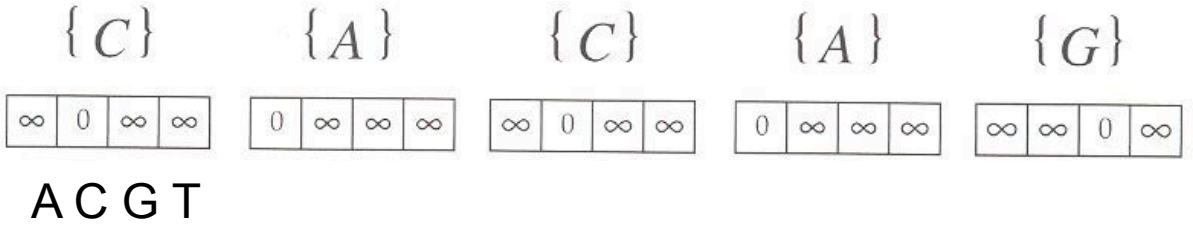
David Sankoff

Gesucht: Modell für Evolution einer Nukleotid-Position.

Konstruiere einen evolutionären Baum und wähle im unteren Endknoten (der zum Ur-Vorläufer gehört) den minimalen Wert, der die minimalen „evolutionären Kosten“ für diesen Buchstaben ausdrückt.

$$S = \min_i S_0(i)$$

Bekannt ist, welche Nukleotidbasen in den heutigen Sequenzen an dieser Position gefunden wird.
Daher ordnen wir an der Spitze des Baums jeder Sequenz die Kosten „0“ für die heute beobachtete Base zu und setzen die Kosten für die anderen 3 Basen auf Unendlich.



Nun brauchen wir einen Algorithmus, der die evolutionären Kosten *S(i)* für den jeweiligen Vorläufer zweier Knoten berechnet.

Sankoff-Algorithmus

Nenne die beiden Kind-Knoten *l* und *r* (für „links“ und „rechts“).

Die evolutionären Kosten für den direkten Vorgänger *a* (für „ancestor“) seien

$$S_a(i) = \min_j \left[c_{ij} + S_l(j) \right] + \min_k \left[c_{ik} + S_r(k) \right]$$

D.h. die geringst mögliche Kosten dafür, dass Knoten *a* den Zustand *i* hat, sind die Kosten c_{ij} um in der linken Vorgängerlinie vom Zustand *i* zum Zustand *j* zu gelangen plus die bis dahin bereits angefallenen Kosten $S_l(j)$.

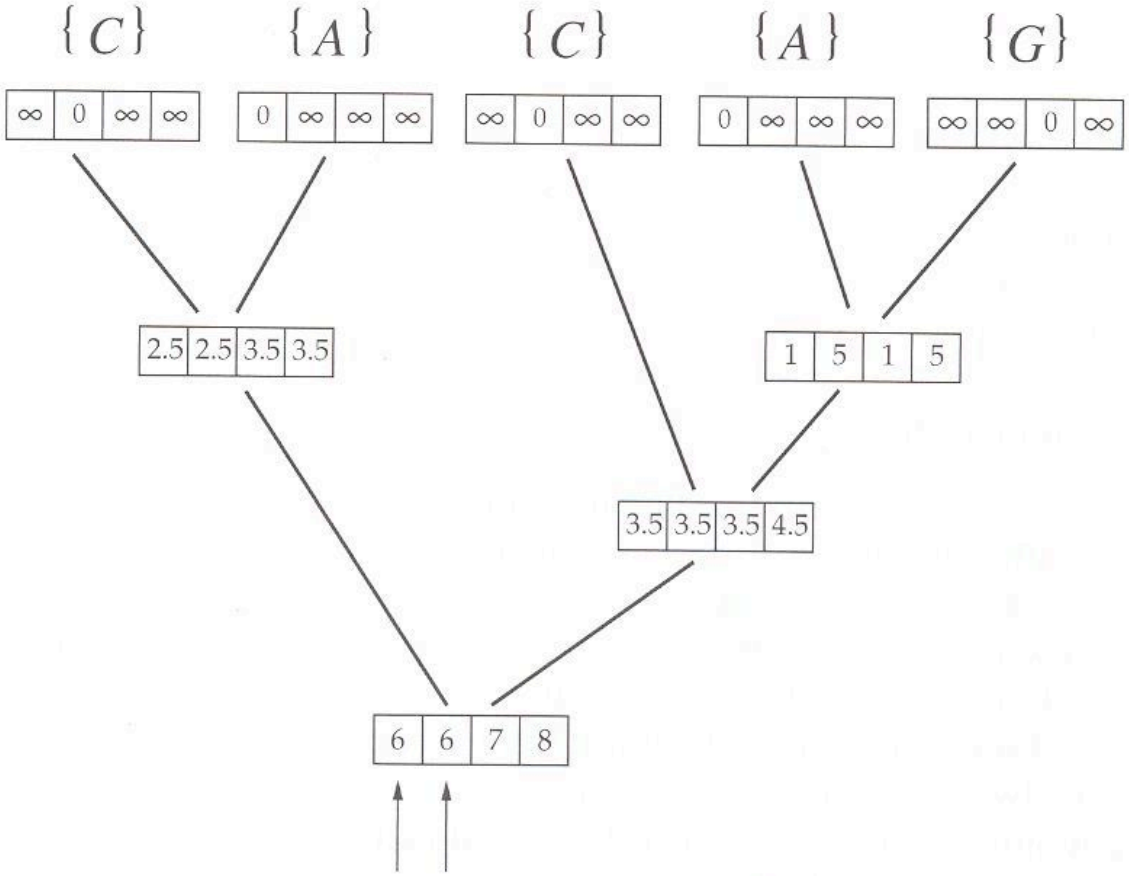
Wähle den Wert *j*, der diese Summe minimiert.

