Inhaltsverzeichnis

Kapitel 0

Überblick

0.1 Sequenzen

- Schwerpunkt:
  - Modellierung der Evolution
  - Alignmentverfahren + Bewertungsfunktionen (Scoringverfahren)
- Beispiel: CpG-Inseln
  - Beobachtung: Vorkommen von CG-Dinukleotiden seltener als erwartet
  - CpG-Inseln: CG-reiche Gegenden in Genen → häufig Promotoren oder Beginn von Genen
  - Wie erkenne ich CpG-Inseln? → stochastische Modellierung von DNA-Sequenzen ≡ "Vorgehen, um zufällig DNA-Sequenzen zu produzieren
  - 1. Modell: Würfel

    Produktion von DNA-Sequenzen durch Würfeln des 1., 2., ... Buchstaben

    Parameter: P_A, P_C, P_G, P_T ... Wahrscheinlichkeit A,C,G,T zu würfeln

    Bsp: Sequenz AACGAT → Wk = $P_A \cdot P_A \cdot P_C \cdot P_G \cdot P_A \cdot P_T = P_A^3 \cdot P_C \cdot P_G \cdot P_T$

    Problem: Bestimmung geeigneter Parameter

    ↪ setze P_A, P_C, P_G, P_T auf relative Häufigkeit von A,C,G,T

    Begründung: relative Häufigkeiten = Maximum Likelihood Schätzung für Vorkommen von A,C,G,T im Genom

    Bsp: Sequenz AACGATT
    $P_A = \frac{3}{7}, P_C = \frac{1}{7}, P_G = 17, P_T = 27$
    Wahrscheinlichkeit, diese Sequenz zu würfeln =
    $P_A^3 \cdot P_T^2 \cdot P_C \cdot P_G = \frac{27 \cdot 4}{7^7} = \frac{108}{7^7}$
    ↪ maximale Wahrscheinlichkeit, andere Parameterwahl keine Verbesserung

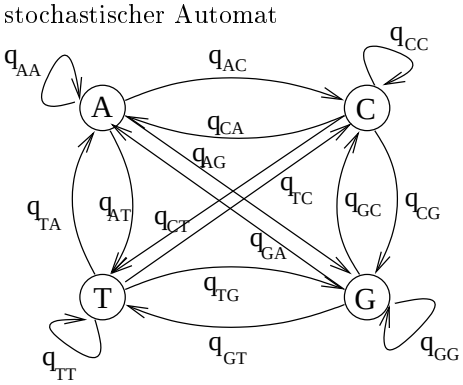
    Problem: Modell nicht für CpG-Inseln geeignet

↪ 2. Modell: Markov-Modell/ Markov-Kette/ Markov-Prozeß

* Beginn: Würfeln des 1. Buchstaben mit P_A, P_C, P_G, P_T

↪ jeder Folgebuchstabe abhängig vom Buchstaben davor
$q_{AA}, q_{AC}, q_{AG}, q_{AT}, \ldots, q_{TG}, q_{TT}$

⇒ stochastischer Automat



bei gegebenen Wahrscheinlichkeiten → Wahrscheinlichkeit für ACGAT

$$p_{(ACGAT)} = p_A \cdot q_{AC} \cdot q_{CG} \cdot q_{GA} \cdot q_{AT}$$

↪ bei CpG-Inseln: Wahrscheinlichkeit für q_{CG} hoch

- Markovketten - Anwendung:
  - Modellierung der Evolution → Bewertungsfunktion für Alignment
  - Hidden-Markov-Modelle → multiples Sequenzalignment
  - Modellierung der Evolution → Stammbaumrekonstruktion
- Alignmentverfahren
  - Beispiel:

    in bisherigen Verfahren erlaubte Mutationen

    ↪ Einfügen/ Löschen/ Ersetzen

    ↪ Einfügen von 5 einzelnen Gaps gleichwertig zu einem 5er-Gap

A	C	A	A	_	G	_	A	_	G	_	T	_	T	A	C	...
A	C	A	A	G	G	T	A	C	G	T	T	A	T	A	C	...
								≡								
A	C	A	A	G	A	G	T	_	_	_	_	_	T	A	C	...
A	C	A	A	G	A	G	T	C	G	T	T	A	T	A	C	...

  - Problem:

    Alignmentverfahren kodieren Annahmen über Evolution

    Evolution funktioniert anders

    ↪ Einfügen von einem 5er-Gap wahrscheinlicher
  - Zusammenhang zu Markovketten:

    Verwendung in PAM-Matrix (Scorefunktion)

effiziente Behandlung von Gaps → GOTOH-Algorithmus
lokales Sequenzalignment

- multiples Alignment (Vergleich mehrerer Sequenzen)
  - Beispiel:

    $a_1 =$ A C _ G A T
    $a_2 =$ A G G G _ T
    $a_3 =$ A C _ G A C

  - Herangehensweise:

    Definition verschiedener Proteinmotive
    Motive: Profiles, Pattern, Hidden-Markov-Modelle

  - Problem:
    - sehr komplexes Optimierungsproblem ⇒ NP-vollständig

    ↪ heuristische Verfahren (Suchverfahren)
  - Bemerkung:

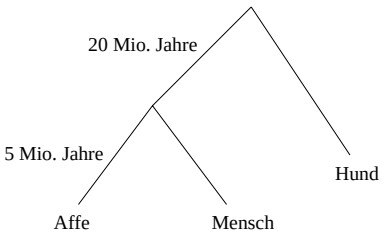
    Evolution erlaubt Duplikationen ≡ Tandem-repeats

    ↪ keine Alignierung mit Standardverfahren möglich
    gegebene mRNA → Finden des passenden Genabschnittes

    ↪ Problem: zu starke Betrachtung von Introns!

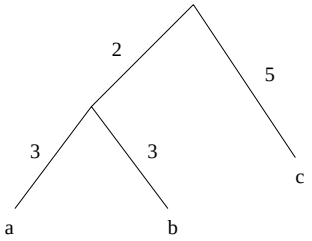
0.2 Stammbaumrekonstruktion

- Stammbaum:



- alte Methode → morphologische Vergleiche
- neue Methode → Bioinformatikmethoden auf Proteinsequenzen
- 1. Ansatz:
  - Proteinsequenzen und deren Distanzen gegeben

  ↪ Clusteringverfahren
  - Beispiel: 3 Sequenzen a, b, c → Distanzen mit Alignmentverfahren bestimmen
    $D(a,b) = 6, D(a,c) = 10, D(b,c) = 10$
    ↪



– Distanz muß Ultrametrik sein !

↪ Kombination Alignment mit Stammbaumrekonstruktion

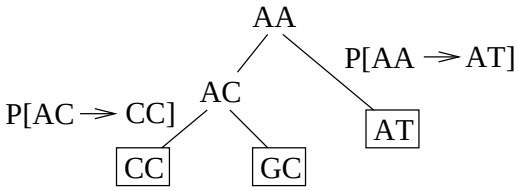
↪ multiples Alignment

- 2. Ansatz:

  – direkte Modellierung des evolutionären Prozesses

  ↪ Maximum-Likelihood-Schätzungen und Markov-Prozesse

  – Beispiel:



  ↪ Betrachtung aller Kombinationen für Blätter und Knoten

0.3 Strukturen

0.3.1 Sekundärstrukur von RNA

- intramolekulare Basenpaarung

↪ Welche WATSON-CRICK-Bindungen werden realisiert?

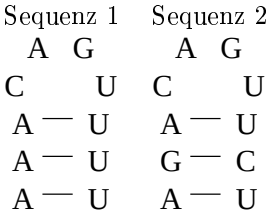
- Bsp.:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	U	A	C	G	A	U	U	A	C	C	G	U	A	U

U U A
A C
G — C
C — G
A — U
U — A
A — U

↪ dynamische Programmierverfahren (NUSSINOV/ZUKER)

0.3.2 Alignment von RNA-Strukturen

- Problem: Struktur höher konserviert als Sequenzalignment → Sequenzalignment funktioniert nicht
- 

↪ Struktur muß berücksichtigt werden ⇒ noch ungelöst

0.3.3 Proteinstruktur

- 3 große Themenbereiche

1. ab initio Strukturvorhersage (nur aus DNA-Sequenz) → schwere Optimierungsprobleme
2. Protein Threading
   - Ausgangspunkt: 2 verwandte Sequenzen (für eine Struktur bekannt)
   - beste Struktur für 2. Sequenz
   - Idee: konservierte Strukturelemente → wo in Sequenz 2

   ↪ Prinzip: Alignment
   - Einfluß der Struktur → Alignment mit Berücksichtigung struktureller Beziehungen (z.B. elektrostatische Wechselwirkung)

   ⇒ KEINE dynamische Programmierung möglich
3. homologiebasierte Modellierung (aus Informationen über verwandte Sequenzen) → biologisches Wissen

Kapitel 1

Sequenzen

1.1 Wahrscheinlichkeit, Statistik, Markovketten

1.1.1 Stochastische Grundlagen

- Wahrscheinlichkeitsraum:
  - Ereignisraum $\Omega \equiv$ Menge von Elementarereignissen
  - Ω heißt diskret, falls Ω endlich bzw. abzählbar unendlich

  Bsp.: Ω für Siedlerspiel $= \{(1,1),(1,2),...,(6,6)\}$
  - Ereignis: $A \subseteq \Omega$

  Bsp.: A = "Summe ist 7" $= \{(1,6),(6,1),(2,5),(5,2),(3,4),(4,3)\}$
  - elementare Wahrscheinlichkeitsfunktion $p : \Omega \to [0..1]$ mit $\sum_{e \in \Omega} p(e) = 1$
  - Wahrscheinlichkeit eines Ereignisses $A \subseteq \Omega Pr(A) = \sum_{e \in A} p(e)$

  Bsp.: Pr["Summe gerade"] $= \frac{18}{36}$
  Pr["Summe 2"] $= \frac{1}{36}$
  - Wahrscheinlichkeiten nicht bekannt $\to$ relative Häufigkeiten eines Ereignisses A bei n Versuchen $\frac{n(A)}{n}$
- bedingte Wahrscheinlichkeit:
  - $Pr[A \mid B] \equiv$ Wahrscheinlichkeit des Ereignisses A, unter der Bedingung, dass B eingetreten ist

  $\hookrightarrow Pr[A \mid B] = \frac{Pr[A \cap B]}{Pr[B]}$

  Bsp.: A = "Summe ist 7", B = "Summe ist ungerade"
  $Pr[A \mid B] = \frac{\frac{6}{36}}{\frac{18}{36}} = \frac{1}{3}$
- unabhängige Ereignisse:
  - $Pr[A \mid B] = Pr[A] \equiv Pr[A \cap B] = Pr[A] \cdot Pr[B]$

Bsp. 1: A_1 ="1. Zahl durch 3 teilbar"
B_1 ="Summe ist ungerade"
Ist $Pr[A_1 \mid B_1] = [A_1]$?
$Pr[A_1] = \frac{12}{36} = \frac{1}{3}$
$Pr[A_1 \mid B_1] = \frac{6}{18} = \frac{1}{3}$
$\hookrightarrow$ A_1 und B_1 unabhängig

Bsp. 2: A_2 ="Summe kleiner gleich 3"
B_2 ="Summe ist ungerade"
$Pr[A_2] = \frac{3}{36} = \frac{1}{12}$
$Pr[A_2 \mid B_2] = \frac{2}{18} = \frac{1}{9}$
$\hookrightarrow$ A_2 und B_2 abhängig, da $\frac{1}{12} \neq \frac{1}{9}$

- disjunkte Ereignisse:
  - sich ausschließende Ereignisse
  - $A \cap B = \emptyset$

  $\hookrightarrow$ falls $A, B \neq \emptyset$ folgt aus Disjunktheit Abhängigkeit
  $Pr[A \cap B] \neq Pr[A] \cdot Pr[B]$

- Formel der totalen Wahrscheinlichkeit:
  - $B_1, ..., B_k$ gegenseitig disjunkte Ereignisse
  - $\Omega = B_1 \uplus ... \uplus B_k$

  $\hookrightarrow$ $Pr[A] = \sum_{l=1}^{k} Pr[A \mid B_l]$

- Bayessche Regel:
  - $Pr[B \mid A] = \frac{Pr[A|B] \cdot Pr[B]}{Pr[A]}$
  - Nebenrechnung:
    1. $Pr[B \mid A] = \frac{Pr[A \cap B]}{Pr[A]}$
    2. $Pr[A \mid B] = \frac{Pr[A \cap B]}{Pr[B]}$

  2. in 1. $Pr[B \mid A] = \frac{Pr[A|B] \cdot Pr[B]}{Pr[A]}$

- Bayessches Lernen:
  - Problem:
    - stochastisches Modell $M(\omega)$ mit Parametern ω (Vektor)
    - $M(\omega)$ produziert Beobachtungen O mit Wk $Pr[O \mid M(\omega)]$ (likelihood von O unter $M(\omega)$

