Bioinformatik

Substitutionsmatrizen

Heuristiken zur Datenbanksuche

Silke Trißl / Ulf Leser

Wissensmanagement in der Bioinformatik

Wiederholung

- Wie ähnlich sind sich 2 Sequenzen

- Dotplot
- Globales Alignment
  - Editabstand = Alignmentabstand
  - Abstand minimieren
- Lokales Alignment
  - Ähnlichkeit von Teilstrings
  - Ähnlichkeit maximieren

Wie ähnlich sind sich zwei Sequenzen?

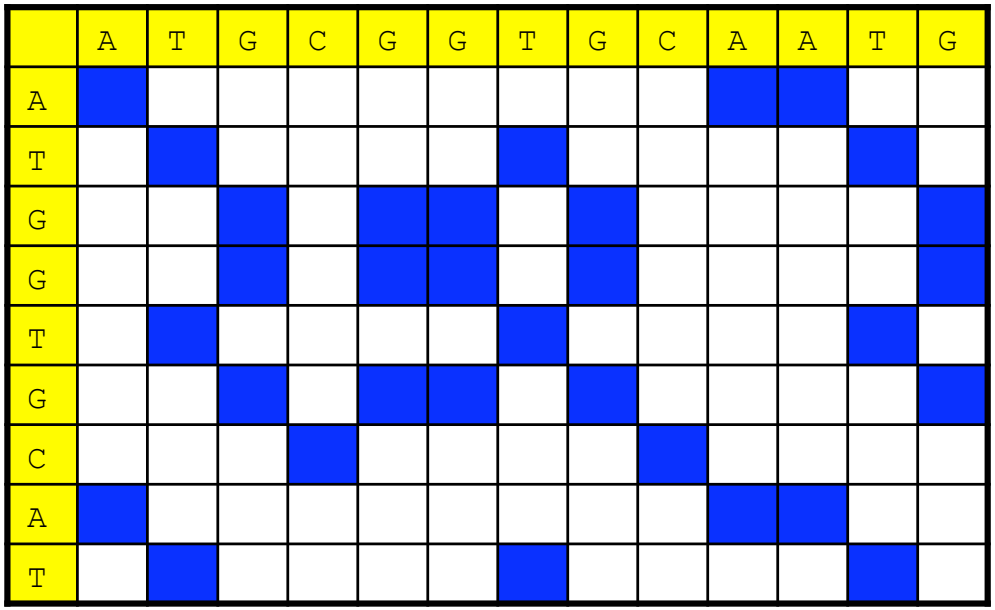
- „AGGTAG“ und
  - AGTAG — G zu wenig
  - AGGTAG — Identisch = überaus ähnlich
  - AGGATAG — A zu viel
  - AGTTCAG — G durch T ersetzen und C löschen
  - …TGAGGTAGGTT… — Sehr viel löschen
- Welche Sequenzen sind sich also besonders ähnlich?
  - „Ähnlichkeit“ muss quantifiziert werden
  - Idee: Wie sehr muss man eine Sequenz verändern, um die andere zu erzeugen
    - Man konnte auch Buchstaben zählen, Länge vergleichen, Anzahl GC nehmen, …

Motivation

- Relevant ist die biologische Funktion, nicht die Sequenz
  - Gleiche Sequenzen – gleiche Funktion
  - Sehr ähnliche Sequenzen – sehr ähnliche Funktion
  - Etwas ähnliche Sequenzen – verwandte Funktion?
- Idee: Funktionsähnlichkeit durch Sequenzähnlichkeit approximieren
  - Comparative Genomics
  - Bestimmung von Funktionen ist extrem aufwändig (wenn überhaupt möglich)
  - Bestimmung von Sequenzen ist vergleichsweise billig
- Außerdem: Hohe Ähnlichkeit kann meistens kein Zufall sein
  - Konservierung von Sequenzen trotz Evolution
  - Hoch ähnliche Sequenzen in evolutionär entfernten Spezies ist starkes Signal für hohen evolutionären Druck auf der Sequenz
  - Billige Methode, um „wichtige" (funktionstragende) Sequenzabschnitte zu finden

Dotplot

- Definition:
  Ein Dotplot zweier Strings A, B ist eine Matrix M:
  - *Die Spalten entsprechen den Zeichen von A*
  - *Die Zeilen entsprechen den Zeichen von B*
  - *M[a,b]=1 (blau) gdw. A[a] = B[b]*
- Beispiel



Abstandsmaße

- Wir suchen ein Maß für die Ähnlichkeit zweier Sequenzen
- Voraussetzung dafür sind Definitionen von:
  - Was heißt ähnlich?
  - Was heißt „am ähnlichsten“?
- Biologischen Hintergrund beachten
  - Situation: Wir haben humane Gensequenz A und suchen ähnliche Sequenzen (B) in anderen Organismen
  - Annahme: A und B haben gemeinsamen Vorfahren X und sind aus diesem jeweils durch evolutionäre Prozesse entstanden
  - Einfaches Modell evolutionärer Prozesse: Basenaustausch, Baseneinfügung, Basenlöschen

Editskripte

- Definition
  Ein Editskript e für zwei Strings A, B aus $\Sigma^{*=}\Sigma \cup$ "_" ist eine Sequenz von Editieroperationen
    - *I (Einfügen eines Zeichen $c \in \Sigma$ in A)*
      - *Dargestellt als Lücke in A; das neue Zeichen erscheint in B*
    - *D (Löschen eines Zeichen c in A)*
      - *Dargestellt als Lücke in B; das alte Zeichen erscheint in A*
    - *R (Ersetzen eines Zeichen in A mit einem anderen Zeichen in B)*
    - *M (Match, d.h., gleiche Zeichen in A und B an dieser Stelle)*

  so, dass e(A)=B
- Beispiel: A=„ATGTA", B=„AGTGTC"
  - `MIMMMR` `IRMMMDI`
    `A_TGTA` `_ATGTA_`
    `AGTGTC` `AGTGT_C`

Editabstand

- Offensichtlich gibt es für A,B ziemlich viele Editskripte
- Definition
  - *Die Länge eines Editskript ist die Anzahl von Operationen o im Skript mit o∈{I, R, D}*
  - *Der Editabstand zweier Strings A, B ist die Länge des kürzesten Editskript für A, B*
- Bemerkung
  - Für den Abstand zählen nur die Änderungen
  - Anderer Name: Levenshtein-Abstand
  - Es gibt oft verschiedene kürzeste Editskripte
  - IMMMMMD ?
    _AGAGAG
    GAGAGA_

Editabstand

- Offensichtlich gibt es für A,B ziemlich viele Editskripte
- Definition
  - *Die Länge eines Editskript ist die Anzahl von Operationen o im Skript mit o∈{I, R, D}*
  - *Der Editabstand zweier Strings A, B ist die Länge des kürzesten Editskript für A, B*
- Bemerkung
  - Für den Abstand zählen nur die Änderungen
  - Anderer Name: Levenshtein-Abstand
  - Es gibt oft verschiedene kürzeste Editskripte
  -
```
IMMMMMD      DMMMMMI
_AGAGAG      AGAGAG_
GAGAGA_      _GAGAGA
```

Alignment

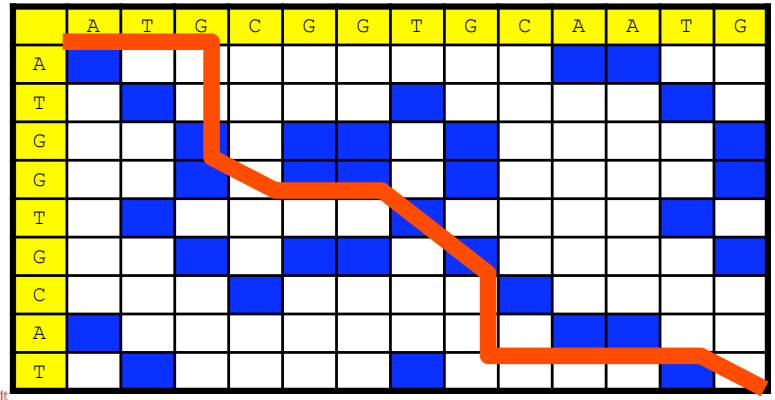
- Definition
  - *Ein (globales) Alignment zweier Strings A,B ist eine Untereinanderanordnung von A und B, jeweils mit beliebigen zusätzlichen Leerzeichen (_), ohne das zwei Leerzeichen untereinander stehen*
    - Achtung: Untereinanderstehende Zeichen müssen nicht matchen
  - *Der Alignmentscore eines Alignment ist die Anzahl von Leerzeichen und Mismatches*
  - *Der Alignmentabstand zweier Strings A, B ist der minimale Alignmentscore aller Alignments der beiden Strings*
- Beispiele