Entsprechende Berechnung für die rechte Vorgängerlinie, bilde Summe.

Wende diese Gleichung sukzessiv auf den ganzen Baum von oben nach unten an.

Berechne $S_0(i)$ und die minimalen Kosten für den Baum: $S = \min_i S_0(i)$

Sankoff-Algorithmus



Cost matrix:

from \ to	A	C	G	T
A	0	2.5	1	2.5
C	2.5	0	2.5	1
G	1	2.5	0	2.5
T	2.5	1	2.5	0

Der Vektor (6,6,7,8) an den Blättern besitzt ein Minimum von 6

= dies sind die minimalen Gesamtkosten dieses Baums für diesen Buchstaben.

Die Ur-Vorgängersequenz enthielt an dieser Position vermutlich „A“ oder „C“.

Konstruiere einen guten Baum: neighbor-joining Methode

durch Saitou und Nei (1987) eingeführt – der Algorithmus verwendet Clustering – eine molekulare Uhr wird nicht angenommen, aber das Modell minimaler Evolution.

„**Modell minimaler Evolution**“
wähle unter den möglichen Baumtopologien die mit minimaler Gesamtlänge der Äste.

Wenn die Distanzmatrix den Baum exakt abbildet, garantiert die Neighbor-joining Methode, als Methode der geringsten Quadrate, den optimalen Baum zu finden.

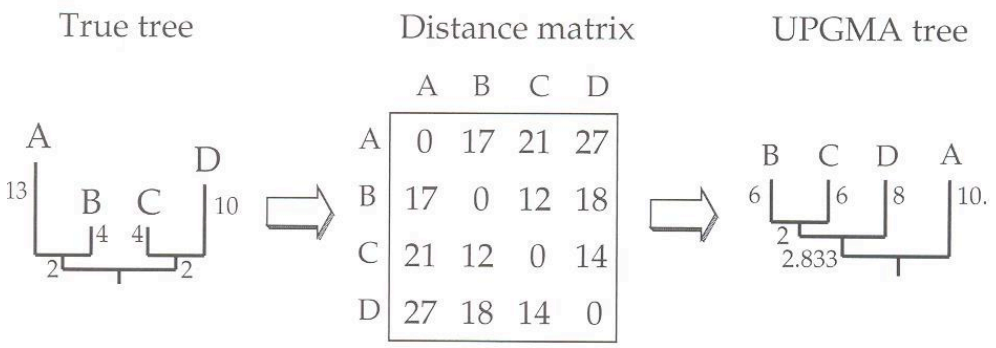


Figure 11.7: A four-species, nonclocklike tree and the expected data matrix it yields, when distances are the sums of branch lengths. The tree estimated by applying the UPGMA method to this distance matrix is shown—it does not have the correct tree topology. In both trees the branch lengths are proportional to the vertical length of the branches.

neighbor-joining Methode

(1) Berechne für jedes Blatt $u_i = \sum_{j \neq i}^{n} \frac{D_{ij}}{n-2}$

(2) Wähle *i* und *j* sodass $D_{ij} - u_i - u_j$ minimal ist.

(3) Verbinde *i* und *j*. Berechne die Astlängen von *i* zum neuen Knoten (v_i) und vom *j* zum neuen Knoten (v_j) als

$$v_i = \frac{1}{2} D_{ij} + \frac{1}{2}(u_i - u_j)$$

$$v_j = \frac{1}{2} D_{ij} + \frac{1}{2}(u_j - u_i)$$

(4) Berechne den Abstand zwischen dem neuen Knoten (*ij)* und den übrigen Blättern als

$$D_{(ij),k} = \frac{D_{ik} + D_{jk} - D_{ij}}{2}$$

(5) Lösche die Blätter *i* und *j* aus den Listen und ersetze sie durch den neuen Knoten, (*ij*), der nun als neues Blatt behandelt wird.

(6) Falls mehr als 2 Knoten übrig bleiben, gehe nach Schritt (1) zurück. Andernfalls verbinde die zwei verbleibenden Knoten (z.B. *l* und *m*) durch einen Ast der Länge D_{lm}.

Zusammenfassung

Multiple Sequenzalignments geben sehr wertvolle Einblicke in **Struktur und Funktion** von **Proteinfamilien**.

Globale dynamische Programmierung ist viel zu aufwändig.

Man benötigt heuristische Verfahren.