    Bsp.: DNA-Sequenzen = Beobachtung O sind Strings
    $\hookrightarrow$ mögliches Modell = Würfel
    - schätze die Parameter ω zu gegebener Beobachtung O

    $\hookrightarrow$ Verwendung der Bayesschen Regel
    $Pr[M(\omega) \mid O] = \frac{Pr[O|M(\omega)] \cdot Pr[M(\omega)]}{Pr[O]}$
  - $Pr[M(\omega)]$ ist prior-Distribution

- $Pr[M(\omega) \mid O]$ ist aposterior-Distribution
- Parameterschätzung: Suche ω mit $Pr[M(\omega) \mid O]$ maximal

↪ Maximum aposterior-Schätzung

- Spezialfall:
  - $Pr[M(\omega)]$ uniform

↪ Maximum von $Pr[M(\omega) \mid O] = \frac{Pr[O|M(\omega)] \cdot Pr[M(\omega)]}{Pr[O]}$ unabhängig von

1. $Pr[O]$, da nur 1 Beobachtung
2. $Pr[M(\omega)]$, da Uniformität

↪ $\max(Pr[O \mid M(\omega)]) \Rightarrow$ "Maximum Likelihood"

Bsp.: Show, 1 Preis, 3 Türen → Wahl einer Tür
zufälliges Öffnen einer anderen Tür
interessanter Fall: nichts gefunden → neue Entscheidung
↪ Modell $M(\omega)$ mit ω = yes (Preis hinter meiner Tür), bzw. ω = no
Parameterschätzung:

1. Maximum Likelihood
   ↪ $Pr[O \mid M(yes)] = 1$
   ↪ $Pr[O \mid M(no)] = \frac{1}{2}$
   $\Rightarrow$ bleibe bei meiner Tür
   aber: Annahmen nicht erfüllt
2. Maximum Aposterior Schätzung
   $Pr[M(\omega) \mid O] = \frac{Pr[O|M(\omega)] \cdot Pr[M(\omega)]}{Pr[O]}$
   $Pr[M(yes)] = \frac{1}{3}, Pr[M(no)] = \frac{2}{3}$
   $Pr[M(yes) \mid O] = 1 \cdot \frac{1}{3} = \frac{1}{3}$
   $Pr[M(no) \mid O] = \frac{2}{3} \cdot \frac{1}{2} = \frac{1}{3}$
   $\Rightarrow$ keine Entscheidung möglich

- Zufallsvariable:
  - Funktion $X : \Omega \to \mathbb{R}$
  - diskrete Zufallsv.: $Pr[X = x] \equiv Pr[\{e \in \Omega \mid X(e) = x\}]$
  - kontinuierliche Zufallsv.: Dichtefunktion $p_X : \mathbb{R} \to \mathbb{R}$

    $Pr[a \leq X \leq b] = Pr[\{e \in \Omega \mid a \leq X(e) \leq b\}] = \int_a^b p_X(x)\, dx$

Bsp.: Ω = Menge der Würfe mit 2 Würfeln
X = Summe bei einem Wurf
$e_{Bsp} = (3, 4)$
$X(e) = 7$
$Pr[X = 7] = \frac{6}{36} = \frac{1}{6}$

- Erwartungswert:
  - diskret: $E(x) = \sum_{i=-\infty}^{\infty} iPr[X = i]$

Bsp.: erwartete Summe bei einem Wurf mit 2 Würfeln
$E(x) = 2 \cdot \frac{1}{36} + 3 \cdot \frac{2}{36} + ... + 12 \cdot \frac{1}{36} = 7$

- kontinuierlich: $E(x) = \int_{-\infty}^{\infty} x p_X(x)\, dx$
- $E(aX + bX) = aE(X) + bE(Y)$
- X,Y unabhängig $\rightarrow E(X \cdot Y) = E(X) \cdot E(Y)$

- Streuung, Varianz:
  - $V(x) = E((x - E(x))^2)$
    $= E(x^2 - 2E(x)x + E^2(x))$
    $= E(x^2) - 2E(x)E(x) - E^2(x) = E(x^2) - E^2(x)$

- Standardabweichung:
  - $\sqrt{V(x)}$

- uniforme Verteilung:
  - jedes Elementarereignis hat gleiche Wahrscheinlichkeit
  - diskret: $Pr[X = \omega] = \frac{1}{|\Omega|}$
  - kontinuierlich: X im Intervall $[a, b] \rightarrow$ Dichtefunktion: $p_X(x) = \begin{cases} \frac{1}{ba} \text{ falls } x \epsilon [a, b] \\ 0 \text{ sonst} \end{cases}$

    $Pr[c \leq X \leq d] = \frac{d-c}{b-a}$

- Bernoulli-Verteilung:
  - $Y = \Omega \rightarrow \{0, 1\}$
  - n Versuche
  - X = Anzahl der Erfolge $= Y_1 + Y_2 + ... + Y_n$

  $\hookrightarrow$ X ist binomialverteilt
  - $E(X) = E(Y_1 + Y_2 + ... + Y_n) = n \cdot p$

1.1.2 Markovketten

- Markov Ketten 1.Ordnung:

  Was ist eine Markov Kette?

  $\hookrightarrow$ stochastischer Prozeß
  - Zeitpunkt: t=0,1,2,...
  - Zufallsvariablen für Zeitpunkt t einer Variablen $\frac{q}{t}$(Werte aus Zustandsmenge Q)
  - Zustand zum Zeitpunkt T hängt von den Zuständen der Vorläufer ab.(d.h.: von den Werten $q_0 ... q_{t-1}$)
  - Wie formuliert man die Abhängigkeit(stochastisch)

  $\Rightarrow$ Verteilung angeben

$=_*$ Pr $[q_0 = S]$ für jedes $s\epsilon$ Q(Anfangsverteilung)

* Abhängigkeiten: bedingte Wahrscheinlichkeit.

$\Pr[q_{t+1} = s_{t+1} \mid q_t = S_t ... q_0 = S_0]$

- Besondere Eigenschaften der Markov-Kette:

  $\Rightarrow$ geschichtslos

  $\Rightarrow$ $\Pr[q_{t+1} = S_{t+1} \mid q_t = S_t ... q_0 = S_0] = \Pr[q_{t+1} = S_{t+1} \mid q_t = S_t]$

  – Zeithomogenität: $\Pr[Q_{t+1} = S_{t+1} \mid q_t = S_t]$ ist unabhängig von t

  – Bsp.: Wetter als Markov-Kette

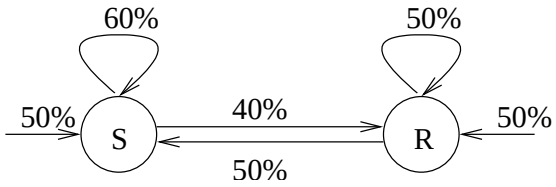
  (Nach "Die Zeit":60% Vorhersagekorrekt)

  * Zeitpunkt: Tage

  * Zustände Q={S,R}Wetter an den Tagen

  * Vorstellung: Wetter von heute hängt stark von dem von gestern ab.

- Modell:



  – Anfangsverteilung:

  $\Pr[q_0 = S]$=0,5

  $\Pr[q_0 = R]$=0,5

  – Folgeverteilung:

  $\Pr[q_1 = S \mid q_0 = S]$=0,6 $\Pr[q_1 = R \mid q_0 = S]$=0,4

  $\Pr[q_1 = S \mid q_0 = R]$=0,5 $\Pr[q_1 = R \mid q_0 = R]$=0,5

  $\Rightarrow$ Folgewahrscheinlichkeiten(Übergangswahrscheinlichkeiten werden in Matrizenform angegeben.)

  $$\begin{pmatrix} 0,6 & 0,4 \\ 0,5 & 0,5 \end{pmatrix}$$

  * Interessante Frage: absolute Wahrscheinlichkeit für S(Sonne) und R(Regen) $\Rightarrow$ stationäre Verteilung

  Wahrscheinlichkeit für S zum Zeitpunkt t=1

  $=$ $\Pr[q_1 = S \wedge q_0 = R] + \Pr[q_1 = S \wedge q_0 = S]$

  $=$ 0,25+0,3=0,55

Im folgendem:Q={1,2,...,n}(Q,endlich durchnummeriert)
Definition: Eine zeithomogene Markov-Kette 1.Ordnung ist ein Triple (Q,π,P), wobei

Q={1,...,n} eine Zustandsmenge,

$\pi=(\pi_1,...\pi_n)$ ein Zeilenvektor(Anfangsverteilung) mit π_i=Pr$[q_o = i]$

und P=$(P_{ij})_{\substack{1\leq i\leq n\\ 1\leq j\leq n}}$ eine stochastische Matrix

von Übergangswahrscheinlichkeiten ist, mit

P_{ij} =Pr$[q_{t+1} = j \mid q_t = i]$

=Pr$[q_1 = j \mid q_0 = i]$

"Wahrscheinlichkeit, den Zustand j im nächsten Zeitpunkt zu haben, falls aktueller Zustand i ist."

Bemerkung.:

- Eine Matrix ist stochastisch, falls für alle Zeilen die Summe der Einträge 1 ist.
- In der Bioinformatik werden auch substochastische Matrizen zur Modellierung endlicher Zustandsfolgen(Sequenzen) eingesetzt.
- P substochastisch falls für jede Zeile die Summe der Einträge $leq1$.

  Bsp.: Purine(P) Pyrimidine(Y)

  $\Rightarrow$ a=1,2 P=$\begin{pmatrix} 0,6 & 0,3 \\ 0,5 & 0,2 \end{pmatrix}$

  $\Rightarrow$ modelliert Purin/Pyrimidinsequenz mit
  Pr[Purin $\mid Purin$]=0,6
  Pr[Pyrimidin $\mid Purin$]=0,3
  Pr[Sequenz endet $\mid Purin$]=0,1

- Stationäre Verteilung:
  - Frage: Wie berechnet man die Wahrscheinlichkeit für einen Zustand i zum Zeitpunkt t?

    $\pi^{(t)} = (\pi_1^{(1)}, ..., \pi_n^{(t)})$ mit $\pi_i^{(t)}$=Pr$[q_t = i]$

    1. $\pi^{(0)} = \pi$
    2. Betrachte $\pi^{(1)}$

    $\pi_i^{(1)}$ =Pr$[q_1 = i]$(Formel für totale Wahrscheinlichkeit)
    =$\sum_{j=1}^{n}$ Pr$[q_0 = j]$Pr$[q_1 = i \mid q_0 = j]$
    =$\sum_{j=1} \pi_j^{(0)} P_{ji}$
    $\Rightarrow \pi^{(1)} = \pi^{(0)} P$
    Beispiel.: Wettermodell

$$\text{P}=\begin{pmatrix}0,6 & 0,4\\ 0,5 & 0,5\end{pmatrix} \qquad \pi^{(1)} = \begin{pmatrix}0,3+0.25 & 0,2+0.25\end{pmatrix}$$
$$\pi = \begin{pmatrix}0,5 & 0,5\end{pmatrix} \qquad = \begin{pmatrix}0.55 & 0.45\end{pmatrix}$$

3. Betrachte $\pi^{(t)}$

$\pi_i^{(t)} = Pr[q_t = i]$

$= \sum_{j=1}^{n} Pr[q_{t-1} = j] Pr[q_t = i \mid q_{t-1} = j]$

$\Rightarrow$ daraus folgt

$\pi^{(t)} = \pi^{(t-1)} P$

$= \pi^{(t-2)} PP$

$\Rightarrow \pi^{(t)} = \pi P^{(t)}$

Bsp. Wettermodell:

$\pi^{(2)} = \pi^{(1)} P$

$= \begin{pmatrix}0,55 & 0,45\end{pmatrix} \begin{pmatrix}0,6 & 0,4\\ 0,5 & 0,5\end{pmatrix}$

$= \begin{pmatrix}0,33+0.225 & 0,445\end{pmatrix} = \begin{pmatrix}0,555 & 0,445\end{pmatrix}$

Vermutung: Je größer t, desto größer ist $\pi_1^{(t)}$ bei 0,5.

Frage: Mit welcher Wahrscheinlichkeit(absolut gesehen) hat man Sonne/Regen
$\Rightarrow$ stationäre Verteilung.

Sei M=(G, π, P)eine Markovkette 1.Ordnung mit der Eigenschaft, daß der Grenzwert von $\pi^* = \lim_{t\to\infty} \pi P^t$ existiert$(= \lim_{t\to\infty} \pi^{(t)})$
Dann heißt π^* die stationäre Verteilung von M

- Fragen:
  1. Wie/Wann weiß ich, daß eine stationäre Verteilung existiert?
  2. Wie kann ich von einer Verteilung testen, ob sie stationär ist?
  3. Wie bestimme ich sie?