A_TGT_A	A_T_GTA	_AGAGAG	AGAGAG_
AGTGTC_	_AGTGTC	GAGAGA_	_GAGAGA
Score: 3	**5**	**2**	**2**

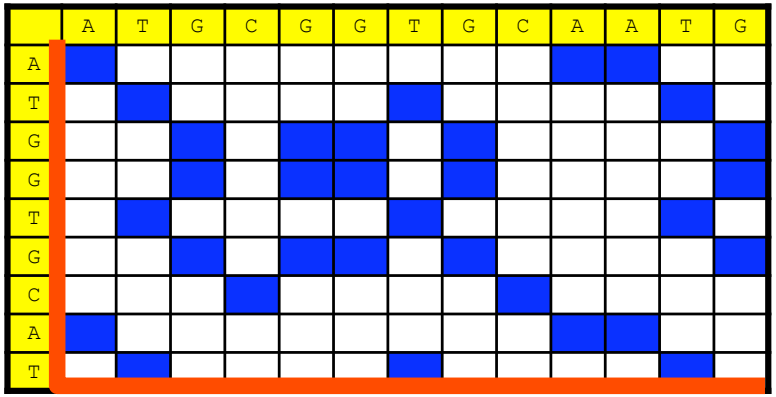
Alignments und Dotplots 2

- Sei |A|=m, |B|=n
- Betrachte Pfade im Dotplot von (1,1) nach (m,n)
  - Alignment: Sei A über B angeordnet
  - Jeder Pfad startet in der linken oberen Ecke
  - Schritt nach rechts: Nächstes Zeichen von A; „_“ in B
  - Schritt nach unten: Nächstes Zeichen von B; „_“ in A
  - Schritt nach rechts-unten: Nächstes Zeichen von A und B

```
ATG___CGGTG__CAATG
___ATGG__TGCA____T
```



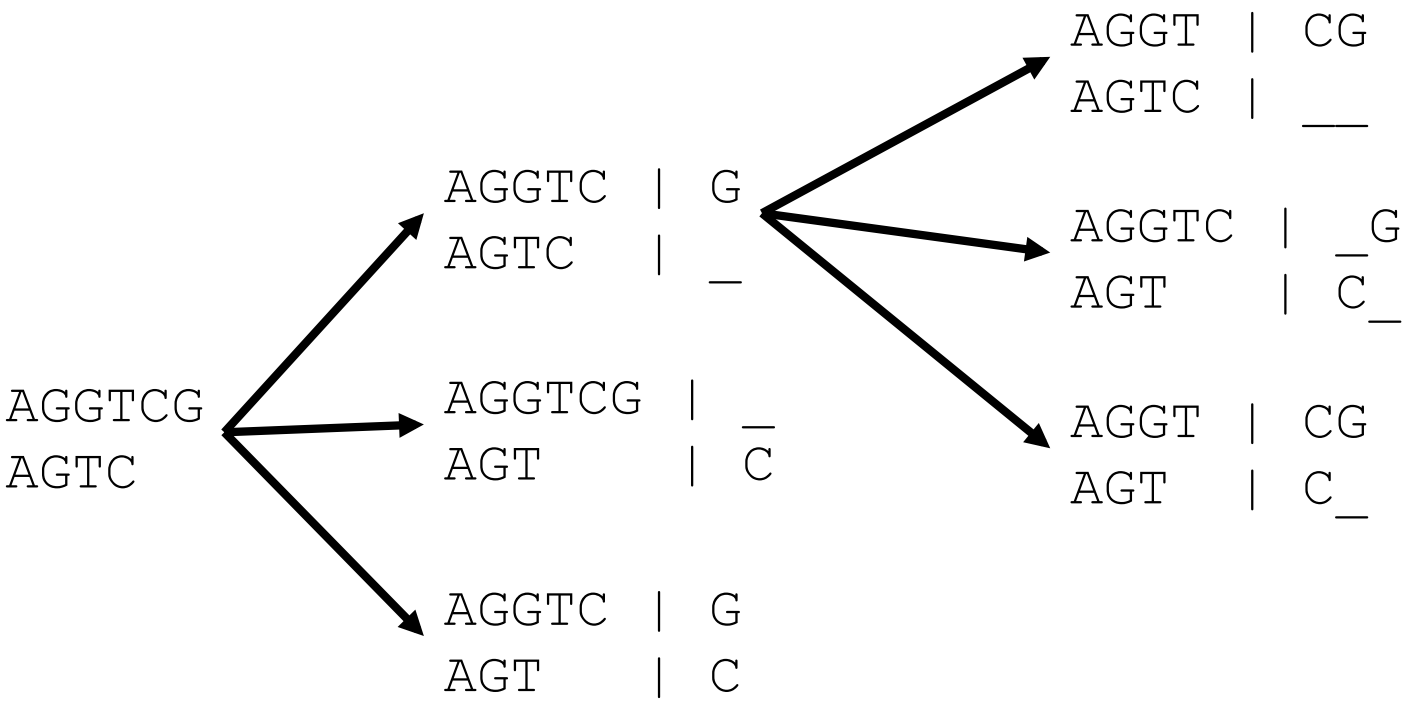

```
___________ATGCGGTGCAATG
ATGGTGCCAT_____________
```



Berechnung von Editabständen

- Definition.
  Gegeben zwei Strings A, B mit |A|=n, |B|=m
  - *Sei dist(A,B) der Editabstand von A und B*
  - *Sei d(i,j), 0≤i≤n und 0≤j≤m, der Editabstand zwischen A[1..i] und B[1..j]*
- Bemerkungen
  - Offensichtlich gilt: d(n, m)=dist(A,B)
  - Definition von d(i,j) dient zur rekursiven Berechnung von dist(A,B)

Rekursive Definition von Alignments



Rekursionsgleichung

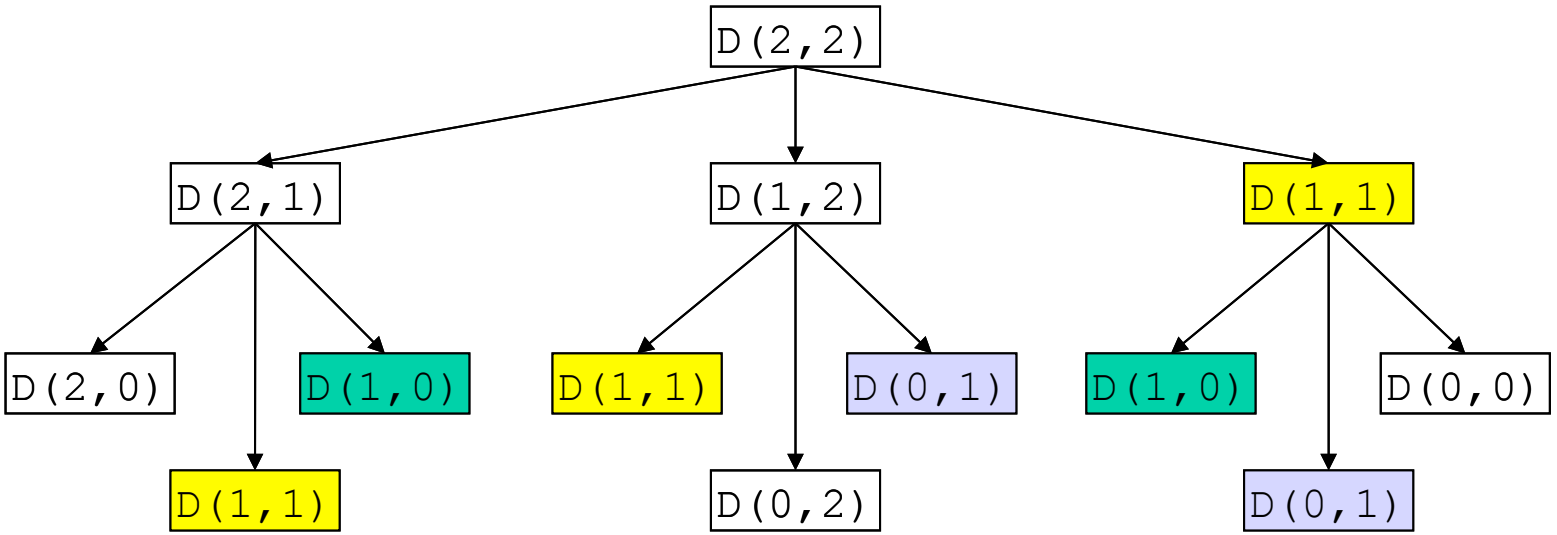
- Wir leiten das nächste Symbol im Editskript aus schon bekannten Editabständen ab
- Wir suchen das kürzeste Skript, also

$$d(i,j) = \min \begin{Bmatrix} d(i,j-1)+1 \\ d(i-1,j)+1 \\ d(i-1,j-1)+t(i,j) \end{Bmatrix}$$