ClustalW: geleitet durch biologische Intuition; langsame Laufzeit.

Es gibt nun viel schnelle Verfahren z.B. MAFFT.

Die Rekonstruktion von phylogenetische Bäumen beruht auf multiplen Sequenz-alignments.

Die abgeleitete Phylogenie beruht stets auf Annahmen darüber, wie Evolution abläuft (z.B. minimale Parsimonie).

V4 – Analyse von Genomsequenzen

- **Gene identifizieren**

  Intrinsische und Extrinsische Verfahren:

  Homologie bzw. Hidden Markov Modelle

- **Transkriptionsfaktorbindestellen** identifizieren

  Position Specific Scoring Matrices (PSSM)

- Ganz kurz: finde **Repeat-Sequenzen**

  Suche nach bekannten Repeat-Motiven

- **Mapping** von **NGS-Daten** auf **Referenzgenom**

- **Alignment zweier Genom-Sequenzen**

  Suffix Bäume

Leitfragen für V4

Frage1: Wie können wir funktionell wichtige Bereiche in Genom-sequenzen finden?

Ansatz: leite aus bekannten Genen bzw. Transkriptionsfaktorbindestellen allgemeine Prinzipen ab und verwende diese dann zur Vorhersage.

Frage2: Wie können wir funktionell entsprechende Bereiche in anderen Genomsequenzen finden?

Ansatz: finde homologe, nur einmal vorkommende Bereiche in beiden Genomen als Ankerpunkte für das Genom-Alignment.

Identifikation von Genen

Die **einfachste** Methode, DNA Sequenzen zu finden, die für Proteine kodieren, ist nach **offenen Leserahmen** (**open reading frames** oder ORFs) zu suchen.

In jeder Sequenz gibt es 6 mögliche offene Leserahmen:
3 ORFs starten an den Positionen 1, 2, und 3 und gehen in die 5‘ 3‘ Richtung,
3 ORFs starten an den Positionen 1, 2, und 3 und gehen in die 5‘ 3‘ Richtung des komplementären Strangs.

In prokaryotischen Genomen werden Protein-kodierende DNA-Sequenzen gewöhnlich in mRNA transkribiert und die mRNA wird ohne wesentliche Änderungen direkt in einen Aminosäurestrang übersetzt.

Daher ist der längste ORF von dem ersten verfügbaren Met codon (**AUG**) auf der mRNA, das als **Codon** für den **Transkriptionsstart** fungiert, bis zu dem **nächsten Stopcodon** in demselben offenen Leserahmen, gewöhnlich eine gute Vorhersage für die Protein-kodierende Region.

Extrinsische und intrinsische Methoden

Viele Verfahren kombinieren nun

(a) Homologie-Methoden = „**extrinsische Methoden**“ mit

(b) Genvorhersage-Methoden = „**intrinsische Methoden**“

Etwa die Hälfte aller Gene kann durch Homologie zu anderen bekannten Genen oder Proteinen gefunden werden. Dieser Anteil wächst stetig, da die Anzahl an sequenzierten Genomen und bekannten cDNA/EST Sequenzen kontinuierlich wächst.

Um die übrige Hälfte an Genen zu finden, muss man Vorhersage-Methoden einsetzen.

Table 1: Availability of gene prediction software

Program	URL	Web interface	Download	Source
Geneid	http://www1.imim.es/software/geneid/index.html	yes	yes	yes[a]
GlimmerM	http://www.tigr.org/softlab/glimmerm/	yes	yes[b]	yes[b]
GenScan	http://genes.mit.edu/GENSCAN.html	yes	yes[b]	no
GenomeScan	http/genes.mit.edu.genomescan/	yes	no	no
MHHgene	http://www.cbs.dtu.dk/services/HMMgene/	yes	no	no
FGENES	http://genomic.sanger.ac.uk/gf/gf.shtml	yes	no	no
Genie	http://www.cse.ucsc.edu/~dkulp/cgi-bin.genie	yes[c]	no	no

[a] Free to all under GNU licence.
[b] Free only to academic users. Commercial users must purchase a licence.
[c] Web interface to Genie uses an older version of this program, not that one referred to in publications of fruit fly and human genomes or the GASP.

Mathé et al. Nucl. Acids. Res. 30, 4103 (2002)

Positions-spezifische Gewichtsmatrix

Populäres Verfahren wenn es eine Liste von Genen gibt, die ein TF-Bindungs-motiv gemeinsam haben. Bedingung: gute MSAs müssen vorhanden sein.

Alignment-Matrix: wie häufig treten die verschiedenen Buchstaben an jeder Position im Alignment auf?