Sei π ein Zeilenvektor, P eine Matrix.
π heißt linker Eigenvektor(zum Eigenwert 1) für P falls: πP=1P

Bem.: Stationäre Verteilungen, sind linke Eigenvektoren zum Eigenwert 1.

$\pi^* P = (\lim_{t\to\infty} \pi P^t) P$

$= \lim_{t\to\infty} \pi P^t P = \pi^*$

Wir werden zeigen, daß unter bestimmten Bedingungen auch die Umkehrung gilt.

π^* linker Eigenvektor für P

$\Downarrow$

π^* stationäre Verteilung von P

Bsp.:

- $\text{P}=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$ Alle Vektoren sind linke Eigenvektoren

$(0,375 \quad 0,625)\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix} = (0,375 \quad 0,625)$

- P=$\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$ hat einen linken Eigenvektor $(0,5 \quad 0,5)$ aber keine stationäre Verteilung.

  $(0,3 \quad 0,7)\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix} = (0,7 \quad 0,3) \Rightarrow \pi^t = \begin{cases}\text{t gerade: } (0,3 \quad 0,7)\\ \text{t ungerade: } (0,7 \quad 0,3)\end{cases}$

- P=$\begin{pmatrix}0,6 & 0,4\\ 0,6 & 0,4\end{pmatrix}$

  $\pi^* = (0,6 \quad 0,4)$ stationäre Verteilung

  $\pi^* P = (0,6 \quad 0,4)\begin{pmatrix}0,6 & 0,4\\ 0,6 & 0,4\end{pmatrix}$

  $= (0,6 * 0,6 + 0,6 * 0,4 \quad ...) = (0,6(0,6 + 0,4) \quad ...) = (0,6 \quad 0,4)$

- P=$\begin{pmatrix}0,8 & 0,2\\ 0,4 & 0,6\end{pmatrix}$ $\quad \pi^* = (\frac{2}{3} \quad \frac{1}{3})$ π^* ist linker Eigenvektor

  $\pi^* P = (\frac{2}{3} \quad \frac{1}{3})\begin{pmatrix}0,8 & 0,2\\ 0,4 & 0,6\end{pmatrix} = (\frac{2*0,8+1*0.4}{3} \quad \frac{0,2*2+0,6}{3})\ (\frac{2}{3} \quad \frac{1}{3})$

Bem.: Bsp $\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$ und $\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$ zeigem, daß möglicherweise die Nulleinträge Schwierigkeiten machen.

Satz: Sei P eine stochastische Matrix, die keine Nulleinträge hat. Dann ist $P^* = \lim_{t\to\infty} P^t$ von der Form, daß die Spalten nur gleiche Einträge haben.

$$P^* = \begin{pmatrix}p_1 & p_2 & ... & p_n\\ p_1 & p_2 & ... & p_n\\ p_1 & p_2 & ... & p_n\end{pmatrix}$$

Für jede Anfangsverteilung π: $\lim_{t\to\infty} \pi P^* = (p_1 \quad ... \quad p_i \quad ... \quad p_n)$

Bsp.:

$$P^* = \begin{pmatrix}0,6 & 0,4\\ 0,6 & 0,4\end{pmatrix} \quad \pi = (0,8 \quad 0,2)$$

$\pi P^* = (0,6 \quad 0,4)$

Beweisidee: Man beweist, daß die Differenz in jeder Spalte bei Multiplikation mit sich selbst abnimmt.

Bsp.: $\begin{pmatrix}0,8 & 0,2\\ 0,4 & 0,6\end{pmatrix}$

Differenz zwischen Minimum und Maximum jeder Spalte = 0,4

Betrachte P^2

$P^2 = \begin{pmatrix}0,8 & 0,2\\ 0,4 & 0,6\end{pmatrix}\begin{pmatrix}0,8 & 0,2\\ 0,4 & 0,6\end{pmatrix} = \begin{pmatrix}0,72 & 0,28\\ 0,56 & 0,44\end{pmatrix}$

Differenz jeder Spalte von P^2 jeweils nur noch 0,16

$P^*=\begin{pmatrix} 0,6 & 0,3 \\ 0,6 & 0,3 \end{pmatrix}$

Satz: Sei P eine Matrix ohne Nulleinträge und ϵ¿0 der minimale Eintrag. Dann gilt für Spalte j:

$max_j(P^*) - min_j(P^*) \leq (max_j(P) - min_j(P))(1-2\epsilon)^{t-1}$

$max_j(P)$ ist maximaler Eintrag von P in spalte j

Bem.: Je kleiner ϵ, desto langsamer wird die stationäre Verteilung erreicht.

Bsp.: P=$\begin{pmatrix} 0,8 & 0,2 \\ 0,4 & 0,6 \end{pmatrix}$ $\epsilon = 0,2$

$max_j(P^2) - min_j(P^2) = 0,16$

$0,16 \leq (max_j(P) - min_j(P))(1-2\epsilon)^1 == 0,4 * 0,6 = 0,24$

Abschätzung: $max_j(P^10) - min_j(P^10) \leq 0,4(0,6)^9$

$$max_j(P^10) - min_j(P^10) \leq 0,004$$

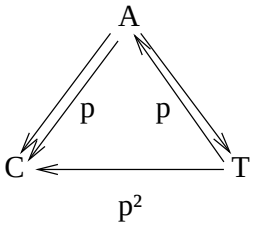
- Reversibilität:
  - Problem: Markov-Kette zur Darstellung der Evolution

    4 Buchstaben{a,c,g,t}

    4x4 Matrix mit Substitutionswahrscheinlichkeit

    z.B.: $\begin{pmatrix} 0,8 & 0,05 & 0,1 & 0,05 \\ 0,1 & 0,6 & 0,1 & 0,2 \end{pmatrix}$

    Alignment:



    A ist das Vorgängernukleotid. Für ein Alignment wird der Pfeil nun umgedreht.

    P heißt reversibel, falls dies geht.

P heißt reversibel, falls P die stationäre Verteilung π^* hat und für alle i,j gilt:

$$\pi^* P_{i,j} = \pi^*_j P_{j,i}$$

1.1.3 Anwendung von Markov-Ketten in PAM-Matrix

- Frage: Was sind odds?
- Bsp.: Sagt man:
  - Die Wahrscheinlichkeit, daß das Pferd "Gnadenbrot" gewinnt ist 0,001? oder
  - Die Chance, daß "Gnadenbrot" gewinnt stehen bei 1:100.
- Odds: Verhältnis von Gewinn zu Verlust

  odds=$\frac{2}{5} \Rightarrow Pr[Gewinn] = \frac{2}{2+5}$

  In der Statistik: Gewinn und Verlust nicht ausschließend

  Man spricht von 1-Hypothese und 0-Hypothese
- odds=$\frac{\text{1-Hypothese produziert Beobachtung}}{\text{0-Hypothese produziert Beobachtung}}$
- Bei PAM-Matrix bzw. Alignment:
  - 1-Hypothese:Alignment(S+S' evolutionär bedingt)
  - 0-Hypothese:Alignment(S+S' zufällig entstanden)
- PAM-Matrizen(Dayhoff):
  - Bewertung von Proteinsequenzalignments

    Ges: Alignmentgewicht: w(x,y) mit x,y Aminosäuren.

    Hier:w(x,y) Ähnlichkeit, nicht Distanz.

    Bsp.:

    A C A G
    A T A G

    sind sehr ähnlich(max. Ähnlichkeit) oder sehr wenig verschieden(min. Distanz)
- Vorgehensweise bei PAM:
  1. Bestimmung der Parameter für Bewertung(max.likelihood):

     suche bekannte/verwandte Sequenzen $S_1; S_2$ (Sequeähnlichkeit $\geq 85\%$)

     Bestimme aus Alignment von $S_1; S_2$ Parameter für Bewertungsfunktion
  2. Bedeutung unbekannter Alignments:

     Vergleich 1-Hypothese/0-Hypothese
  3. Übertragung auf entfernte Verwandschaft: Markov-Modelle

zu 2. Bed.: keine gaps, Sequenzen gleich lang

geg.: Sequenzen $a = a_1 \dots a_n, b = b_1 \dots b_n$

ges.: Wie gut ist das Alignment

a_1 a_2 a_3 $\dots$ a_n
b_1 b_2 b_3 $\dots$ b_n

Lsg.: - 0-Hypothese $\rightarrow$ keine evolutionäre Verwandtschaft $\rightarrow$ Ähnlichkeit zufällig

$\hookrightarrow$ Modell R: a, b unabhängig produziert und a_n, a_m unabhängig produziert

$\hookrightarrow$ Würfel mit 20 Seiten

$\equiv$ Verteilung über Aminosäuren
$q_x = Pr$[Aminosäure x an beliebiger Stelle]
Berechnung von q_x über relative Häufigkeit
$\Rightarrow$ Backgroundfrequency

$$Pr[Align(a,b)|R] = \prod_{i=1}^{n} q_{a_i} \prod_{i=1}^{n} q_{b_i} = \prod_{i=1}^{n} q_{a_i} q_{b_i}$$

- 1-Hypothese $\rightarrow$ evolutionäre Verwandtschaft
$\hookrightarrow$ Modell E mit Parameter p_{xy}
$p_{xy} \equiv$ Wahrscheinlichkeit, ein Paar x,y in einem Alignment homologer Sequenzen zu sehen

$$Pr[Align(a,b)|E] = \prod_{i=1}^{n} p_{a_i b_i}$$

$$\Rightarrow S(a,b) = \frac{\text{1-Hypothese}}{\text{0-Hypothese}} = \frac{\prod_{i=1}^{n} p_{a_i b_i}}{\prod_{i=1}^{n} q_{a_i} q_{b_i}} = \prod_{i=1}^{n} \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}}$$

Umwandeln in additive Gewichte

$$\Rightarrow S'(a,b) = \log(\prod_{i=1}^{n} \frac{p_{a_i b_i}}{q_{a_i} q_{b_i}}) = \sum_{i=1}^{n} \log(\frac{p_{a_i b_i}}{q_{a_i} q_{b_i}})$$

$\log(\frac{p_{a_i b_i}}{q_{a_i} q_{b_i}}) \equiv$ Gewicht für eine Spalte

zu 1. Schätzung der Parameter p_{xy} (20x20 = 400), z.B. aus Alignment homologer Sequenzen $\rightarrow$ Teufelskreis

geg.: Alignment homologer Sequenzen s,s' ($|s| = |s'|$)
$\hookrightarrow$ Beobachtung $O \equiv Align(s,s')$
$\Rightarrow$ Maximum-Likelihood-Schätzung

ges.: $\arg\max Pr[O|E]$

Lsg.: $Pr[O|E] = \prod_{i=1}^{n} p_{s_i s'_i}$
Bestimmung von $p_{xy} \rightarrow$ partielle Ableitungen
Durchnummerieren von $p_{xy} : p_1 \ldots p_{400}$

$$f(p_1 \ldots p_{400}) = Pr[O|E] = \prod_{i=1}^{400} p_i^{n_i(s,s')}$$

$n_i(s,s') \ldots$ #Vorkommen des i-ten Paares in Align(s,s')
Bsp.: 1. Paar: A-A
$n_1(ACAG, AAAG) = 2$
partielle Ableitung:

$$\frac{\delta f(p_1 \ldots p_{400})}{\delta p_i} = n_i p_i^{n_i - 1} \prod_{j=1}^{i-1} p_j^{n_j} \prod_{j=i+1}^{400} p_j^{n_j} = \frac{n_i}{p_i} \prod_{j=1}^{400} p_j^{n_j}$$

Nebenbedingung:

$$\bigwedge_i 0 \leq p_i \leq 1 \rightarrow \text{schwer}$$

$$\sum_{i=1}^{400} p_i = 1 \rightarrow \text{leicht}$$

Lagrangscher Multiplikator
Wenn $f(p_1 \ldots p_{400})$ stetig und differenzierbar ist, so gibt es einen Extremwert (Sattelpunkt) bei $(p_1^e, \ldots, p_{400}^e)$ falls es ein λ, mit $(p_1^e, \ldots, p_{400}^e)$ Extrempunkte, gibt.