$$t(i,j) = \begin{cases} 1 : wenn\ \ A[i] \neq B[j] \\ 0 : sonst \end{cases}$$

Sicher nicht optimal

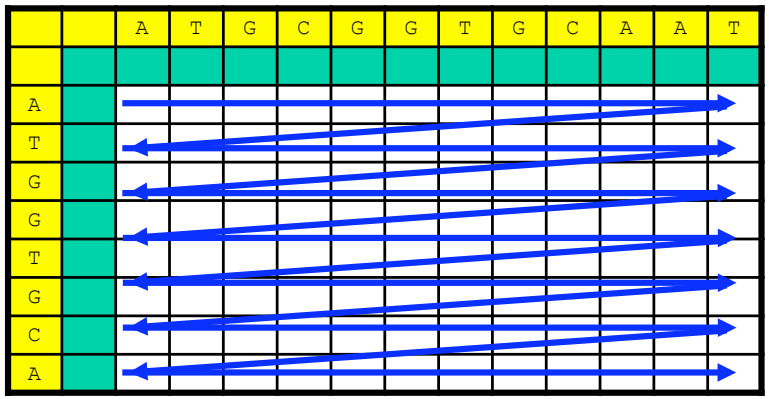
- Durch die Rekursionsgleichung werden viele Teillösungen mehrfach berechnet

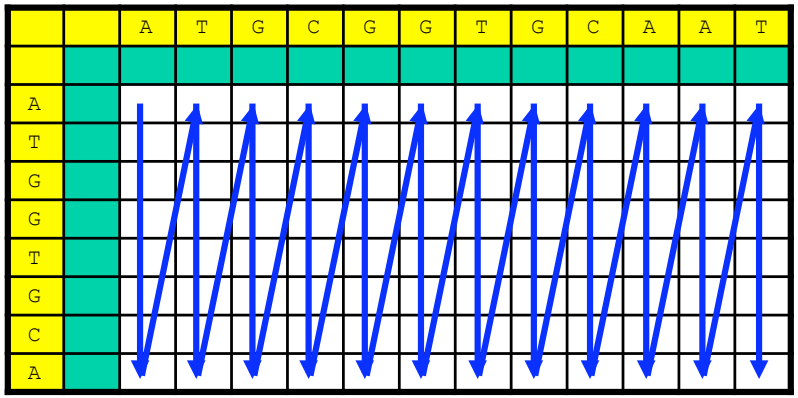


- Es gibt nur (n+1)*(m+1) verschiedene Aufrufe
- Wie kann man die redundanten Berechnungen sparen?

Tabellarische Berechnung

- Grundidee
  - Speichern der Teillösungen in Tabelle (2-dimensionales Array)
- Aufbau der Tabelle: Bottom-Up (statt rekursiv Top-Down)
  - Initialisierung mit festen Randwerten d(i,0) und d(0,j)
  - Sukzessive Berechnung von d(i,j) für steigende i,j
  - Zur Berechnung eines d(i,j) brauchen wir d(i,j-1), d(i-1,j) und d(i-1,j-1)
  - Verschiedene Reihenfolgen möglich





Vom Pfad zum Alignment

		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
T	2	1	0	1	2	3	4	5
G	3	2	1	0	1	2	3	4
G	4	3	2	1	1	1	2	3

- Jeder Pfad von (n,m) nach (1,1) ist ein optimales Alignment
  - Starte von (1,1)
  - Nach rechts: Deletion in A
  - Nach unten: Insertion in A
  - Diagonal: Match/Replace

		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
T	2	1	0	1	2	3	4	5
G	3	2	1	0	1	2	3	4
G	4	3	2	1	1	1	2	3

```
ATGCGGT
ATG_G__
```

		A	T	G	C	G	G	T
	0	1	2	3	4	5	6	7
A	1	0	1	2	3	4	5	6
T	2	1	0	1	2	3	4	5
G	3	2	1	0	1	2	3	4
G	4	3	2	1	1	1	2	3

```
ATGCGGT
AT__GG_
```

Formal

- Definition
  Gegeben Alphabet Σ'=Σ∪'_', Strings A,B über Σ' mit |A|=|B|=n
  - *Eine Scoringfunktion ist eine Funktion s: Σ'xΣ' → Integer*
    - *Substitutionsmatrix*
  - *Die Ähnlichkeit von A,B bzgl. der Scoringfunktion s ist*

$$sim(A,B) = \sum_{i=1}^{n} s\left(A[i], B[i]\right)$$

- Bemerkung
  - Wir betrachten i.d.R. Alignments, nicht beliebige A,B
  - Optimales Alignment ~ Alignment mit höchster Ähnlichkeit

Rekursionsgleichung

- Nur kleine Veränderung

$$d(i,0) = \sum_{k=1}^{i} s(A[k], \_) \qquad d(0,j) = \sum_{k=1}^{j} s(\_, B[k])$$

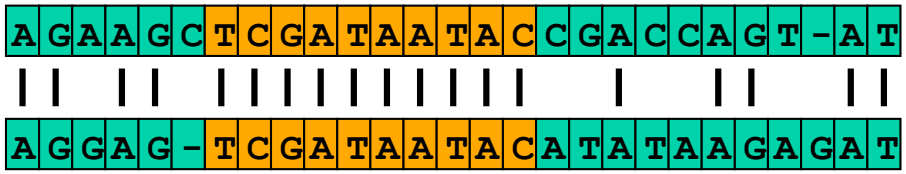
$$d(i,j) = \max \begin{Bmatrix} d(i, j-1) + s(\_, B[j]) \\ d(i-1, j) + s(A[i], \_) \\ d(i-1, j-1) + s(A[i], B[j]) \end{Bmatrix}$$

Lokale Alignments

- Definition. *Gegeben zwei Strings A, B.*
  - *Seien a,b Substrings mit a∈A, b∈B so dass*

$$sim(a,b) = \max_{\forall a' \in A, b' \in B} \left(sim(a',b') \right)$$

  - *Das vom (globalen) Alignment von a und b induzierte Alignment von A und B heißt lokales Alignment von A und B*
  - *Der lokale Ähnlichkeitsscore* $dist_{local}(A,B) = sim(a,b)$
- Bemerkung
  - Lokale Alignments sind unempfindlich gegen unterschiedliche lange Strings
- Beispiel
  - Lokales A. findet den identischen Substring
  - Das ist die biologisch wichtige Information



Smith-Waterman Algorithmus

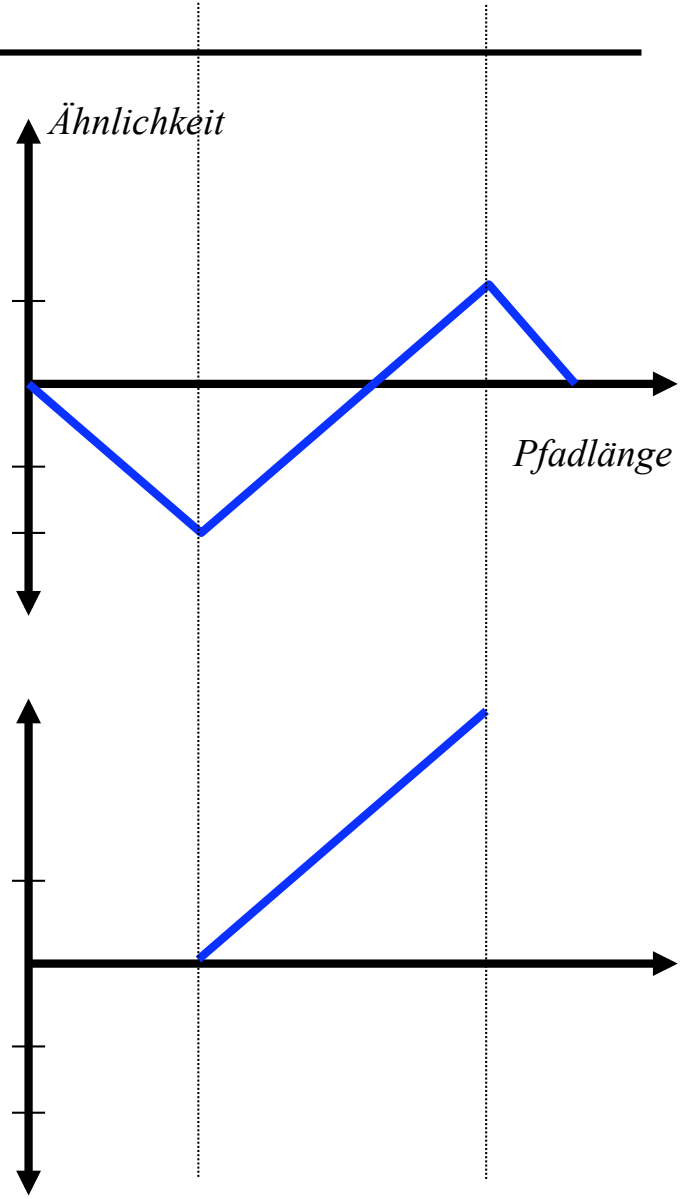
- Smith, Waterman: „Identification of common molecular subsequences“, J. Mol. Bio 147, 1981
- Grundidee
  - Geringfügige Veränderung des Algorithmus für globales Alignment
  - Annahme: Scoring Funktion mit positiven Werten für Matches und negativen Werten für alles andere
  - Eine Reihe von Matches beim Vergleich erzeugt also sukzessive größere Ähnlichkeitswerte
  - Bei Mismatch oder Insertion wird der Wert wieder kleiner
  - Um lange zusammenhänge Regionen zu erhalten, können wir Substringmatches mit negativem Gesamtwert vergessen
  - Also: Statt in negative Werte zu rutschen, darf der Algorithmus auch bei 0 anfangen