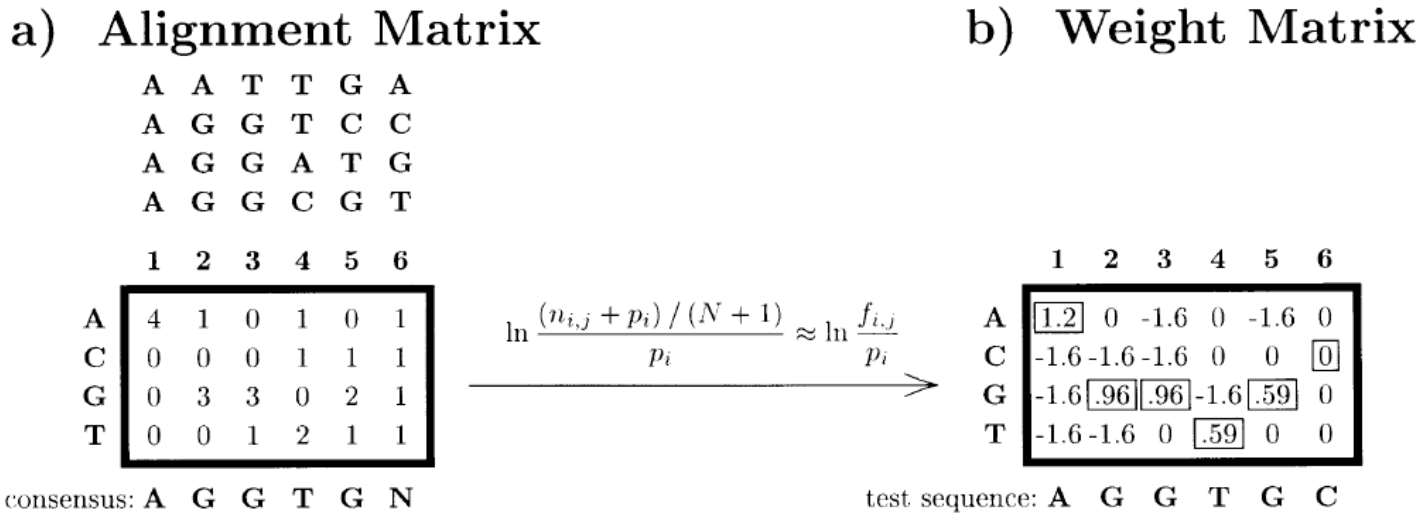


Fig. 1. Examples of the simple matrix model for summarizing a DNA alignment. (**a**) An alignment matrix describing the alignment of the four 6-mers on top. The matrix contains the number of times, $n_{i,j}$, that letter i is observed at position j of this alignment. Below the matrix is the consensus sequence corresponding to the alignment (N indicates that there is no nucleotide preference). (**b**) A weight matrix derived from the alignment in (a). The formula used for transforming the alignment matrix to a weight matrix is shown above the arrow. In this formula, N is the total number of sequences (four in this example), p_i is the *a priori* probability of letter i (0.25 for all the bases in this example) and $f_{i,j} = n_{i,j}/N$ is the frequency of letter i at position j. The numbers enclosed in blocks are summed to give the overall score of the test sequence. The overall score is 4.3, which is also the maximum possible score with this weight matrix.

Hertz, Stormo (1999) Bioinformatics 15, 563

Zusammenfassung

Es gibt große Datenbanken (z.B. TRANSFAC) mit Informationen über Promoterstellen. Diese Informationen sind experimentell überprüft.

Microarray-Daten erlauben es, nach gemeinsamen Motiven von ko-regulierten Genen zu suchen.

Auch möglich: gemeinsame Annotation in der Gene Ontology etc.

TF-Bindungsmotive sind oft überrepräsentiert in der 1000 bp-Region upstream. Die klare Funktion dieser Bindungsmotive ist oft unbekannt.

Allgemein gilt:
- relativ wenige TFs regulieren eine große Anzahl an Genen
- es gibt globale und lokale TFs
- Gene werden üblicherweise durch mehr als einen TF reguliert

http://www.gene-regulation.com

Was ist MUMmer?

- A.L. Delcher *et al*. 1999, 2002 Nucleic Acids Res.
- http://www.tigr.org/tigr-scripts/CMR2/webmum/mumplot

- nimm an, dass zwei Sequenzen eng verwandt sind (sehr ähnlich)
- MUMmer kann zwei bakterielle Genome in weniger als 1 Minute alignieren
- nutzt **Suffix-Bäume** um Maximal Unique Matches zu finden
- Definition eines Maximal Unique Matches (MUM):
  - Eine Subsequenz, die in beiden Sequenzen genau einmal ohne Abweichungen vorkommt und in keine Richtung verlängert werden kann.
- Grundidee: ein MUM ausreichender Länge wird sicher Teil eines globalen Alignments sein.