Lsg.: $L(p_1, \ldots, p_{400}, \lambda) = f(p_1, \ldots, p_{400} + \lambda g(p_1, \ldots, p_{400})$

$$= \prod_{i=1}^{400} p_i^{n_i} + \lambda(\sum_{i=1}^{400} p_i - 1)$$

$\hookrightarrow$ Extrempunkte von L durch partielle Ableitung

$\frac{\delta L(p_1,\ldots,p_{400},\lambda)}{\delta p_i} = \frac{n_i}{p_i} \prod_{j=1}^{400} p_j^{n_j} + \lambda = 0$

$\hookrightarrow p_i = \frac{n_i}{-\lambda} \prod_{j=1}^{400} p_j^{n_j}$

$\frac{\delta L(p_1,\ldots,p_{400},\lambda)}{\delta \lambda} = \sum_{i=1}^{400} p_i - 1 = 0$

$\hookrightarrow \sum_{i=1}^{400} p_i = 1$

$= \sum_{i=1}^{400} \frac{n_i \prod_{i=1}^{400} p_j^{n_j}}{-\lambda}$

$= \frac{\prod_{i=1}^{400} p_j^{n_j}}{-\lambda} \sum_{i=1}^{400} n_i = 1$, da $\frac{\prod_{i=1}^{400} p_j^{n_j}}{-\lambda}$ unabhängig

$\hookrightarrow -\lambda = \prod_{i=1}^{400} p_j^{n_j} (\sum_{i=1}^{400} n_i) = |s| \prod_{i=1}^{400} p_j^{n_j}$

$\hookrightarrow p_i = \frac{n_i \prod_{i=1}^{400} p_j^{n_j}}{|s| \prod_{i=1}^{400} p_j^{n_j}} = \frac{n_i}{|s|} \Rightarrow$ relative Häufigkeit

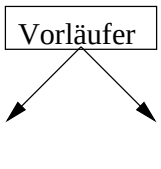
- Somit nun gelöst:
  1. Parameterschätzung für Parameter.
  2. Bewertung von Alignments unbekannter Sequenzen.
- Neues Problem:

  Paramterschätzung nur für festen evolutionären Abstand

  =Abstand der Sequenzen, die für Schätzung verwendet werden.
- Idee:

  Übertragung der Parameter auf weiter entfernte Proteine.

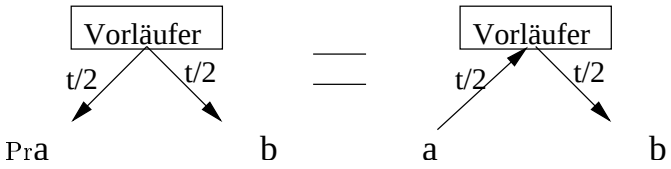
  $\Rightarrow$ Markov-Kette als Modell der Evolution.
- Bisher:



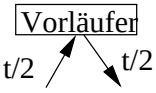
  Pr[Alignment(a,b)| E]= Unterschied für a,b unter Annahme eines gemeinsamen Vorläufers unter der Bedingung, dass E eingetreten ist.

  Problem: Vorläufer meist nicht bekannt.

A) Da Markov-Kette reversibel:



B) Aus Markov-Eigenschaft Evolution ist geschichtslos:



$\Rightarrow \Pr[a \xrightarrow{t} b] = \sum \Pr$

- Formulierung der Evolution als Markov-Modell:

  Anmerkung: Alle Positionen in der Sequenz unabhängig evolvieren.

  $\Rightarrow \Pr[a \xrightarrow{t} b] = \prod_{l=1}^{|a|} \Pr[a_l \xrightarrow{t} b_l]$

  * Wiederholung:Formale Definition der Markov-Kette

    M=(Q,π,P) Q={1,...,20}(AS) $\pi=(\pi_1, ..., \pi_20)$

    π_i=Pr[beliebige Position in Sequenz a hat die i-te AS]

    = Background frequency

    $= q_i$

    R=$(r_{i,j})$ $1 \leq i \leq 20$, $1 \leq j \leq 20$

    $r_{i,j}$=Pr[bel. Position wird die AS i in Sequenza durch AS j in Sequenzb ersetzt.]

    $\Rightarrow \Pr[a_l \rightarrow b_l] = \Pr[q_t = b_l \mid q_0 = q_l] = r_{a_l b_l}$

- Zusammenhang zwischen Paarwahrscheinlichkeit:$P_{i,j}$ (Ergebnis der Parameterschätzung) und Substitutionswahrscheinlichkeit $r_{i,j}$ in Markov-Kette

  P_{ij}=Pr[an bel. Stelle l das AS Paar i,j in Sequenz a und b zu haben]

  $\Pr[q_0 = a_l, q_t = b_l] = P_{a_l b_l}$

  $\Pr[q_t = b_l \mid q_0 = a_l] \Pr[q_0 = a_l] = \Pr[q_0 = a_l, q_t = b_l]$

  $r_{a_l b_l} * q_{a_l} = P_{a_l b_l}$

- Beispiel:

  S=AAACAACTCC

  S'=AAAGGAGACC

  Sequenzen als Beispiel für Parameterschätzung einer PAM unrealistisch, da normal Sequenzidentität von $\geq$ benutzt wird.

  * Background frequency:

    $q_A = \frac{5}{10}$ $q_C = \frac{2}{5}$

    $q_G = 0$ $q_T = \frac{1}{10}$

  * Paarwahrscheinlichkeit:

    $P_{AA} = \frac{4}{10}$ $P_{AG} = \frac{1}{10}$

    $P_{CG} = \frac{1}{5}$ $P_{CC} = \frac{1}{5}$ $P_{TA} = \frac{1}{10}$

    $\Rightarrow$ Paarwahrscheinlichkeitsmatrix:

    $$\begin{pmatrix} \frac{2}{5} & 0 & \frac{1}{10} & 0 \\ 0 & \frac{1}{5} & \frac{1}{5} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{1}{10} & 0 & 0 & 0 \end{pmatrix}$$

    $\Rightarrow$ Substitutions Matrix

$r_{xy} = \frac{P_{xy}}{q_x}$

$r_{AA} = \frac{P_{AA}}{q_A} = \frac{\frac{2}{5}}{\frac{1}{2}} = \frac{4}{5}$

$r_{AG} = \frac{1}{5} \quad r_{CG} = \frac{1}{2} \quad r_{CC} = \frac{1}{2} \quad r_{TA} = 1$

$\Rightarrow$ Substitutionswahrscheinlichkeits-Matrix

$$\begin{pmatrix} \frac{4}{5} & 0 & \frac{1}{5} & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

* Problem: Zero-Values(Substitutionen für die es keine Bsp. in den Daten gibt.
* Lösung: z.B.:Pseudocounts

  Statt Berechnung von P_{XY}durch$P_{XY} = \frac{n_{XY}(S,S')}{|S|}$ Berechnung von P_{XY} durch $P_{XY} = \frac{n_{XY}(S,S')+1}{|S|+16}$

  Idee ist an das Alignment alle möglichen Paare anzuhängen um Nullwerte zu umgehen.
* Problen: Hier generierte Matrix nicht symmetrich. Daher Betrachtung von $\frac{SS'}{S'S}$

– Übergang auf entfernte Verwandschaften

$P^{2t}_{i,j}$=Pr$[a_l(=i) \stackrel{2t}{\rightarrow} b_l(=j)]$

$= \Pr[q_{2t} = j, q_0 = i]$

$\Rightarrow$ zunächst berechnung von $r^{2t}_{i,j}$

$R^2 = (r^{2t}_{i,j}) \Rightarrow P^{2t}_{i,j} = r^{2t}_{i,j} * q_i$

Für beliebigen Abstand kt gilt:$R^k = (r^{kt}_{i,j})$

– PAM1: Entsteht aus Substitutions-Matrix durch Normalisierung auf 1% erwartete Anzahl von Ersetzungen.

Berechnung von R' aus R wie folgt:

$r_{XY} = \lambda r_{XY}$(mit X $\neq$ Y)

$r'_{XY} = 1 - \sum r'_{XY}$

Wähle λ so, dass 1% erwartete Substitution.

PAM250: Exponentiation mit 250

1.2 Alignments

- Kurze Wiederholung von Scoring-Funktion für Alignments

NEEDLEMAN/WUNSCH-Kostenfunktion:

– Zugrunde liegende Evolutionäres Modell:

Einfügen/Löschen und Substitution.

– Kostenfunktion:

w(x,y)=Kosten für Substitution von x,y

w(x,-)=Kosten für Löschen von x

w(-,y)=Kosten für Einfügen von y

- Kosten eines Alignments:

  D$\begin{pmatrix} A & A & C & A & G \\ A & A & - & A & G \end{pmatrix}$

  = Summe der Einzelkosten w(A,A)+...+w(G,G)=...

- Berechnung des besten Alignments

  $\Rightarrow D_{i,j}$=Kosten für bestes Alignment der Prefixe $a_1...a_i$ mit $b_1...b_j$

  $\Rightarrow D_{n,m}$= Kosten für bestes Alignment von a,b

- Rekursionsgleichung:

$$D_{i,j} = min \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ D_{i,j-1} + w(-, b_j) \\ D_{i-1,j} + w(a_i, -) \end{cases}$$

- Beispiel:

a=	A	A	A	A	_	_	_	_
b=	A	A	A	A	A	A	A	A

1)

A	_	A	_	A	_	A	_
A	A	A	A	A	A	A	A

2)

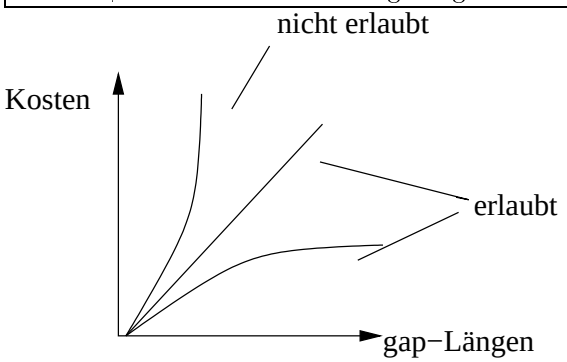
A	A	A	A	_	_	_	_
A	A	A	A	A	A	A	A

Für 1) sind die Kosten 4*w(_,A)

Für 2) sind die Kosten ebenfalls 4*w(_,A)

- Problem:
  - * evolutionäres Modell nicht korrekt
  - * Einfügen/Löschen: In der Wirklichkeit ist das Einfügen eines Gaps der Länge k leichter als k-mal Einfügen eines Gaps der Länge 1.
- Idee:Das Starten eines Gaps ist teuer, das Verlängern billig.

Eine Gap Penalty ist eine Funktion g(k):$N \Rightarrow R$ mit der eigenschaft, dass g subadditiv ist:
g(k+l)$\leq$g(k)+g(l)
g heisst affin falls: g(k)=$\alpha + k\beta$ wobei α die Kosten für gap-Einfügen ist und β die Kosten für Verlängerung



1.2.1 Algorithmus für Alignment mit Gap-Penalties

- Idee:
  - NEEDLEMAN/WUNSCH-Algorihtmus
  - Betrachtung der Matrix $D_{i,j}$ ... Kosten für bestes Alignment der Präfixe $a_1 \dots a_i$ mit $b_1 \dots b_2$
  - Rekursionsgleichung bei NEEDLEMAN/WUNSCH

$$D_{i,j} = \min \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ D_{i,j-1} + w(\_, b_j) \\ D_{i-1,j} + w(a_i, \_) \end{cases}$$

- Beispiel:

  $a = ACGAT$
  $b = AGCAT$

  $D_{2,3}$ ... kosten für bestes Alignment von AC und AGC

$$D_{2,3} = \min \begin{cases} D_{1,2} + w(C, C) \\ D_{2,2} + w(\_, C) \\ D_{1,3} + w(C, \_) \end{cases}$$

  ↪ mgliches Alignment:
  in letzter Spalte C,C
  in letzter Spalte _,C
  in letzter Spalte C, _

↪ Fallunterscheidung bei Rekursion ≡ Fallunterscheidung der letzten Spalte des besten Alignments

1. letzte Spalte = Substitution

   $\begin{array}{c|c} \dots & a_i \\ \dots & b_j \end{array}$

   Kosten: $D_{i-1,j-1}; w(a_i, b_j)$
   ↪ Gesamtkosten = $D_{i-1,j-1} + w(a_i, b_j)$
2. letzte Spalte = Insertion (Deletion analog)

   $\begin{array}{c|c} \dots & \_ \\ \dots & b_j \end{array}$

   Kosten: $D_{i,j-1}$; lokal g(1) (für Gap)
   aber: Anzahl der Gaps zu beachten
   ↪ Fallunterscheidung

   (a) $\begin{array}{cc|c} \dots & a_i & \_ \\ \dots & ? & b_j \end{array}$ ⇒ Gesamtkosten = $D_{i,j-1}$ + g(1)

   (b) $\begin{array}{cc|c} \dots & \_ & \_ \\ \dots & b_{j-1} & b_j \end{array}$ ⇒ Gesamtkosten noch nicht berechenbar
   ⇒ weitere Fallunterscheidung...