Match: +1
I/R/D: -1

Beispiel

		A	T	G	T	G	G
	0	-1	-2	-3	-4	-5	-6
G				-1			
T					0		
G						1	
A							0

		A	T	G	T	G	G
	0	-1	0	-3	-4	-5	-6
G				1			
T					2		
G						3	
A							0



Optimales lokales Alignment

- Theorem
  Gegeben Strings A,B. Mit der folgenden Funktion v(i,j), mit 0≤i≤n und 0≤j≤m

$$v(i,j) = \max \begin{Bmatrix} 0 \\ v(i,j-1) + s(\_, B[i]) \\ v(i-1,j) + s(A[i], \_) \\ v(i-1,j-1) + s(A[i], B[j]) \end{Bmatrix}$$

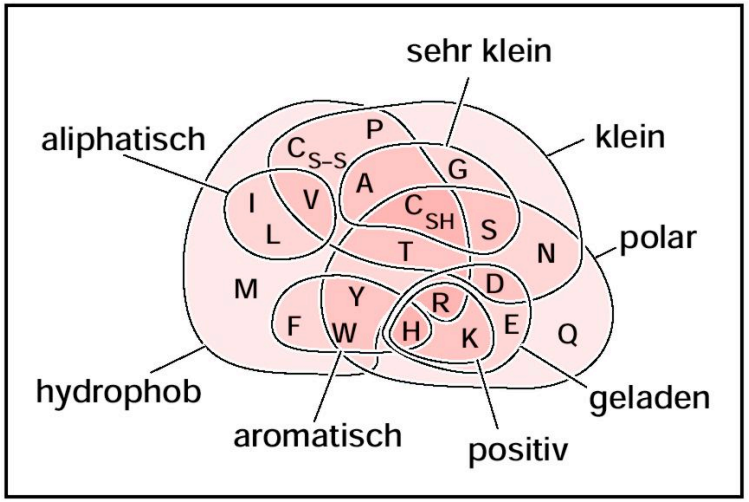
  - *Gilt:* $dist_{local}(A,B) = \max_{i,j}(v(i,j))$
- Traceback
  - Starte beim maximalen Wert in der Matrix
  - Verfolge beliebigen Pfad bis zu einer Zelle mit Wert 0

Inhalt dieser Vorlesung

- Substitutionsmatrizen: PAM
- Heuristiken zur Datenbanksuche
  - BLAST
  - (BLAT)

Hintergrund

- Schon öfters angesprochen ...
  - Ähnlichkeitsmatrizen, Substitutionsmatrizen, Scorefunktionen, ...
- Ersetzung einer Base/Aminosäure durch eine andere hat unterschiedliche Bedeutung
  - Aminosäuren
    - Ersetzung mit „sehr ähnlichen" Aminosäuren ändert Proteinstruktur in der Regel kaum
    - Ersetzung mit „wenig ähnlichen" Aminosäuren kann Struktur vollkommen ändern



Substitutionsmatrizen

- Individuelle Bewertung von Replacements/Mismatches
- Alignmentalgorithmus bleibt unberührt
  - Das Ergebnis kann sich aber vollkommen ändern
- Beispiele: Blosum62, Identitätsmatrix

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

	A	C	G	T
A	5	-4	-4	-4
C	-4	5	-4	-4
G	-4	-4	5	-4
T	-4	-4	-4	5

Ist das notwendig?

Code	Häufig-keit	Mutier-barkeit
L	0.091	54
A	0.077	100
G	0.074	50
S	0.069	117
V	0.066	98
E	0.062	77
K	0.059	72
T	0.059	107
I	0.053	103
D	0.052	86
P	0.051	58
R	0.051	83
N	0.043	104
Q	0.041	84
F	0.040	51
Y	0.032	50
M	0.024	93
H	0.023	91
C	0.020	44
W	0.014	25

- Häufigkeiten der Ersetzung einer Aminosäure durch irgendeine andere AA im Verhältnis zu allen Ersetzungen
- Alanin (A) willkürlich als 100% gesetzt
- Keine Gleichverteilung
- Mutationen sind mehr oder weniger erfolgreich, je nachdem, welche AA ersetzt wird
  - Besser: Mutationen werden durch Selektion mehr oder weniger geduldet
  - Tryptophan (W) sehr selten (25)
  - Serin (S) sehr häufig (117)

Woher nehmen?

- Wie kann man sinnvolle Werte für die Matrix bestimmen?
  - Wir wollen Ähnlichkeit der biologischen Bedeutung messen
  - Für Aminosäuren benötigt man ~200 (verschiedene) Werte
- Möglichkeit 1: Chemische Eigenschaften
  - Ladung, Größe, Polarität, ...
  - Viele Faktoren mit unklaren Gewichten
  - Wie soll man das durch ein Bewertungsschema ausdrücken?
  - Keine Verwendung in der Praxis
- Möglichkeit 2: Beobachtung
  - Beobachtung der Evolution statt analytischer Vorhersage
  - Lernen aus Beispielen, also „tatsächlich" vorgekommener Mutationen
  - Benötigt Beispieldaten: Paare von homologen Sequenzen

Margaret O. Dayhoff

- Ziel: "Deduce evolutionary relationships of the biological kingdoms, phyla, and other taxa from sequence evidence"
- Collection of all known protein sequences
- First edition contained sequence information of 65 proteins
- Several releases followed
- Resulted in the Protein Information Resource (PIR)