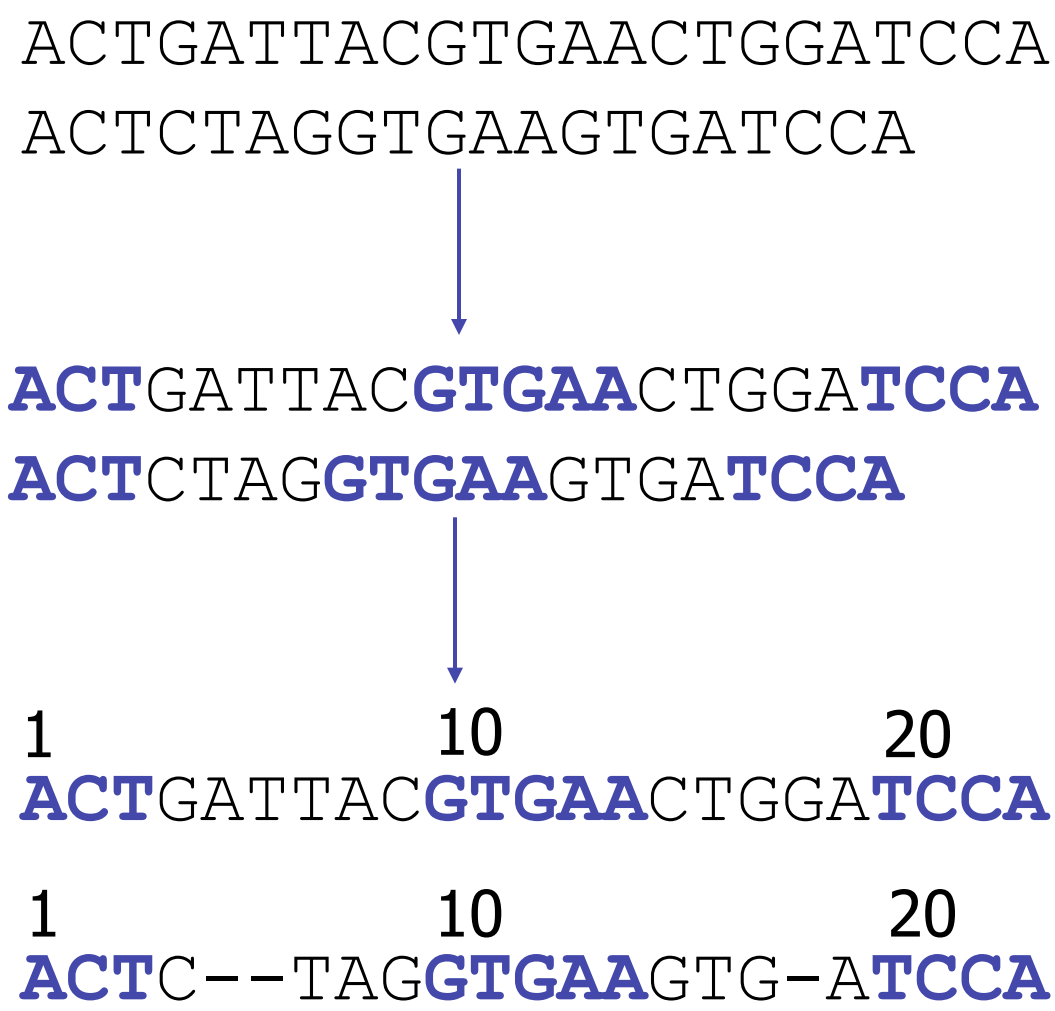
Genome A: tcgatcGACGATCGCGGCCGTAGATCGAATAACGAGAGAGCATAAcgactta
Genome B: gcattaGACGATCGCGGCCGTAGATCGAATAACGAGAGAGCATAAtccagag

A maximal unique matching subsequence (MUM) of 39 nt (shown in uppercase) shared by Genome *A* and Genome *B*. Any extension of the MUM will result in a mismatch. By definition, an MUM does not occur anywhere else in either genome.

Delcher et al. Nucleic Acids Res 27, 2369 (1999)

MUMmer: wichtige Schritte

- Erkenne MUMs (Länge wird vom Benutzer festgelegt)



Definition von MUMmers

- Für zwei Strings S1 und S2 und einen Parameter *l*
- Der Substring *u* ist eine MUM Sequenz wenn gilt:
  - $|u| > l$
  - *u* kommt genau einmal in S1 und genau einmal in S2 (Eindeutigkeit) vor
  - Für jeden Buchstaben *a* kommt weder *ua* noch *au* sowohl in S1 als auch in S2 vor (Maximalität)