- Rekursionsgleichung (WATERMAN/SMITH/BEYER)

$$D_{i,j} = \min \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ \min \begin{cases} D_{i,j-1} + g(1) \\ D_{i,j-2} + g(2) \\ \vdots \\ D_{i,0} + g(j) \end{cases} \\ \min \begin{cases} D_{i-1,j} + g(1) \\ D_{i-2,j} + g(2) \\ \vdots \\ D_{0,j} + g(i) \end{cases} \end{cases}$$

- kompaktere Formulierung

$$D_{i,j} = \min \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ \min\limits_{1 \le k \le j} \{D_{i,j-k} + g(k) \\ \min\limits_{1 \le k \le i} \{D_{i-k,j} + g(k) \end{cases}$$

- Inititalisierung:
  - $D_{0,0} = 0$
  - $D_{i,0} = g(i)$
  - $D_{0,j} = g(j)$
- Berechnung:
  - Initialisierung
  - Tracematrix
    - für Alignment notwendig
    - Eintrag für Gap-Penalty und Substitution
    - $tr_{i,j} \subseteq \{\nwarrow, \leftarrow_k, \uparrow_l | \ k \le j, l \le i\}$
    - Traceback wie üblich (von rechts unten nach links oben, den Pfeilen folgend)
    - Erinnerung:
      Def. von $tr_{i,j}$
      $\nwarrow \in tr_{i,j} \leftrightarrow D_{i,j} = D_{i-1,j-1} + w(a_i, b_j)$
      $\uparrow \in tr_{i,j} \leftrightarrow D_{i,j} = D_{i-l,j} + g(l)$
      $\leftarrow \in tr_{i,j} \leftrightarrow D_{i,j} = D_{i,j-k} + g(k)$
- Komplexität:
  - Annahme: $| \ a \ | = | \ b \ | = n$
  - zu füllen: n^2 Zellen $\rightarrow O(n^2)$
  - Zeitaufwand für das Füllen einer Zelle nicht konstant $\rightarrow O(n)$

↪ Komplexität $= O(n^3)$

- Beispiel: durchschnittlicher Zeitaufwand für Zeile 1

  Spalte 1: 2 (oben & diagonal) + 1 (links)

  Spalte 2: 2 + 2

  $\vdots$

  Spalte n: 2 + n

  Summe: $2n + \frac{n(n+1)}{2}$

  ↪ durschnittliche Kosten für 1. Zeile
  $\frac{2n+\frac{n(n+1)}{2}}{n} = 2 + \frac{n+1}{2} \equiv O(n)$

- Beispiel:
  - 2 RNA-Sequenzen mit $n = 30000 = 3 \cdot 10^4$
  - $(3 \cdot 10^4)^3 = 27 \cdot 10^{12}$
  - Rechenzeit für einen Rechner mit 1 Ghz und 1 Operation pro Takt $\rightarrow 27000s \approx 7,5h$

- Verbesserung:
  - Problem: Gap-Penalties
  - Lösung: GOTOH's Algorithmus mit affinen Gap-Penalties

$$g(k) = \alpha + k \cdot \beta$$

  - Betrachtung von $D_{i,j}$
  - Fallunterscheidung nach letzter Spalte

1. $\begin{array}{c|c} \cdots & a_i \\ \cdots & b_j \end{array}$
2. $\begin{array}{c|c} \cdots & a_i \\ \cdots & - \end{array}$

   (a) $\begin{array}{cc|c} \cdots & ? & a_i \\ \cdots & b_j & - \end{array}$ Kosten $= D_{i-1,j} + g(1)$

   (b) $\begin{array}{cc|c} \cdots & a_{i-1} & a_i \\ \cdots & - & - \end{array}$ Kosten $= * + g(k) - g(k-1) = * + \alpha + k \cdot \beta - (\alpha + (k-1) \cdot \beta) = * + \beta$

   $*$ ... Kosten für bestes Alignment von $a_1 \ldots a_{i-1}$ mit $b_1 \ldots b_j$, das mit Gap in b endet

↪ Hilfsmatrizen

- P_{ij} ... Kosten füe bestes Alignment von $a_1 \ldots a_i$ mit $b_1 \ldots b_j$, das mit einem Gap in b endet
- Q_{ij} ... Kosten für bestes Alignment, das mit einem Gap in a endet
- Rekursionsgleichungen:

$$D_{ij} = \min \begin{cases} D_{i-1,j-1} + w(a_i, b_j) \\ P_{ij} \\ Q_{ij} \end{cases}$$

$$P_{ij} = \min \begin{cases} D_{i-1,j} + g(1) \\ P_{i-1,j} + \beta \end{cases}$$

$$Q_{ij} = \min \begin{cases} D_{i,j-1} + g(1) \\ Q_{i,j-1} + \beta \end{cases}$$

- Berechnung:
  1. Initialisierung
  2. `FOR i=1 ... |a|`
     `FOR j=1 ... |b|`
     `Berechne P(i,j);`
     `Berechne Q(i,j);`
     `Berechne D(i,j);`

- Komplexität: $O(n^2)$
  - n-faches Zurückgreifen bei Berechnung von D_{ij} entfällt
  - Komplexität von $P_{ij}, Q_{ij} = 4 =$ konstant
  - RNA-Sequenz mit n = 30000
    $\hookrightarrow (3 \cdot 10^4)^2 = 9 \cdot 10^8$
    Gigahetz-Prozessor benötigt ≤ 1 Sekunde

- Initialisierung:
  - $D_{0,0} = 0; D_{0,j} = g(j); D_{i,0} = g(i);$
  - Welche P_{ij}, Q_{ij} müssen initialisiert werden? $\rightarrow P_{0,j} \& Q_{i,0}$
  - $P_{0,j} \ldots$ Kosten für bestes Alignment mit $b_1 \ldots b_j$, das mit einem Gap in b endet

$\hookrightarrow$ $\begin{matrix} - & - & - & - & - \\ b_1 & b_2 & \ldots & b_j & - \end{matrix}$ Widerspruch (schlechtes Alignment)

$\hookrightarrow$ $P_{0,j} = \infty Q_{i,0} = \infty$

- Tracematrix
  - Ansatz: Füllen der Tracematrix wie bei NEEDLEMAN/WUNSCH

    $tr_{ij} \subseteq \{\nwarrow, \uparrow, \leftarrow\}$ mit
    $\nwarrow \in tr_{ij} \leftrightarrow D_{ij} = D_{i-1,j-1} + w(a_i, b_j)$
    $\uparrow \in tr_{ij} \leftrightarrow$ ein Gap in b $(D_{ij} = P_{ij})$
    $\leftarrow \in tr_{ij} \leftrightarrow$ ein Gap in a $(D_{ij} = Q_{ij})$
  - Problem: falscher Ansatz, da 3 Kostenmatrizen
  - Lösung: nicht 1 sondern 3 Tracematrizen mit für D, P, Q getrennten Elementen:
    $\{\nwarrow^D, \bullet^P, \bullet^Q, \uparrow^P, \uparrow^D, \leftarrow^Q, \leftarrow^D\}$
  - neue Definition der Tracematrix:

$\forall_{ij} > 0:$	$\nwarrow^D \in tr_{ij}^D \leftrightarrow D_{ij} = D_{i-1,j-1} + w(a_i, b_j)$
	$\bullet^Q \in tr_{ij}^D \leftrightarrow D_{ij} = Q_{ij}$
	$\bullet^P \in tr_{ij}^D \leftrightarrow D_{ij} = P_{ij}$
$\forall_{ij} > 0:$	$\leftarrow^D \in tr_{ij}^Q \leftrightarrow Q_{ij} = Di, j-1+g(1)$
	$\leftarrow^Q \in tr_{ij}^Q \leftrightarrow Q_{ij} = Q_{i,j-1} + \beta$
$\forall_{ij} > 0:$	$\uparrow^D \in tr_{ij}^P \leftrightarrow P_{ij} = D_{i-1,j} + g(1)$
	$\uparrow^P \in tr_{ij}^P \leftrightarrow P_{ij} = P_{i-1,j} + \beta$

- Beispiel GOTOH-Algorithmus
  - a = C C
  - b = A C C T
  - g(k) = 4 + k
  - $w(x.y) = \begin{cases} 1 & fallsx \neq y \\ 0 & sonst \end{cases}$
  - $D_{ij} =$

		A	C	C	T
	0	5	6	7	8
C	5	1	5	6	8
C	6	6	1	5	7

  $P_{ij} =$

		A	C	C	T
		∞	∞	∞	∞
C	?	10	11	12	13
C	?	6	10	11	13

  $Q_{ij} =$

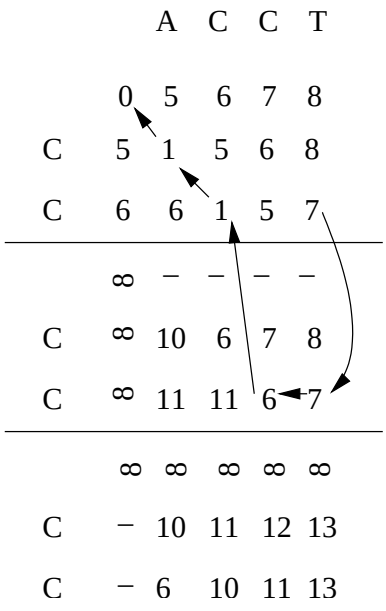
		A	C	C	T
		?	?	?	?
C	∞	10	6	7	8
C	∞	11	11	6	7

  - Traceback: $\leftarrow\nwarrow\nwarrow\leftarrow$

  $\hookrightarrow$ Alignment:

-	C	C	-
A	C	C	T

  Kosten = (4+1) + 0 + 0 + (4+1) = 10 $\rightarrow$ Widerspruch zu $D_{2,4} = 7$
  - eigentliches Traceback:



1.2.2 Lokale Sequenzalignments

- Idee: Man möchte nicht vollständige Sequenzen sondern nur noch Teile alignieren.
- Beispiel:

  CC mit ACCT

  ↪ perfekter lokaler Match bzw. Domäne
- Algorithmus muss leisten:

  ⇒ Finde Bereiche mit größter lokaler Übereinstimmung.

  (konservierte Domäne)

  → initiale und terminale Gaps bewertet man mit 0

  (keine Kosten)
- Problem:

  Distanz!

  Was ist das beste lokale Alignment von : $\begin{matrix} A & C & C & A & T & \\ G & G & A & G & G & G \end{matrix}$?

  $\begin{matrix} A & C & C & - & - & - & A \\ - & - & - & G & G & G & A \end{matrix}$ mit einem initial gap in a und einem initial gap in b ergibt 0

  noch schlimmer: Was sind die Kosten des besten lokalen Alignments von:

  $\begin{matrix} A & A & A & A \\ G & G & G & G \end{matrix}$?

A	A	A	A		-	-	-	-
-	-	-	-		G	G	G	G

Kosten=0 0 Kosten=0

Problem: $\begin{matrix} A & A & A & A \\ A & A & A & A \end{matrix}$ und $\begin{matrix} A & A \\ A & A \end{matrix}$ haben beide die gleichen Kosten 0!

- Lösung: Man verwendet statt der Distanz Ähnlichkeit der beiden Stränge.

Eine Ähnlichkeitsfunktion aus $\sigma \times \sigma$ ist eine Funktion S(x,y) $\sigma \times \sigma \to \mathbb{R}$ mit der Eigenschaft, dass S(x,x) positiv ist.

Bem.:

1. Negative Werte definieren Unähnlichkeit.
   Erinnerung: Pam-Matrix ist eine Ähnlichkeitsmatrix
2. Gaps werden häufig durch affine oder lineare Gap-Penalties festgelegt:
   affine: g(k)=$\alpha + \beta k$
   lineare: g(k)=δ k
   lineare gap penalty: Nach Definition: g(k+l)=g(k)+g(l) ⇒g(0)=0
   ⇒ S(_,x)=S(x,_)=δ
   Ähnlichkeit kann auf gap-Penalties erweitert werden.
3. Bei Ähnlichkeit haben gap-Penalties negative Werte!

1.2.3 Lokaler Sequenzalignment Algorithmus

SMITH-WATERMAN

- Wesentlicher Unterschied Distanz ↔ Ähnlichkeit

  Distanz ist Metrik
  ⇒w(x,x)=0
  Maß für evolutionäre Distanz

- Lokales Sequenzalignment

Aus Sequenzen $a_1....a_n; b_1....b_n$ folgt Matrix $(H_{i,j})_{\substack{1 \le i \le n \\ 1 \le j \le m}}$

$H_{i,j}$ = max Ähnlichkeit des globalen Alignments von $a_{k+1}...a_i$ mit $b_{l+1}...b_j (k \le i, l \le j)$
=max Ähnlichkeit eines lokalen Alignments von $a_1...a_i$ mit $b_1...b_j$, wobei sowohl a_i und b_j Teil des lokalen Alignments sind.