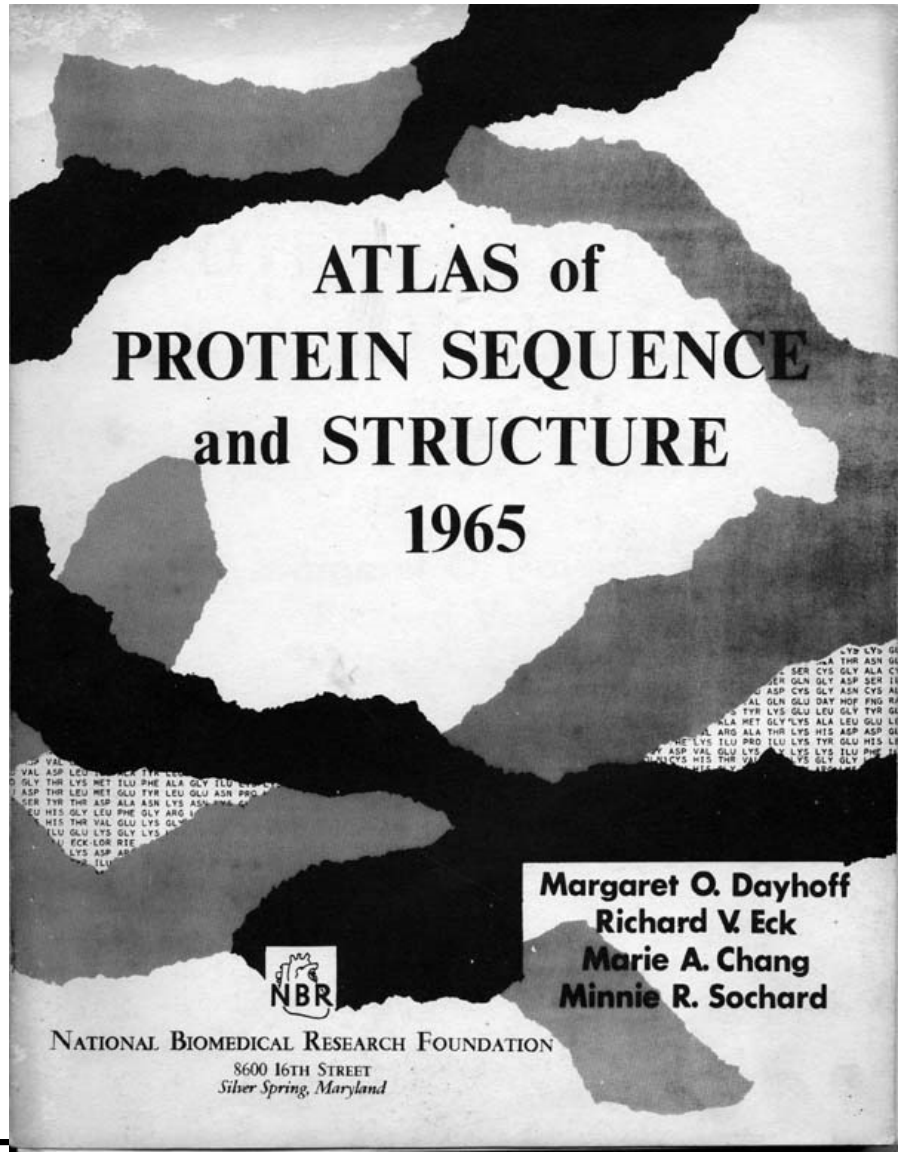


Bild: Dank an Antje Krause

PAM: Point-Accepted Mutations

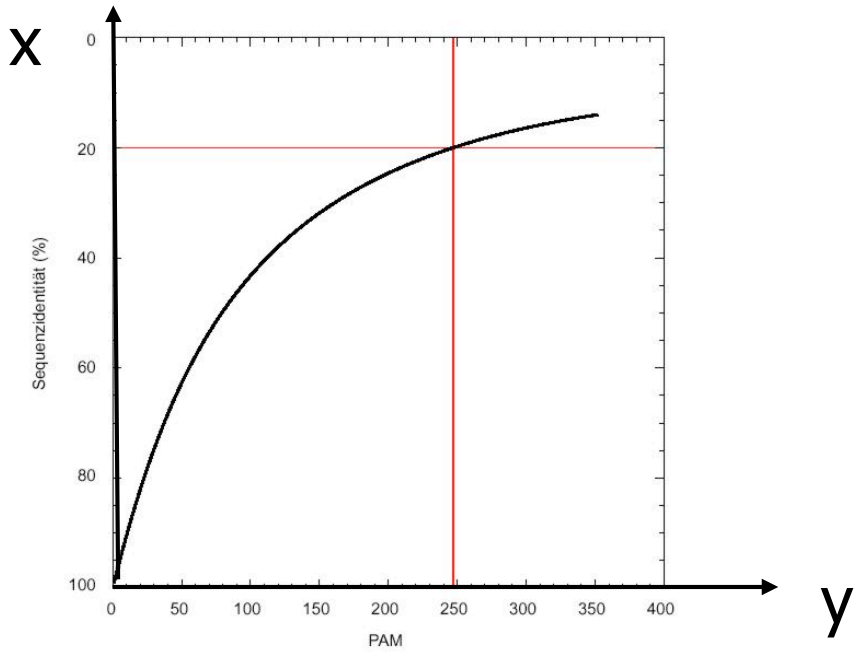
- Dayhoff, M. O., R. V. Eck, C. M. Park. 1972. A model of evolutionary change in proteins. Pp. 89–99. *in* M. O. Dayhoff, ed., Atlas of Protein Sequence and Structure Vol. 5.

- PAM: Zwei Bedeutungen
  - 1 PAM – Einheit für den Abstand von Proteinsequenzen
  - PAM-X Matrix – Berechnete Substitutionsmatrix für zwei Sequenzen, die X PAM entfernt sind

Vorbetrachtung

- Annahme
  - Sequenz von 100 Aminosäuren
  - Mit jeder Generation ändert sich eine zufällig ausgewählte Position
    - Achtung: Das ist eine starke Vereinfachung
    - Molecular Clock theory
- Betrachten wir y Generationen
  - Menge der stattgefundenen Änderungen = y
  - Die Zahl x der beobachtbaren Änderungen korreliert mit y
  - Wenn man Ausgangs- und Ergebnissequenz kennt (also x berechnen kann), kann man y abschätzen
    - Mit welcher Wahrscheinlichkeit hat sich eine gegebene Position verändert?
    - Wie viele Positionen haben sich wahrscheinlich verändert?

Mutationshäufigkeit und Sequenzidentität



- Kein linearer Zhg: Rückmutationen, Doppelmutationen, etc.
- Mit steigendem y wächst der Abstand zur Ausgangssequenz ab einem bestimmten Punkt kaum noch
- Ab diesem Punkt kann man über Homologie kaum noch eine Aussage machen

PAM als Sequenzabstand

- Definition
  Seien S_1 und S_2 zwei Proteinsequenzen mit $|S_1|=|S_2|$. S_1 und S_2 heißen x PAM entfernt, wenn S_1 in S_2 überführt wurde mit x Punktmutationen pro 100 Aminosäuren
- Eigenschaften
  - PAM beachtet keine Inserts und Deletions
  - x schätzt man aus den Unterschieden von S_1 und S_2
    - Kann man analytisch berechnen oder durch Simulation bestimmen
  - 50 PAM Abstand heißt nicht 50 Veränderungen pro 100 Aminosäuren

PAM Matrizen

- Definition
  Seien ($S_{1,1}$, $S_{2,1}$), ..., ($S_{1,n}$, $S_{2,n}$) Paare von Sequenzen, die jeweils x PAM entfernt sind. Dann ist die PAM-x Matrix M_x wie folgt definiert:
  - *Sei f(i) die relative Häufigkeit der Aminosäure A_i*
  - *Seien alle Paare optimal aligniert*
    - *Sei $S_{k,l}$' das $S_{k,l}$ mit den durch das Alignment eingefügten Leerzeichen*
  - *Sei f(i,j) die relative Übergangshäufigkeit von A_i zu A_j in allen alignierten Paaren*
    - *Anzahl von Positionen k mit $S_{1,z}$'[k]=A_i und $S_{2,z}$'[k]=A_j oder umgekehrt*
      - *Übergang ist „richtungslos": f(i,j) = f(j,i)*
      - *Paare (A_x, _) werden ignoriert*
    - *f(i,j) wird auf die Gesamtzahl aller Paare ohne INSDEL normiert*
  - *Damit:*

$$M_x(i,j) = \log\left(\frac{f(i,j)}{f(i) * f(j)}\right)$$

Erläuterung

- Log-Odds Ratio
- Numerischer Trick: Logarithmus zur Ersetzung von Multiplikation mit Addition
- Bruch
  - Verhältnis der beobachteten Ersetzung zu den erwarteten Übersetzungen bei statistischer Unabhängigkeit

$$M_x(i,j) = \log\left(\frac{f(i,j)}{f(i) * f(j)}\right)$$

  - M(i,j) = 0 (Bruch = 1)
    - Keine Selektion - Anzahl Übergänge entspricht statistischer Erwartung
  - M(i,j) < 0 (Bruch < 1)
    - Negative Selektion – Übergang wird unterdrückt
  - M(i,j) > 0 (Bruch > 1)
    - Positive Selektion – Übergang wird bevorzugt

Beispiel

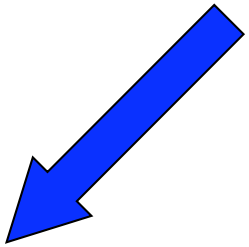
$S_{1,1}$: ACGGTGAC
$S_{2,1}$: AGG_TGCC

$S_{1,3}$: GTT_AGCTA
$S_{2,4}$: TTTCAG_TA

$S_{1,2}$: GGTCAA
$S_{2,2}$: AGTC_A

Absolute Häufigkeiten

A: 11/42	C: 8/42	G: 12/42	T: 11/42



Übergangshäufigkeiten

	A	C	G	T
A	4/19	1/19	1/19	0/19
C		2/19	1/19	0/19
G			4/19	1/19
T				5/19

Substitutionsmatrix

	A	C	G	T
A	0,48	0,02	-0,15	-
C		0,46	-0,01	-
G			0,41	-0,15
T				0,58

Reale PAM Matrizen

- Vorgehen von Dayhoff et al.
  - Paare eng verwandter Sequenzen auswählen
    - >85% Identität, 34 Proteinfamilien
  - Manuell alignieren
  - PAM-1 Matrix M_1 aus Häufigkeiten berechnen
  - PAM-x Matrizen wie folgt berechnen: $M_x=(M_1)^x$
- Dem liegen viele Annahmen zugrunde
  - Evolutionary Clock Theory: Evolution verläuft gleichmäßig
    - in der Zeit und in den Sequenzpositionen
    - Proportionalität von Veränderungen
    - Hochrechnung langer Distanzen aus kurzen
  - Keine Insertions oder Deletions
  - Unabhängigkeit der Mutationswahrscheinlichkeit von der Position in der Sequenz (und der Nachbarschaft)