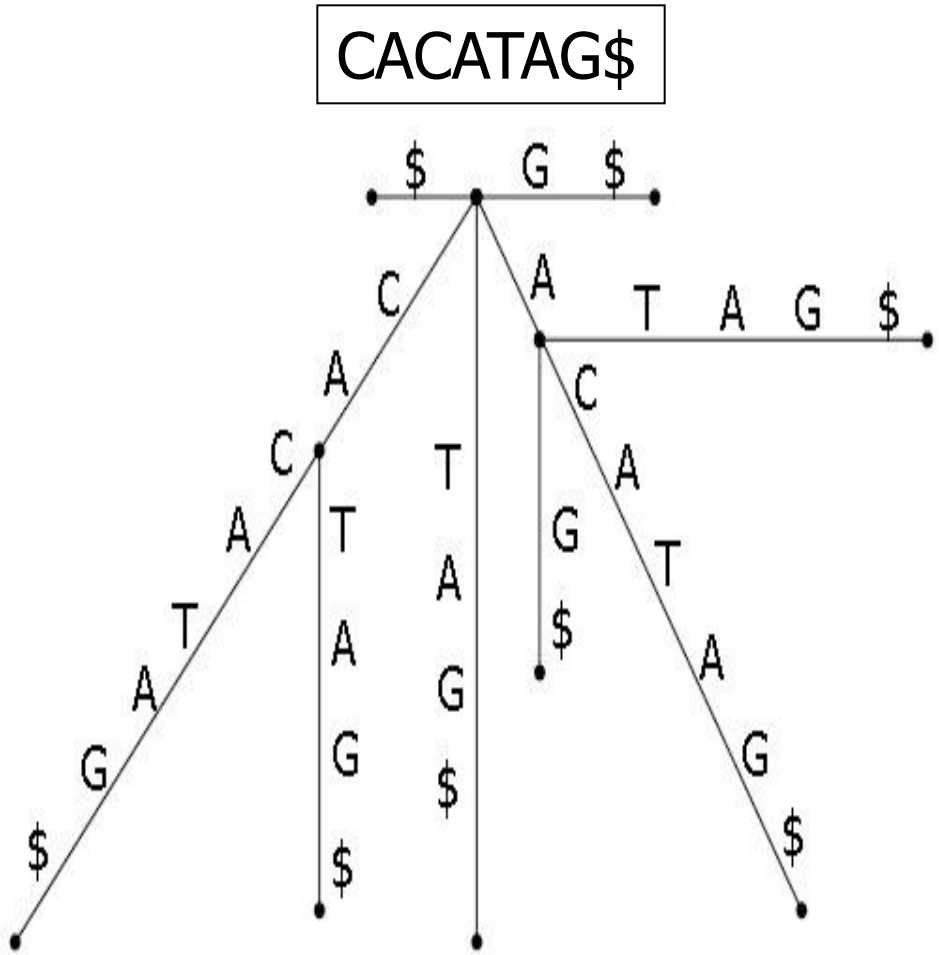
Wie findet man MUMs?

- Naiver Ansatz
  - Vergleiche alle Teilsequenzen von A mit allen Teilsequenzen von B.
    Dies dauert $O(n^n)$

- verwende Suffix-Bäume als Datenstruktur
  - ein naiver Ansatz, einen Suffix-Baum zu konstruieren hat
    eine quadratische Komplexität in der Rechenzeit und dem Speicherplatz

  - durch klevere Benutzung von Pointern gibt es lineare Algorithmen in Rechenzeit und Speicherplatz wie den Algorithmus von McCreight

Suffix-Bäume



Suffix-Bäume sind seit über 30 Jahren wohl etabliert.

Einige ihrer Eigenschaften:

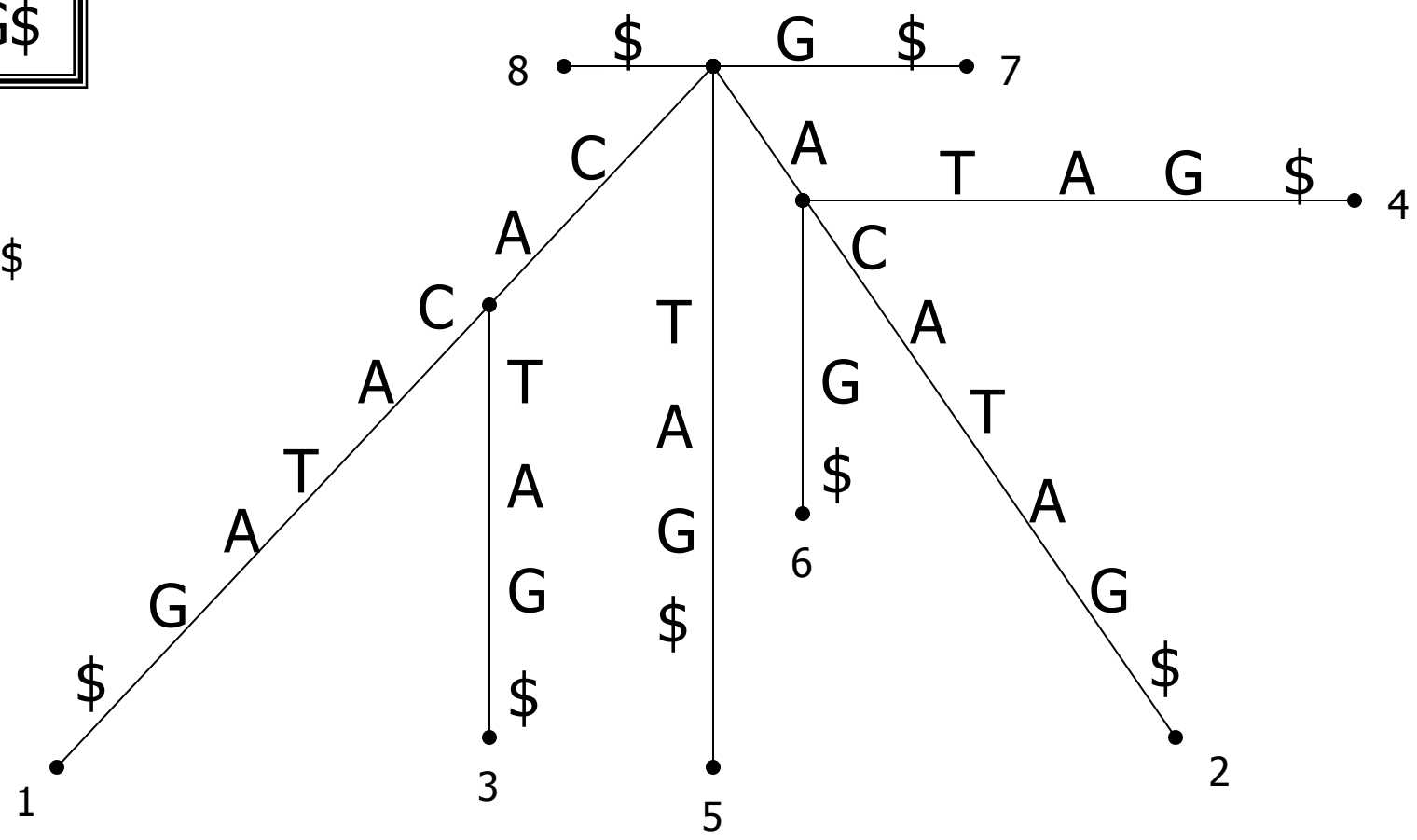
- ein “Suffix” beginnt an jeder Position *I* der Sequenz und reicht bis zu ihrem Ende.
- Eine Sequenz der Länge *N* hat *N* Suffices.
- Es gibt *N* Blätter.
- Jeder interne Knoten hat mindest zwei Kinder.
- 2 Kanten aus dem selben Knoten können nicht mit dem selben Buchstaben beginnen.
- Am Ende wird $ angefügt