Lokales Sequenzalignment:$S_{loc}(a_1...a_n, b_1...b_n)$ =max$\{S(a_k...a_i, b_l...b_j)\}$
wobei S_{glob} globale Sequenzähnlichkeit.
"Bestes globales Alignment von Teilsequenzen von a,b"

- Eigenschaften von $H_{i,j}$

1. Kann $H_{i,j} \leq 0$ sein?

   Das beste Alignment ist Bestandteil jedes lokalen Alignments

   $\Rightarrow$ Wert des leeren Alignments:0

   $\Rightarrow H_{i,j} \geq 0$

2. Rekursionsgleichung für $H_{i,j}$:
   - 1. Fall: Das beste lokale Alignment das bei i,j startet $\Rightarrow H_{i,j}$=0
   - 2.Fall: Bestes lokales Alignment aligniert a_i mit b_j

     $\Rightarrow H_{i,j} = H_{i-1,j-1} + S(a_i, b_j)$
   - 3.Fall: a_i auf Gap $\Rightarrow H_{i,j} = H_{i-q,j} + S(a_i, \_)$
   - 4.Fall: b_j auf Gap $\Rightarrow H_{i,j} = H_{i,j-1} + S(\_, b_j)$

   $\Rightarrow$ Rekursionsgleichung:

$$H_{i,j} = max \begin{cases} H_{i-1,j-1} + S(a_i, b_j) \\ H_{i,j-1} + S(\_, b_j) \\ H_{i-1,j} + S(a_i, \_) \\ 0 \end{cases}$$

$$\text{Initialisierung: } \begin{cases} H_{0,0} = 0 \\ H_{i,0} = 0 \\ H_{0,j} = 0 \\ S_(x, y) = \begin{cases} 2 \text{ falls } x = y \\ -1 \text{ sonst} \end{cases} \\ \Rightarrow S(\_, x) = S(x, \_) = -1 \end{cases}$$

- Beispiel:

		A	C	A	C	C	T	T
	0	0	0	0	0	0	0	0
C	0	0	2	1	2	2	1	0
C	0	0	2	1	3	4	3	2
C	0	0	2	1	3	5	4	3

Wert des besten lokalen Alignments?

$\Rightarrow$ Welches Alignment ist zu nehmen?

Welche Indizes i,j?

$\Rightarrow$ Traceback und Ähnlichkeit des besten lokalen Alignments:

Suche i,j mit $H_{i,j}$ maximal

Hier:(3,5) mit $H_{3,5}$=5

Das Traceback des lokalen Alignments startet vom maximalen $H_{i,j}$, über normale Tracematrix, bis ein 0-Wert erreicht wird.

$\Rightarrow$: $\nwarrow \leftarrow \nwarrow \nwarrow$

dazugehöriges Alignment: $\begin{matrix} C & - & C & C \\ C & A & C & C \end{matrix}$ $\Rightarrow$ Wert:2+2+2-1=5

1.2.4 Einführung Multiple Alignments

- Alignmentbewertung:

  1. Distanz = evolutionäre Distanz

     $\Rightarrow$ Konstruktion vn phylogenetischen Bäumen durch Clustering.
  2. Ähnlichkeit: z.B. für Aminosäuren PAM

     "physikalisch chemische Kompatibilität"

  Lässt sich Distanz in Ähnlichkeit bzw umgekehrt umrechnen?

- Feng-Doolittle Umwandlung

  Ähnlichkeit in Distanz für multiples Sequenzalignment.

  Gegeben: 2Sequenzen a,b

  $S_{a,b}^{\text{obs(observed)}}$ = tatsächliche Ähnlichkeit von a, b

  $S_{a,b}^{max}$ =maximale Ähnlichkeit $=\frac{S_{a,a}^{obs}+S_{b,b}^{obs}}{2}$

  S_{rand} = Erwartungswert der Ähnlichkeit von Sequenz a',b' mit gleicher Komposition wie a,b.

  $\Rightarrow$ D(a,b)$=\frac{S_{a,b}^{obs}-S_{rand}}{S_{a,b}^{max}-S_{rand}}$

  - Das schlechteste was vorkommen kann D(a,b) liegt zwischen 0 und 1.

    $\Rightarrow$ prozentuale Ähnlichkeit zwischen evolutionär nicht verwandt(S_{rand}) und perfekt verwandt($S_{a,b}^{max}$)
  - D(a,b) geht exponentiell (langsam) gegen 0 mit ansteigender evolutionärer Distanz.

    $D'(a,b) = -log(D(a,b)) \Rightarrow D'(a,b)$ ist linear mit evol. Distanz.

- Problem des multiplen Sequenzalignment

  Paarweises Sequenzalignment zwar gut für Homologiesuche doch was ist mit der Frage:Was sind biologisch relevante Teile eines Proteins?

  - Bindungsstellen
  - konservierte Position

  Geht mit paarweisem Alignment nicht

  $\rightarrow$ entweder zu homolog oder zu weit entfernt.

- Fragen zum multiplen Sequenzalignment:

  1. Was ist das(formale Definition)
  2. Warum?
  3. Wie wird es bewertet?
  4. Wie wird es berechnet?

→ heuristische Verfahren liefern ein (hoffentlich gutes Ergebnis(güte unbekannt)

- Formale Definition:

Seien $a^1 ... a^N$ N-Sequenzen. Ein multiples Alignment von $a^1 ... a^N$ ist eine Matrix $(A_{i,j})$ $1 \leq i \leq N$(zeilen) $1 \leq j \leq K$(spalten) mit: $\forall_{i,j} A_{i,j} \epsilon \Sigma \bigcup \{\_\}$ $\forall_i A_{i,1} ... A_{i,k} \mid_\Sigma$ (eingeschränkt auf Σ) $= a^i$ $A_{i,j} = \_$

- Bewertung:

  1. Annahme: Spalten unabhängig (wie bei paarweisen Alignment)

$$S(A_{ij}) = \sum_{Spaltenj} S_c(A_{1,j} \dots A_{N,j}) \quad S_c \dots \text{Score für Column}$$

  2. Annahme: richtige Bewertung konservierter Spalten
  3. Annahme: Berücksichtigung evoutionärer Verwandtschaft

- Ansatz: Sum of Pairs

  – Idee:

    Score aus Summe des paarweisen Alignments

    ↪

$$S(A_{ij} = \sum_{Spaltenj} S_c(A_{\bullet j})$$

$$S_c(A_{1,j} \dots A_{N,j}) = \sum_{1 \leq k < l \leq N} s(A_{kj}, A_{lj})$$

    s ... übliche Ähnlichkeit (PAM) mit affiner Gap-Penalty

    Beispiel:

  – Vergleich: dreifaches multiples Alignment mit paarweisen Alignment

    log odds score: $s(x,y) = -\log(\frac{p_{xy}}{q_x \cdot q_y})$

    ↪ $s(x,y,z) = -\log(\frac{p_{xyz}}{q_x \cdot q_y \cdot q_z}$

    Sum of Pairs: $s(x,y) = -\log(\frac{p_{xy}}{q_x \cdot q_y}) - \log(\frac{p_{yz}}{q_y \cdot q_z}) - \log(\frac{p_{xz}}{q_x \cdot q_z})$

    ↪ kein Zusammenhang zu paarweisen Alignment ⇒ keine statistische Begründung für Sum of Pairs

  – Beispiel:

L G N A _	L G N A
_ L N A G	L N A G
_ L G G N	L G G N

    Wieviel schlechter ist ein G in konservierter L-Spalte versus einer kompletten L-Spalte?

BLOSSUM 90: s(G,L)=-4, s(L,L) = 5

↪

$$S_c(L, L, \ldots, L) = \sum_{1 \le k < l \le N} 5 = \frac{5 \cdot (N-1) \cdot N}{2}$$

$$\begin{aligned} S_c(G, L, \ldots, L) &= -4 \cdot (N-1) + \sum_{1 \le k < l \le N-1} 5 \\ &= S_c(L, L, \ldots, L) - (N-1) \cdot (s(L,L) - s(G,L)) \\ &= S_c(L, L, \ldots, L) - 9(N-1) \end{aligned}$$

$$\begin{aligned} \text{rel. Differenz} &= \frac{S_c(L, L, \ldots, L) - S_c(G, L, \ldots, L)}{S_c(L, L, \ldots, L)} \\ &= \frac{9 \cdot (N-1)}{\frac{5 \cdot N \cdot (N-1)}{2}} = \frac{18}{5 \cdot N} \end{aligned}$$

↪ geht gegen Null für große N → biologisch falsch!

- Berechnung multipler Alignments

1. Ansatz: Multidimensionales Programmieren

   am Beispiel des Alignments von 3 Sequenzen

   $a_1 \ldots a_n, b_1 \ldots b_m, c_1 \ldots c_r$

   ↪ $D_{ijk} \ldots$ bestes Alignment der Profile $a_1 \ldots a_i, b_1 \ldots b_j, c_1 \ldots c_k$

   Bewertung einer Spalte $S_c(x, y, z)$ → lineare Gap-Penalties (affine Gap-Penalties zu kompliziert)

$$D_{ijk} = \max \begin{cases} D_{i-1,j-1,k-1} + S_c(a_i, b_j, c_k) \\ D_{i-1,j-1,k} + S_c(a_i, b_j, \_) \\ D_{i-1,j,k-1} + S_c(a_i, \_, c_k) \\ D_{i,j-1,k-1} + S_c(\_, b_j, c_k) \\ D_{i-1,j,k} + S_c(a_i, \_, \_) \\ D_{i,j-1,k} + S_c(\_, b_j, \_) \\ D_{i,j,k-1} + S_c(\_, \_, c_k) \end{cases}$$

   Vorteil: beliebige Spaltenbewertung S_c möglich (nicht notwendigerweise Sum of Pairs)

   Nachteil: Darstellung (z.B. beim Traceback) → Rechenaufwand $O(N^N)$ → exponentiell in N

2. Ansatz: Heuristische Verfahren für multiples Alignment

   liefern eine Lösung (nicht unbedingt die optimale)

   möglichst schnell (polynomial $O(N^k)$ mit k fest)

   möglichst gut (Optimum nicht garantiert, Distanz zum Optimum unbekannt)

   Grund:

   – multiples Alignment NP-vollständig

- NP ... $\underline{p}$ olynomiale Zeit auf $\underline{n}$ ichtdeterministischen Turingmaschinen → (meist) nicht polynomialer Zeitaufwand (sondern exponentiell)
- vollständig ... Problem $p \in NP$, in dem man jede Instanz von jedem $p' \in NP$ in eine Instanz von p übersetzen kann
- normalerweise keine schnelle Lösung (s. Multidimensionales Programmieren)

Aufbau: Score → Sum of Pairs

Idee:

- erst alle paarweisen Alignments

↪ Zeitaufwand: $\frac{N \cdot (N-1)}{2} \cdot O(n^2) = O(N^2 \cdot n^2)$

- Zusammenfassen der Alignments, geordnet nach Distanz der Paare → Guide Tree

⇒ progressives Alignment

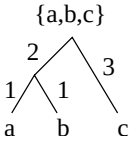
Guide Tree ... phylogenetischer Baum aus paarweisen Distanzen

Beispiel:

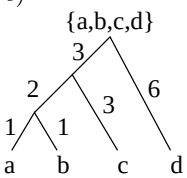
- Sequenzen a,b,c,d
- Distanzen: D(a,b) = 2
  D(a,c) = 6
  D(a,d) = 6
  D(b,c) = 12
  D(b,d) = 12
  D(c,d) = 12

- UPGMA-Verfahren (hierarchisches Clustering)
  a) Suche nächste Verwandte → a,b
  Bestimme Distanzen $\{a,b\}$ zu c und d
  $D_{\{a,b\},c} = \frac{D_{a,c}+D_{b,c}}{2} = 6$
  $D_{\{a,b\},d} = \frac{D_{a,d}+D_{b,d}}{2} = 12$

b)

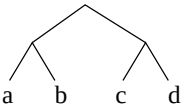


c)



↪ Alignment von a,b ≡ Profile $\{a,b\}$

Profile ... multiples Alignment

Guide Tree anders (z.B: 

) → anderes Profile

1.2.5 Verfahren für multiple Alignments

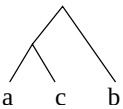
1. FENG/DOOLITTLE-Verfahren

- Ausgangspunkt:
  - Proteinsequenzen
  - PAM-Ähnlichkeit
  - affine Gap-Penalty
- Algorithmus:
  - Berechne Distanzen aus den Ähnlichkeiten für paarweise Alignments