BLOSUM Matrizen

- Hauptkritikpunkte am PAM Ansatz
  - Nur Verwendung sehr ähnlicher Sequenzen
    - Realistische Zahlen für evolutionär weiter entfernte Sequenzen?
  - Einbeziehung kompletter Proteinsequenzen
    - Unterliegen alle Positionen der selben Mutationsrate?
  - PAM-x vervielfältigen Fehler in PAM-1
- Anderer (neuerer) Ansatz: BLOSUM
  - Henikoff, S. and Henikoff, J. G. (1993). "Performance evaluation of amino acid substitution matrices." *Proteins* **17**(1): 49-61.
  - BLOcks SUbstitution Matrix
  - Multiple Alignments evolutionär entfernter, aber homologer Proteinsequenzen
  - Benutzung nur der konservierten Blöcke
  - Heute populärer als PAM Matrizen

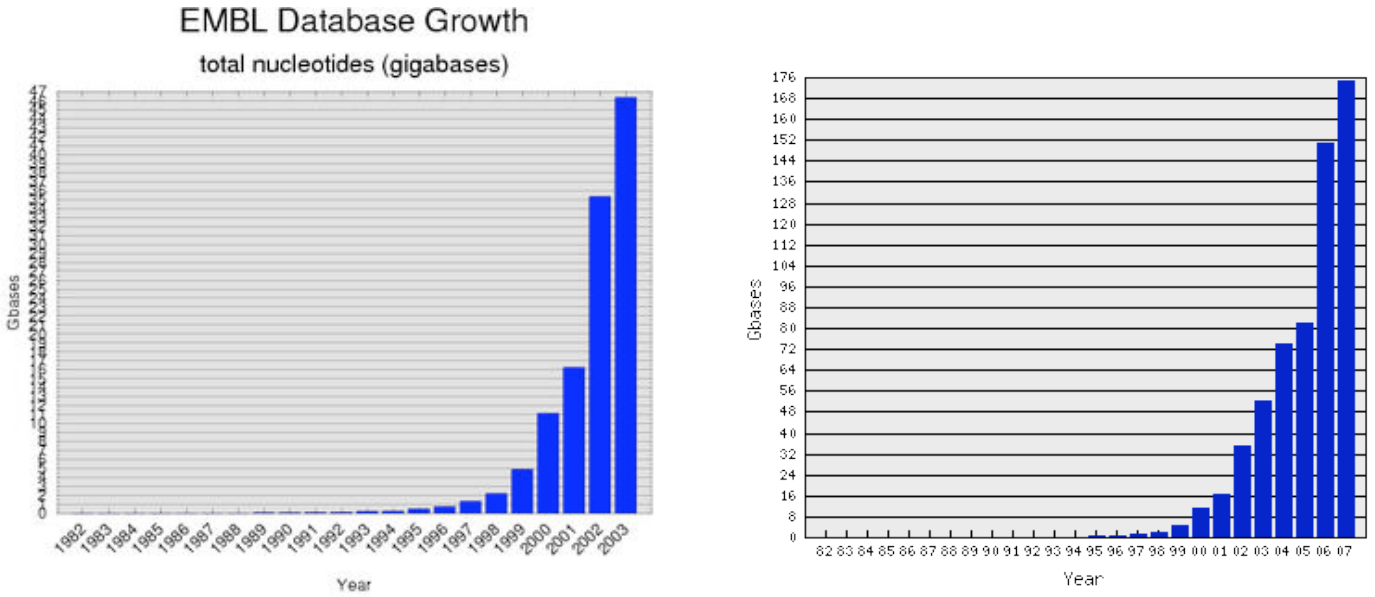
Verwendung

- Welche PAM Matrix soll man nun zur Alignierung zweier Sequenzen verwenden?
  - Die, die dem PAM-Abstand der Sequenzen entspricht
  - Den kennt man aber nicht – schätzen
  - Schätzung benötigt Alignments
    - Zur Berechnung der Sequenzidentität
  - Alignments basieren auf Substitutionsmatrizen
  - Henne – Ei Problem
- Also
  - Verschiedene Matrizen testen
  - Alignments prüfen
  - Externes Wissen (Chemie, Strukturen, etc.) hinzuziehen

Inhalt dieser Vorlesung

- Substitutionsmatrizen: PAM und BLOSUM
- Heuristiken zur Datenbanksuche
  - BLAST

Heuristische Alignierung



- Annotation neuer Sequenzen basiert auf Suche nach homologen Sequenzen in Sequenzdatenbanken
- Datenmenge wächst exponentiell – selbst lineare Algorithmen sind zu langsam
- Gesucht sind schnelle Verfahren
  - Auch wenn wir dabei ein paar (schlechte?) Ergebnisse verlieren

Suche in Datenbanken

- Gedankenkette
  - Gegeben Sequenz S und Datenbank mit Sequenzen S_1, ..., S_n
  - Gesucht: Welche Sequenzen in DB sind homolog zu S?
    - Kann nicht beantwortet werden
  - Annäherungen
    - Welche Sequenzen in DB sind sehr ähnlich zu S?
      - Was heißt ähnlich?
    - Welche Sequenzen haben einen hohen Alignment-Score zu S?
      - Berechnung Alignmentscore dauert zu lange
    - Näherung: Welche Sequenzen in DB haben wahrscheinlich einen hohen Alignmentscore zu S?
      - BLAST
  - Was heißt hier „wahrscheinlich“?

Sensitivität und Spezifität

		Reality	
		+	-
Prediction	+	TruePositive (TP)	FalsePositive (FP)
	-	FalseNegative (FN)	TrueNegative (TN)

- Spezifität = TP/(TP+FP) (Precision)
  - Wie viele der Treffen des Verfahrens sind wirklich welche?
- Sensitivität = TP/(TP+FN) (Recall)
  - Wie viele der echten Treffer findet das Verfahren?
- Oftmals eine Balance
  - Algorithmen berechnen einen Score pro Sequenz
  - Hoher Score – Positiv; Niedriger Score – Negativ
  - Wenn Score mit Wahrscheinlichkeit für korrekte Klassifikation korreliert, folgt daraus
    - Ergebnismenge klein: SP=hoch, SE=klein
    - Ergebnismenge groß: SP=niedrig, SE=hoch

Beispiel

- Gegeben
  - Sequenz S und Datenbank mit 10.000 Sequenzen
- Algorithmus findet
  - 15 Sequenzen homolog zu S
- In Wahrheit
  - 20 Sequenzen homolog zu S
  - 10 davon hat der Algorithmus gefunden

	Real: Positive	Real: Negative
Alg: Positive	TP = 10	FP = 5
Alg: Negative	FN = 10	TN= 9.975

- Spezifität = TP/(TP+FP) = 10/15 = 66%
- Sensitivität = TP/(TP+FN) = 10/20 = 50%

BLAST

- Altschul, Gish, Miller, Myers, Lipman: „Basic Local Alignment Search Tool“, J Mol Bio, 1990.
  - Heuristischer Suchalgorithmus
  - Datenbanksuche mit lokalem Alignment
  - Sehr schnell, findet aber nicht alle optimalen Alignments
- **Die** Erfolgsgeschichte der Bioinformatik
  - Für Biologen teilweise äquivalent zu „Bioinformatik“
  - Eingesetzt auf NCBI/EBI Server
  - Software frei erhältlich
- Diverse Weiterentwicklungen
  - Gapped-BLAST, PSI-BLAST, MegaBlast, BLAST-ALL, PATHBLAST, Name-BLAST, …

BLAST Varianten

- BLAST gibt es in verschiedenen Ausprägungen
  - Blastn : DNA-Anfrage/DNA-Datensammlung
  - Blastp : Protein-Anfrage/Protein-Datensammlung
  - Blastx : translatierte DNA-Anfrage/Protein-Daten
  - Tblastn : Protein-Anfrage/translatierte DNA-Daten
  - Tblastx: translatierte DNA-Anfrage/DNA-Daten
- Bei DNA/Proteinsuche immer Suche mit allen sechs Reading Frames
- Prinzip immer identisch
- Suche in Proteinen läuft etwas anders als in DNA

BLAST Parameter

- Zunächst
  - Suche nach guten lokalen Alignments in DNA Sequenzen
- Gegeben
  - Suchsequenz P, Datenbank DB={S_1,...,S_n}
  - Substitutionsmatrix M
  - Parameter w: Länge der „Seeds“
  - Parameter t: Initialer Schwellwert
  - Parameter c: Gesamtschwellwert
    - Wird berechnet in Abhängigkeit von t, M, |DB|, |P|
  - Parameter v: Erwünschte Anzahl Treffer
    - Blast berechnet die v ähnlichsten Subsequenzen

BLAST Schritt 1 und 2

- Schritt 1
  - Bestimme alle Teilwörter $P_1,...,P_m$ der Länge w in P
    - Mit Überlappung – keine Partitionierung
    - Wie viele gibt es?
- Schritt 2
  - Suche nach Hits von $P_1,...,P_m$ in DB mit Score über t
    - Hits müssen nicht exakt sein
    - Vergleiche alle P_i mit allen Teilwörtern in DB der Länge w in T
      - Keine INSDELs, Verwendung von M
  - Durch den „unscharfen" Hit mit Schwellwert t
    - werden auch Hits gefunden, die keine perfekten Matches sind
    - werden wenig aussichtsreiche Positionen in DB ausgeschlossen
      - Annahme: Ein gutes lokales Alignment A in Sequenz S muss mindestens einen Hit enthalten
      - Es ist nicht notwendig, dass jedes Teilwort in A mit seinem Teilwortpartner in P einen Hit hat.