Konstruktion eines Suffix-Baums

CACATAG$

Suffixes:

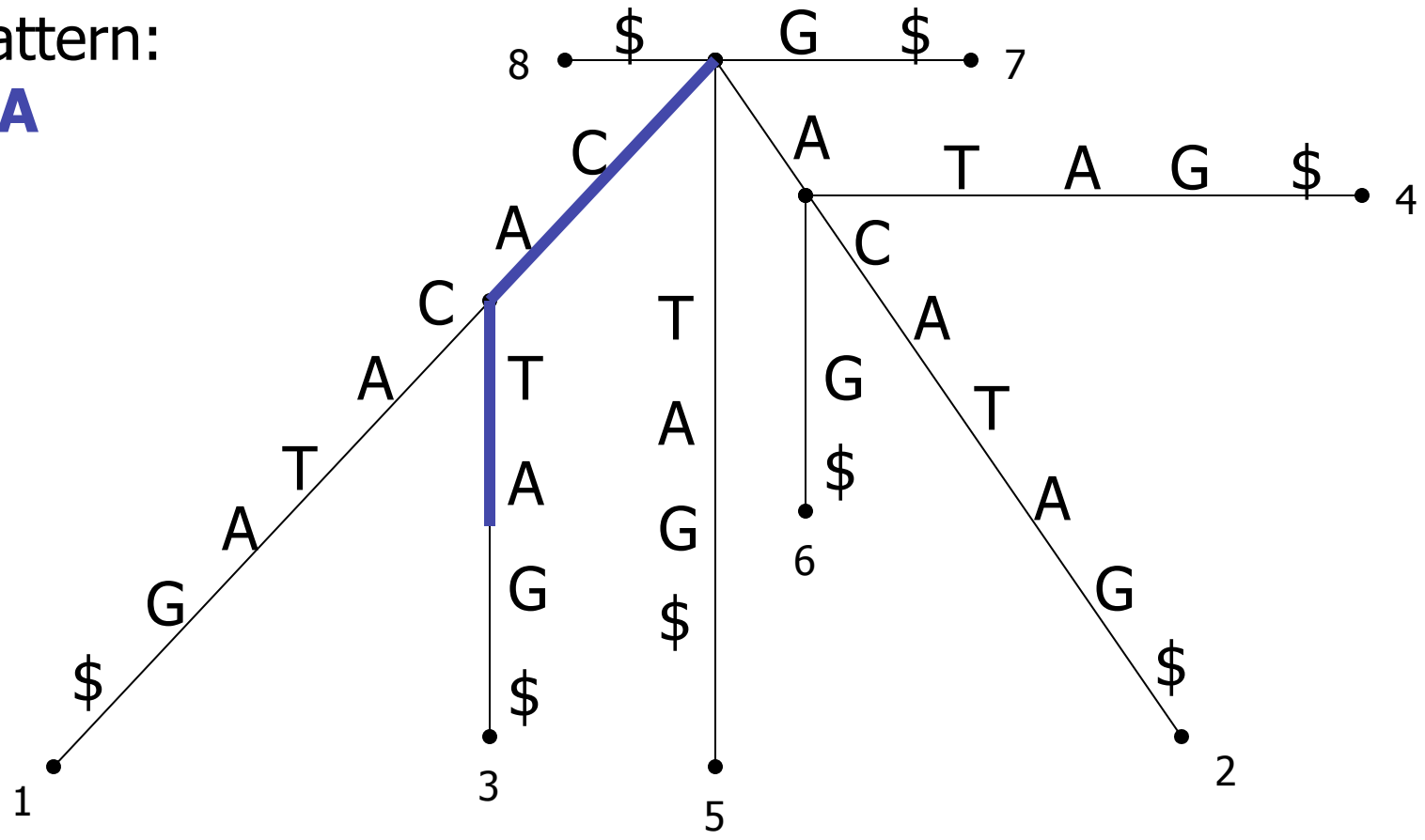
1. CACATAG$
2. ACATAG$
3. CATAG$
4. ATAG$
5. TAG$
6. AG$
7. G$
8. $