  ↪ Guide Tree mit UPGMA

  ↪ Generiere sukzessive multiples Alignment (nach Guide Tree)
  → Ersetze Gap im paarweisen Alignment durch X
  ↪ Score: s(•, X) = 0
  - Prinzip: "Once a gap, always a gap."
- Beispiel:
  - 3 Sequenzen: a = ACCAT
    b = ACGGAT
    c = AACCAT
  - Score: $s(x,y) = \begin{cases} 1 & \text{falls } x = y \\ -1 & \text{sonst} \end{cases}$
  - paarweises Alignment:

a ↔ b = 2	A	C	_	C	A	T
	A	C	G	G	A	T
b ↔ c = 0	A	C	G	G	A	T
	A	A	C	C	A	T
a ↔ c = 4	_	A	C	C	A	T
	A	a	C	C	A	T



↪ Guide Tree: a c b

↪ Erstellen eines multiplen Alignments
Starte mit a,c
Ersetze gaps durch X

- Gruppe 1: $\begin{bmatrix} X & A & C & C & A & T \\ A & A & C & C & A & T \end{bmatrix}$
- Füge b hinzu
  alle paarweisen Alignments von b mit Elementen der Gruppe 1

bestes Alignment → multiples Alignment

a' ↔ b = 2	X	A	C	_	C	A	T
	_ A	C	G	G	A	T	
b ↔ c = 0	A	C	G	G	A	T	
	A	A	C	C	A	T	

- Nutze $a' \leftrightarrow b$, füge in jedem Element neue gaps aus a' ein (hier an der 4. Position)

$$\hookrightarrow \begin{bmatrix} X & A & C & \_ & C & A & T \\ A & A & C & \_ & C & A & T \\ \_ & A & C & G & G & A & T \end{bmatrix} \rightarrow \begin{bmatrix} X & A & C & X & C & A & T \\ A & A & C & X & C & A & T \\ X & A & C & G & G & A & T \end{bmatrix}$$

- Zusammenfassen von 2 Gruppen:
  - paarweise Alignierung der Elemente der 1. Gruppe mit der 2. Gruppe

  ↪ bestes Alignment bestimmt Zusammenfassung
- Problem:
  - Größe (bzw. Konserviertheit) einer Gruppe unwichtig (da bestes paarweises) Alignment bestimmend
  - Beispiel:

$$\begin{bmatrix} I & A & C & L \\ V & A & C & I \end{bmatrix} \text{ und } \begin{bmatrix} I & A & C & L \\ V & A & C & I \\ X & A & C & Y \\ G & A & C & V \\ W & A & C & F \end{bmatrix} \text{ gleichwertig}$$

↪ Verbesserung: progressive Verfahren mit Profiles

1.2.6 Hidden Markov Modelle in multiplen Alignments

- Wie wird Information eines multiplen Alignments repräsentiert anhand des Beispiels:

A	T	A	A	T
A	_	C	A	T
A	T	G	A	T
C	T	T	A	T

1. Reguläre Ausdrücke(notwendige Bedingung)

   ⇒[AC][T_]•AT ⇒ CCAT

   •= alles einsetzbar
2. Bessere Variante: Profile

   Berücksichtigt Häufigkeiten der Buchstaben in jeder Spalte

	1	2	3	4	5
A	3/4	_	1/4	1	_
C	1/4	_	1/4	_	_
G	_	_	1/4	_	_
T	_	3/4	1/4	_	1
_	_	1/4	_	_	_

   Qualität einer Sequenz:

Wahrscheinlichkeit: P(CCAT)=P(C_CAT)=1/64

P(ATAAT)=9/64

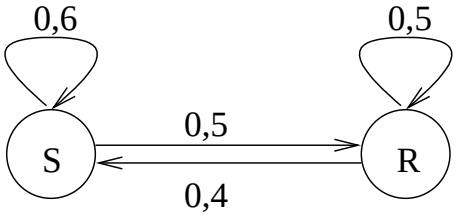
3. Noch bessere Repräsentation:
   Hidden-Markov-Modelle(HMMs)

- Was sind HMMs?

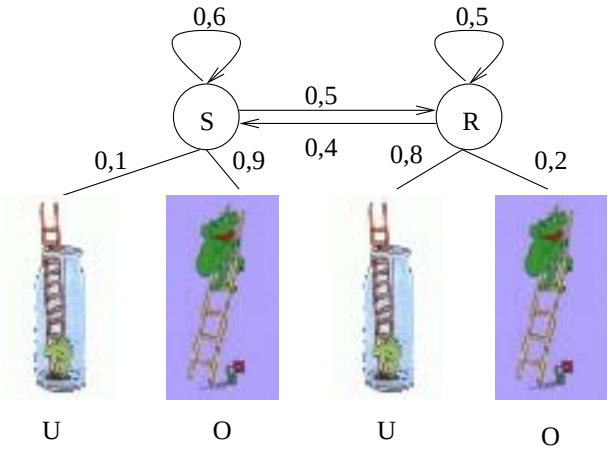
  HMMs sind Markov-Modelle mit versteckten Zuständen

  – Beispiel:

  Sonne-Regen-Modell:



  Markov-Modell:



  u=unten o=oben

Ein HMM 1. Ordnung = (Q,Σ,pi,A,B)
Q={1....n} Zustände
Σ={1....m} Beobachtungen
π={π_1....π_n} Anfangswert A=$(a_{ij})_{\substack{1 \le i \le n \\ 1 \le j \le n}}$ Übergangsverteilung
B=$(b_{ik})_{\substack{1 \le i \le n \\ 1 \le k \le m}}$ Ausgabewert
b_{ik}= Wahrscheinlichkeit für Ausgabe von Symbol k in Zustand

- Beipiel: Sonne/Regen:

  S→1 R→2

  O→1 U→2

$$A=\begin{pmatrix}0,6 & 0,4\\ 0,5 & 0,5\end{pmatrix} \qquad B=\begin{pmatrix}0,9 & 0,1\\ 0,2 & 0,8\end{pmatrix}$$

- Klassische Probleme bei HMMs

  1. Gegeben: HMM Wahrscheinlichkeit einer Beobachtung.
     Ist UUUUUOOUUUUUOO wahscheinlich?
  2. Gegeben: Beobachtung und HMM.
     Gesucht: Wahrscheinlichst Zustandsequenz
     U U U U U O O U U U U U O O war R R R R R S S R R R R R S S die (wahrscheinlichst) Ursache?

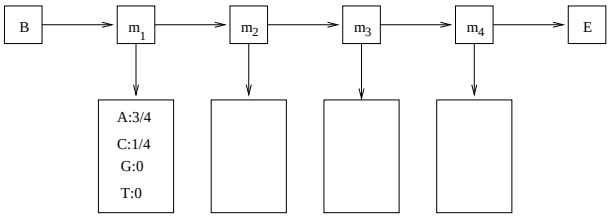
Lösung Dynamisches Programmieren(Viterbi-Algorithmus)

  3. Klimaveränderung
     Gegeben: Beobachtungen(Daten)
     Gesucht: Bestes HMM(Beste Parameter π,A,B für gegebenes Q,Σ)
     $\Rightarrow$ Maximum Likelihood Schätzung(leider micht berechenbar)
     $\Rightarrow$ Expectation Maximization

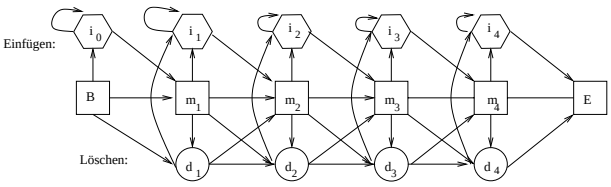
- Frage: HMM wie Profile?(Selbe Eigenschaft)

  Ausgabealphabet Σ={A,C,G,T} Q=$\{m_1, m_2, m_3, m_4\}$(m=match)

- Profile HMM($m_1, ..., m_4$ repräsentieren Vorläufersequenz):



- Einfügen und Löschen am Beispiel Vorläufersequenz mit 4 Nukleotiden:



- Multiples Alignment mit HMM:

  1. Aligniere Sequenzen mit HMM
  2. Diese Alignments bestimmen dann mult. Alignment

- Beispiel:

  geg.: 3 Sequenzen GGCT, ACCGAT, CT

  1. Aligniere Sequenz mit HMM

⇒ Suche Zustandspfad der Sequenz, die mit höchster Wahrscheinlichkeit produziert wird.

Annahme:

GGCT $\rightarrow B, m_1, m_2, m_3, m_4, E$

ACCGAT $\rightarrow B, i_0, m_1, d_2, m_3, i_3, i_3, m_4, E$

CT $\rightarrow B, m_1, d_2, d_3, m_4, E$

Dazugeöriges multiples Alignment:

		*	*	*			*
1	•	G	G	C	•	•	T
2	a	C	_	C	g	a	T
3	•	C	_	_	•	•	T
		m_1	m_2	m_3			m_4
	i_0	m_1	d_2	m_3	i_3	i_3	m_4
		m_1	d_2	d_3			m_4

- Trainieren von HMMs

  2 Möglichkeiten

  1. Geg,; n Repräsentanten der Familie

     → Baum-Welsch-Expectation Maximization

  2. n-Sequenzen
     ↓
     multiples Alignment
     ↓
     markiere m-Spalten
     ↓
     Schätze HMM Parameter

- Beispiel zu 2:

  multiples Alignment:

_	G	C	C	G	A	T	→	m_1	i_1	m_2	m_3	i_3	m_4
_	G	G	C	_	_	T	→	m_1	i_1	m_2	d_3	m_4	
A	C	_	C	G	A	T	→	i_0	m_1	m_2	m_3	i_3	m_4
-	C	-	-	-	-	-	→	m_1	d_2	d_3	m_4		
	*		*	*		*							

  – Bestimme für jede Sequenz die HMM Zustände.
  – Schätze Übergangswahrscheinlichkeiot und Ausgabewahrsch.

    ⇒ rel. Häufigkeit eines Übergangs bzw. Ausgabe.

    $a_{qq'} = \frac{A_{qq'}}{\sum_r A_{qr}}$

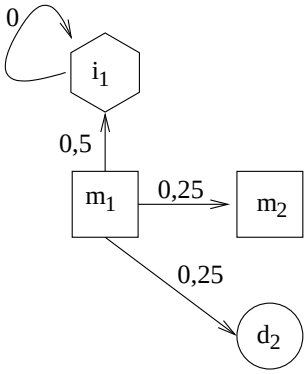
    $A_{qq'}$=Anzahl der Übergänge q→q'(in geg. Zustandsfolgen)

    Beipiel:

    $A_{m_1 i_1} = 2 \qquad A_{m_1 m_2} = 1 \qquad A_{m_1 d_2} = 1$

$\Rightarrow a_{m_1 i_1} = 2/4 \qquad a_{m_1 m_2} = 1/4 \qquad a_{m_1 d_2} = 1/4$

$\Rightarrow$ Ausschnitt aus HMM:

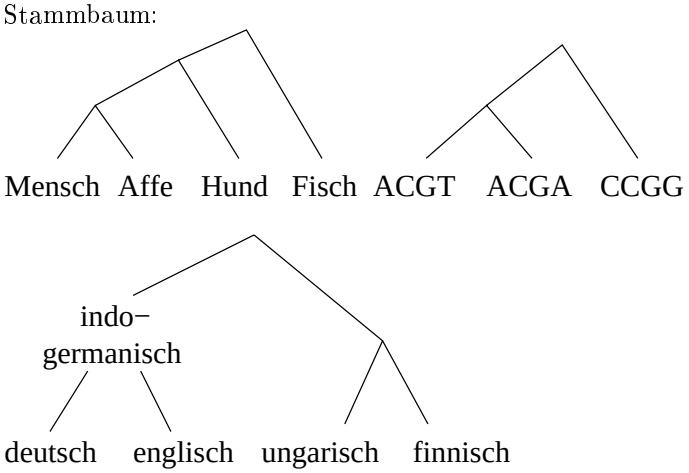


Kapitel 2

Stammbaumrekonstruktion

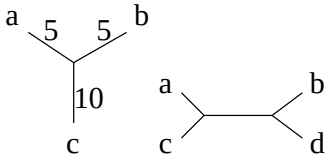
2.1 Einführung

- Stammbaum:

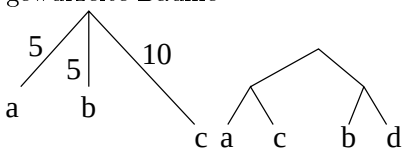


- Anwendung der Stammbaumrekonstruktion:
  - multiples Alignment
  - Arten-Stammbaum
  - Finden krankheitsbedingter Gene
- Begriffe:
  - taxon/taxa ... zu ordnende Begriffe
  - binärer Baum ... Stammbaum
  - n Taxa an n Blättern
  - Kanten mit Distanzen versehen
- Baumtypen:

1. ungewurzelte Bäume



2. gewurzelte Bäume



- Verfahren:
  - Clustering-Verfahren

    Eingabe: n Taxa (Sequenzen von DNA/Protein) und $n \times n$ Distanzmatrix $(D_{ij})_{\substack{1 \le i \le n \\ 1 \le j \le n}}$

    hierarchisches Clustering

    $\hookrightarrow$ UPGMA, WPGMA

    $\hookrightarrow$ FARRIS-transformed method

    neighbourhood joining, Parsimony
  - Maximum-Likelihood-Verfahren

    Eingabe: n Taxa und evolutionäres, statistisches Modell

    FELSENSTEIN (kontinuierlicher Markov-Prozeß)

    Quadtreesampling (heuristische Implementierung von Felsenstein)

2.2 Clustering-Verfahren

2.2.1 Hierarchisches Clustering

- Eingabe: Distanzmatrix für n Taxa
- Bsp.: n= 5 $\{a, b, c, d, e\}$

	a	b	c	d	e
a	0	6	10	10	10
b		0	10	10	10
c			0	2	6
d				0	6
e					0

- Baum: Summe am Pfad $i \to j = D_{ij}$

D_{ij} ist eine additive Baummetrik, falls es einen Baum t gibt, so dass

$$\forall_{ij} \sum \text{Distanzen am Pfad i} \to j = D_{ij}$$

t realisiert D_{ij}
D_{ij} heißt Ultrametrik, falls t die Bedingung erfüllt, dass alle Distanzen Wurzel $\to$ Blätter gleich sind.