BLAST Schritt 3

- Schritt 3
  - Für jeden Hit H zwischen DB-Sequenz S und P_i
  - Verlängere Bereich um H sowohl in P als auch in S
    - Gapfree Alignment nach links und rechts wachsen
    - Solange, bis
      - Sequenz P oder S zu Ende ist, oder
      - Alignmentscore fällt unter geschätzten Schwellwert c, oder
      - Alignmentscore fällt „signifikant" unter bisherige v beste Treffer
        - » „Signifikant" heuristisch bestimmt
    - Ergibt „Maximal Segment Pairs (MSP)"
    - Die besten v MSP sind das Ergebnis

Beispiel

```
W=5, t=5, Kosten: M=+1, R=-3
P=ACGTGATA
S=GATTGACGTGACTGCAAGTGATACTATAT
```

Schritt 1
Teilwörter

P_1=ACGTG
P_2=CGTGA
P_3=GTGAT
P_4=TGATA

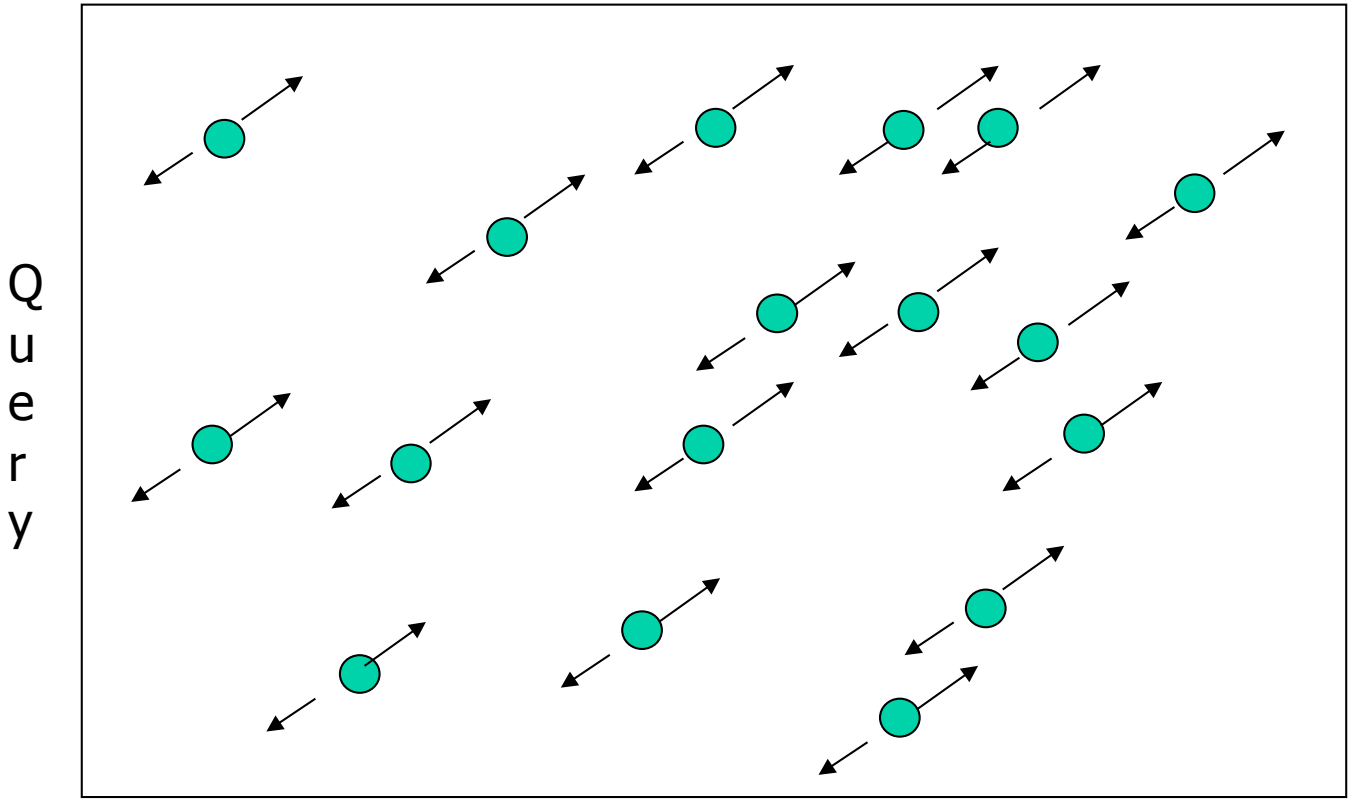
Schritt 2
Hitsuche

```
GATTGACGTGACTGCAAGTGATACTATAT
GATTGACGTGACTGCAAGTGATACTATAT
GATTGACGTGACTGCAAGTGATACTATAT
```

Schritt 3
Verlängerung

```
GATTGACGTGACTGCAAGTGATACTATAT
      ACGTGATA                  5
      ACGTGATA                  5+1=6
      ACGTGATA                  6-3=3
      ...                       ...
```