Suchen in einem Suffix-Baum

Search Pattern:

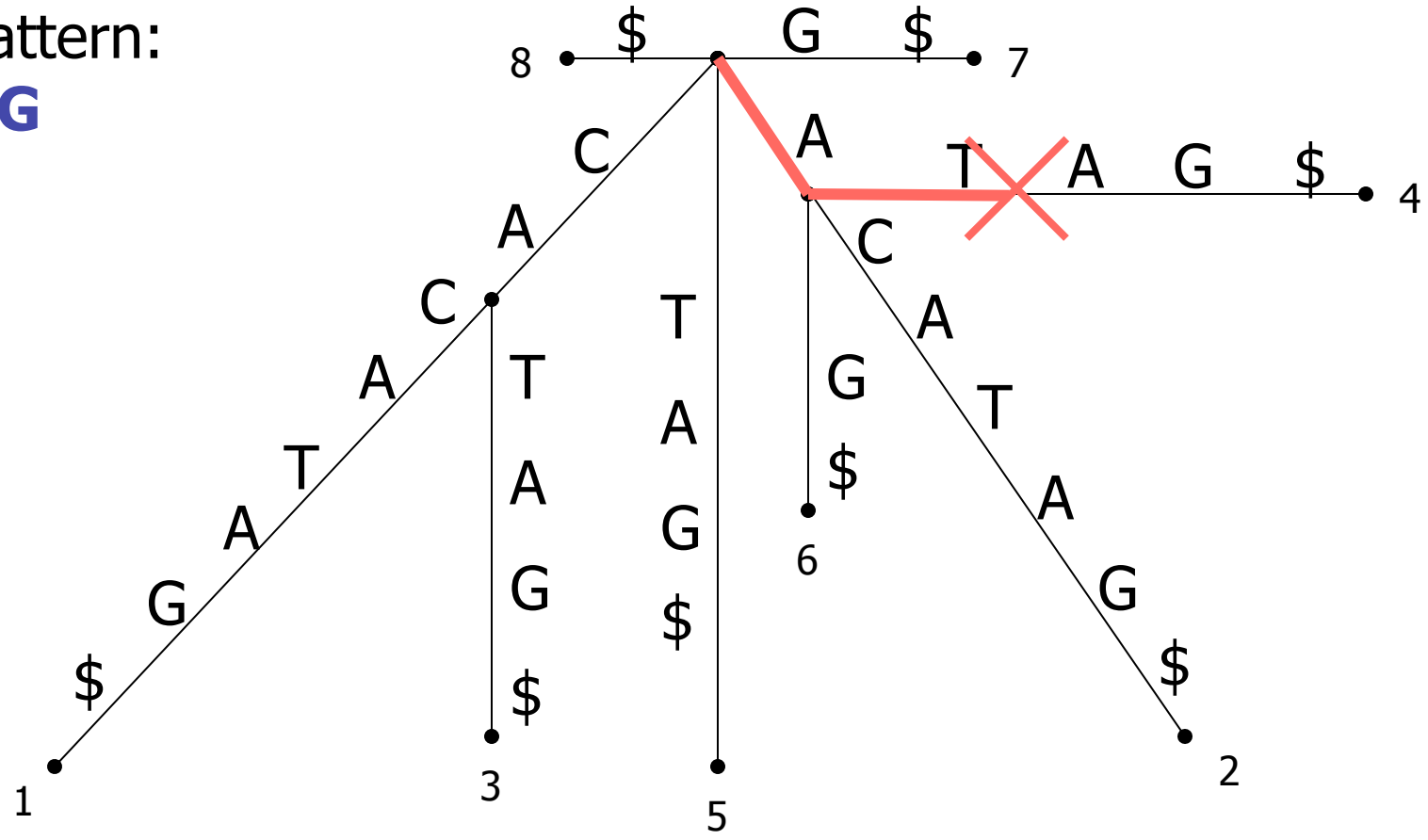
CATA



Suchen in einem Suffix-Baum

Search Pattern:

ATCG



Zusammenfassung

- Die Anwendung der Suffix-Bäume war ein Durchbruch für die Alignierung ganzer Genome

- MUMmer 2 besitzt zusätzliche Verbesserung für die Rechenzeit und den Speicherplatz
  - die Verwendung von Suffix-Arrays anstatt von Suffix-Bäumen gibt eine verbesserte Datenstruktur ($\rightarrow$ Stefan Kurtz, Hamburg)
  - es wird nun möglich, mehr als zwei Genome zu alignieren (implementiert in MGA)