Veranschaulichung



- Original BLAST – kein Zusammenfügen mehrerer MSP
- Keine Beachtung von Gaps

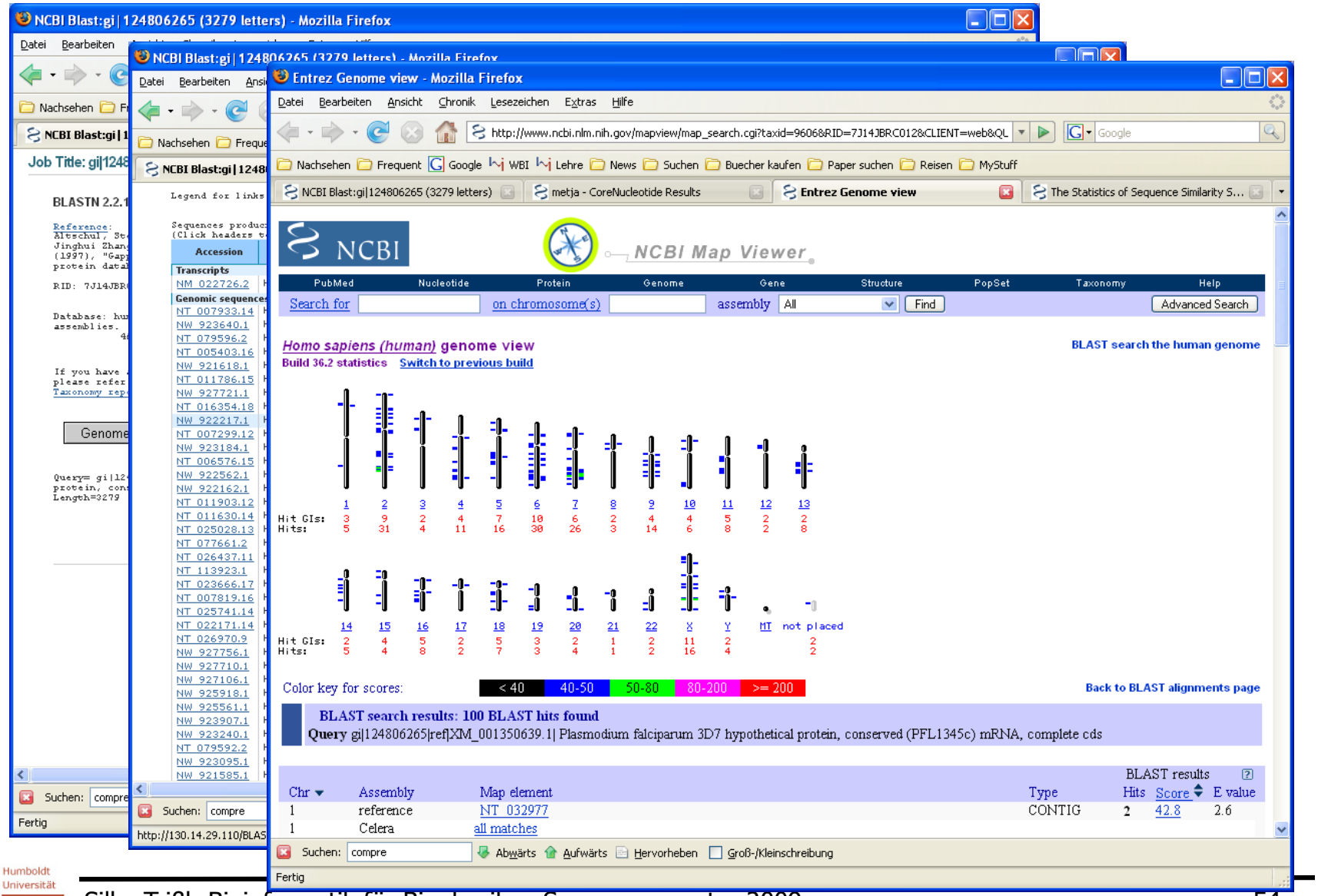
Bemerkungen

- BLAST ist eine Exklusionsmethode
  - Bestimme und suche nach Seeds = minimale Alignments, die (höchstwahrscheinlich) im Kern jedes sehr guten lokalen Alignments stecken
  - Erweitere nur diese zu Alignments für die gesamte Suchsequenz (bzw. deren „lokalen" Kern)
  - Praktisch alle Heuristiken zur Sequenzsuche arbeiten nach diesem Muster
    - Quasar, Biohunter, BLAT, FASTA, ...
- Heuristik
  - BLAST findet i.d.R. nicht alle hinreichend guten Alignments
  - Grund: Keine INSDEL, Schwellwerte w und t

Eigenschaften

- t vergrößern bei gleichem w (oder w und t proportional vergrößern)
  - Anforderungen an Seeds wachsen
  - Es werden weniger Seeds gefunden
  - Es werden weniger Hits verlängert
    - Performanz steigt
  - Gefundene Matches sind mit höherer Wahrscheinlichkeit gute lokale Alignments
    - Spezifität steigt
  - Aber mehr gute Alignments werden übersehen
    - Sensitivität sinkt

BLAST Screenshot



BLAST-2

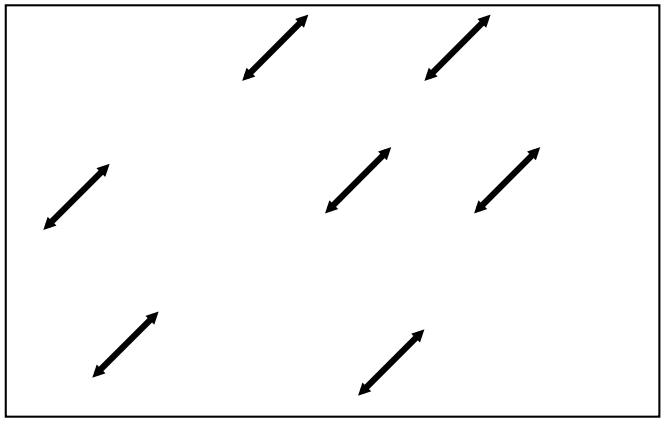
- 7 Jahre nach der Originalveröffentlichung
  - Altschul, Madden, Schaffer, Zhang, Zhang, Miller, Lipman: „Gapped BLAST and PSI-BLAST: a new generation of protein database search programs", NAR, 1997
- Zwei Verbesserungen
  - Performance verbessern
    - Denn: Sequenzdatenbanken wachsen schneller als Geschwindigkeit der Computer
  - Gaps beachten
    - Denn: Mehrere kurze Alignments mit Gaps werden von BLAST-1 übersehen, wenn keines davon signifikant bzgl. t ist
    - Zusammen können diese Alignments aber hochsignifikant sein
    - BLAST-1 liefert bei mehrere, nahe beieinander liegenden MSP auch mehrere Ergebnisse (statt einem größeren)

Zwei-Hit-Strategie

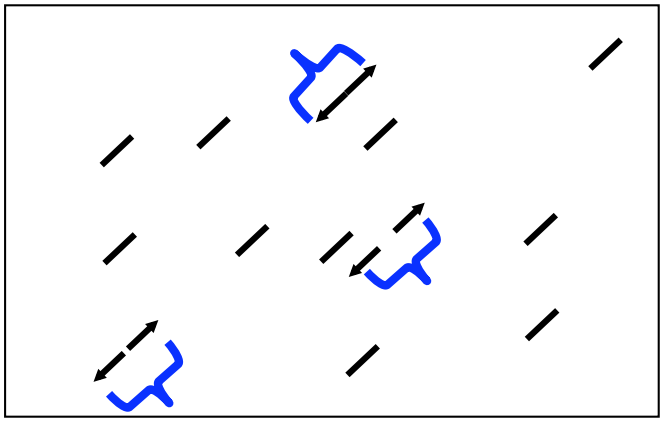
- Neue Strategie
  - Extension erfolgt nur, wenn zwei nicht-überlappende Hits auf einer Diagonale mit Abstand höchstens a gefunden wurden
  - Dadurch werden weniger Extensionen ausgeführt
    - Performance steigt
    - Sensitivität sinkt
  - Abhilfe: w/t verkleinern
- Ergebnis
  - Performance verdoppelt bei gleichbleibender Sensitivität

Illustration

Blast-1



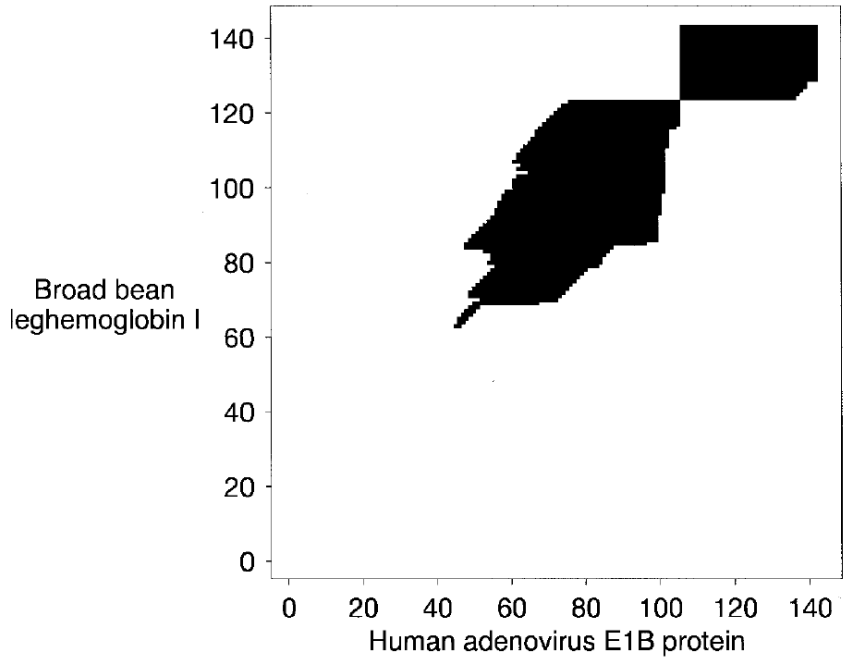
Blast-2



Gaps in BLAST-2

- Original: Hits werden verlängert ohne Gaps
- BLAST-2: Gapped Alignment
  - Wenn zwei Hits H_1, H_2 in Abstand a gefunden werden, wird in dieser Diagonale ein Smith-Waterman Alignment berechnet
  - Dazu sucht man das Sequenzstück zwischen H_1 und H_2 der Länge w mit dem höchsten Score ohne Gaps
  - Von diesem „Seedpoint" aus lokales Alignment berechnen
  - Da SW sensitiver ist als Extensionen ohne Gaps, kann man t wieder erhöhen (musste man wegen der zwei-Hits Strategie verringern)
- Weitere Performanceverbesserung trotz besserer Sensitivität
  - 500x langsamere Extension durch SW, aber 4000x weniger Extensionen durch Erhöhung von t von 11 auf 13

Lokales Smith-Waterman



- SW ausgehend vom Seed-Point in beide Richtungen
- Abbruch bei Unterschreiten bestimmter Schranken
  - Abhängig von bisherigen besten Treffern etc.

Zusammenfassung

- Heuristik zur Suche in Datenbanken
  - Wesentlich schneller als Smith-Waterman
  - Trotzdem gute Sensitivität
  - Signifikanz von Treffern durch e-Werte gemessen
  - De-Facto Standard zur Suche in Sequenzdatenbanken
- Viele weitere Heuristiken
  - FASTA: kaum noch in Gebrauch
  - BLAT: 500*schneller als BLAST bei gleicher Sensitivität; speziell für sehr ähnliche Sequenzen (EST Suche)
  - BioHunter, Quasar, Oasis, ...
    - Tricks: Nicht zusammenhängende Seeds, Seedpaare, spezielle Indexstrukturen zur Seedsuche, ...