1. UPGMA

- Unweighted Pair Group Method with Arithmetic mean
- Initalisierung:
  - Input $n \times n$ Matrix D_{ij}
  - C ... Menge der n Singleton Clusters $\{1\} \ldots \{n\}$
  - $\forall_{i,j}$ dist$(\{i,j\} = D_{ij}$
- Rekursion: repeat n-1
  - bestimme Paar von Cluster $c, d \in C$ mit minimaler Distanz d_{min} =dist(c,d)
  - definiere neuen Cluster $e = c \cup d$
    $C = C \cup \{e\} \backslash \{c\}\{d\}$
  - definiere neuen Knoten e mit Töchtern c,d
    Distanz zu Blättern $\frac{d_{min}}{2}$
  - Distanzfunktion

$$dist(e,f) = \frac{dist(c,f) + dist(d,f)}{2}$$

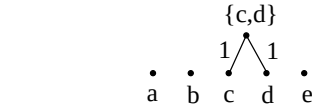
- Bem.: WPGMA analog, aber Distanzfunktion:

$$dist(e,f) = \frac{|c| \cdot dist(c,f) + |d| \cdot dist(d,f)}{|c| + |d|}$$

- Beispiel:
  - $C = \{\{a\},\{b\},\{c\},\{d\},\{e\}\}$
  - dist $= D_{ij} =$

	a	b	c	d	e
a	0	6	10	10	10
b		0	10	10	10
c			0	2	6
d				0	6
e					0

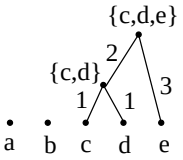
  - Schleife:
    - $\{c\},\{d\}$ minimale Distanz $\rightarrow d_{min} = 2$
      $C = \{\{a\},\{b\},\{c,d\},\{e\}\}$



      neue Distanzmatrix

	$\{a\}$	$\{b\}$	$\{c,d\}$	$\{e\}$
$\{a\}$	0	6	10	10
$\{b\}$		0	10	10
$\{c,d\}$			0	6
$\{e\}$				0

    - $\{c,d\}$ mit $\{e\} \rightarrow d_{min} = 6$
      $C = \{\{a\},\{b\},\{c,d,e\}\}$

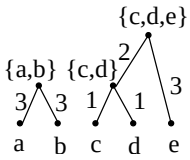


neue Distanzmatrix

	$\{a\}$	$\{b\}$	$\{c,d,e\}$
$\{a\}$	0	6	10
$\{b\}$		0	10
$\{c,d,e\}$			0

* $\{a\}$ mit $\{b\} \rightarrow d_{min} = 6$

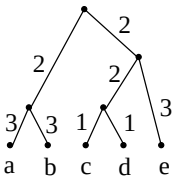
$C = \{\{a,b\},\{c,d,e\}\}$



neue Distanzmatrix

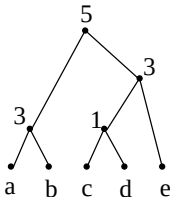
	$\{a,b\}$	$\{c,d,e\}$
$\{a,b\}$	0	10
$\{c,d,e\}$		0

* Wurzel $\{a,b,c,d,e\}$



UPGMA liefert immer einen ultrametrischen Baum.
Falls D_{ij} ultrametrisch ist, liefert UPGMA den richtigen Baum.

- alternative Darstellung für ultrametrischen Baum

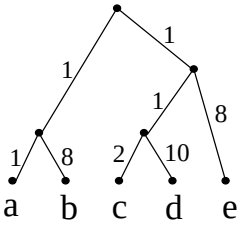


- biologische Annahme: alle Spezies gleichschnelle Evolution → "molecular clock"

↪ unwahrscheinlich

- keine Ultrametrik → Folgen
  ⇒ richtige Topologie, aber falsche Distanzen
  ⇒ oder falsche Topologie
- Beispiel:

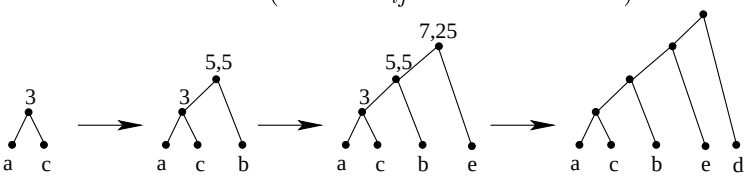
– Ursprungsbaum:



↪ $D_{ij} =$

	a	b	c	d	e
a	0	9	6	14	11
b		0	13	21	18
c			0	12	11
d				0	19
e					0

– Baum durch UPGMA (obwohl D_{ij} keine Ultrametrik)



2. FARRIS Transformed Distanz Method

Sei D_{ij} eine Distanzmatrix (additiv, nicht ultrametrisch), dann ist die transformierte Distanz definiert durch

$$E_{ij} = \frac{D_{ij} - D_{ir} - D_{jr}}{2} + \overline{d_r}$$

wobei $\overline{d_r}$ die durchschnittliche Distanz Wurzel $\leftrightarrow$ Blätter ist.

↪ Anwendung von UPGMA auf $E_{ij} \rightarrow$ richtiger Baum

- Beispiel:
  - D_{ij} siehe oben
  - $D_{ar} = 2 \quad D_{cr} = 4 \quad D_{er} = 9$
    $D_{br} = 9 \quad D_{dr} = 12$
  - $\overline{d_r} = \frac{2+9+4+12+9}{5} = \frac{36}{5}$

↪ $E_{ij} =$

	a	b	c	d	e
a	0	$\frac{31}{5}$	$\frac{36}{5}$	$\frac{36}{5}$	$\frac{36}{5}$
b		0	$\frac{36}{5}$	$\frac{36}{5}$	$\frac{36}{5}$
c			0	$\frac{26}{5}$	$\frac{31}{5}$
d				0	$\frac{31}{5}$
e					0

- Problem: r meistens nicht bekannt

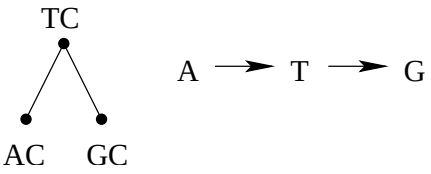
↪ Lösung: Verwenden einer outgroup (weit entfernt)

2.2.2 Maximum Likelihood Bäume

1. FELSENSTEIN-Verfahren

- Problem der Clusteringverfahren:
  - Distanz aus Alignment

  ↪ keine Berücksichtigung multipler Ersetzungen



  ↪ keine wirklichen Distanzen

- Lösung:
  - direkte Beschreibung der Evolution als stochastisches Modell

  ↪ wie bei PAM-Markovkette
  - Problem: Zeit t bei Markovkette fest

  ↪ Matrix für jeden Zeitpunkt gesucht

- allgemeine Form:
  -
$$P(t) = \begin{pmatrix} P_{AA}(t) & \dots & P_{AT}(t) \\ \vdots & & \\ P_{TA}(t) & \dots & P_{TT}(t) \end{pmatrix}$$
  - mit Markoveigenschaft
  - P(t) müssen stochastisch sein

$$\forall_t : \sum_y P_{xy}(t) = 1$$

$$\forall_t : 0 \leq P_{xy}(t) \leq 1$$

  - $P_{xy}(t) \dots$ Wahrscheinlichkeit, dass Nukleotid y an einer Stelle i nach t Zeiteinheiten zu haben ist, falls x an Stelle i zum Zeitpunkt 0 ist

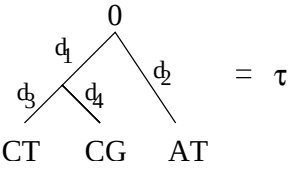
$$\begin{array}{ll} \text{t} = 0 & \text{x} \\ & \vdots \\ \text{t} & \underbrace{y}_{\text{Stelle i}} \end{array}$$

  ↪ kontinuierliche Zeitpunkte ⇒ Markovprozesse

- Zusammenhang statistisches Modell und Baum
  - Beispiel:

    Sequenzen: a = CT, b = CG, c = AT

    möglicher Baum:

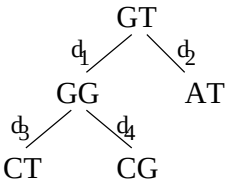


$= \tau$

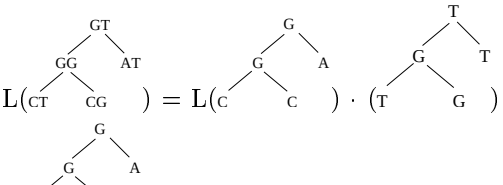
↪ Likelihood eines Baumes:

$$Pr[Seq(a,b,c)|\tau] = L(\tau)$$

- Berechnung der Wahrscheinlichkeit $Pr[Seq(a,b,c)|\tau]$
  - Betrachtung aller Variationen für 0,1
  - Bedingung:
    alle Taxa und interne Knoten haben gleiche Länge
  - Begründung:
    evolutionäres Modell erlaubt nur Substitutionen
  - Beispiel:



  - Annahme:
    Stellen (Positionen) unabhängig



↪ $\mathrm{L}(\;) = \Pi_{GA} P_{GA}(d_2) \cdot P_{GG}(d_1) \cdot P_{GC}(d_3) \cdot P_{GC}(d_4)$

  - komplette Berechnung:
    alle Variationen für 0,1

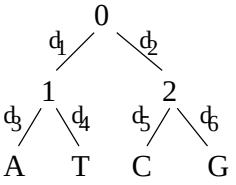
↪ $\mathrm{L}(\;) = \sum_{S_0 \in \{A,C,G,T\}} \sum_{S \in \{A,C,G,T\}} \Pi_{S_0} P_{S_0 T}(d_2) P_{S_0 S_1}(d_1) P_{S_1 T}(d_3) P_{S_1 G}(d_4)$

- Komplexität:
  - n Taxa → n-1 innere Knoten

↪ Anzahl Summanden: $4^{n-1} \rightarrow O(4^{n-1})$

  - exponentiell

- Verbesserung:
  - Umordnen der Terme → Struktur: [( )( )]
  - Beispiel:



$$\begin{aligned}
& \sum_{S_0}\sum_{S_1}\sum_{S_2}\Pi_{S_0}\cdot P_{S_0S_1}(d1)\cdot P_{S_0S_2}(d2)\cdot P_{S_1A}(d3)\cdot P_{S_1T}(d4) \\
& \qquad \cdot P_{S_2C}(d_5)\cdot P_{S_2G}(d6) \\
= \quad & \sum_{S_0}\sum_{S_1}\Pi_{S_0}\cdot P_{S_0S_1}(d1)\cdot P_{S_1A}(d3)\cdot P_{S_1T}(d4)\cdot\sum_{S_2}P_{S_0S_2}(d2) \\
& \qquad \cdot P_{S_2C}(d5)\cdot P_{S_2G}(d6) \\
= \quad & \sum_{S_0}[\Pi_{S_0}\cdot(\sum_{S_2}P_{S_0S_2}(d2)\cdot P_{S_2C}(d5)\cdot P_{S_2G}(d6))\cdot(\sum_{S_1}P_{S_0S_1}(d1) \\
& \qquad \cdot P_{S_1A}(d3)\cdot P_{S_1T}(d4))]
\end{aligned}$$

  - Lösung: Dynamisches Programmieren

2. Dynamisches Programmieren

$L_{k,S}$ ... Likelihood für Teilbaum unter Knoten k, falls mit S belegt

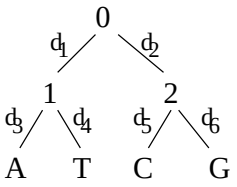
- Initialisierung:
  - für jedes Blatt k

$$L_{k,s} = \begin{cases} 1 & \text{falls von k = S} \\ 0 & \text{sonst} \end{cases}$$

- Rekursionsgleichung:
  - k ... interner Knoten

$$L_{k,S} = (\sum_{S'} P_{SS'}(d_i)\cdot L_{k',S'})(\sum_{S''} P_{SS''}(d_j)\cdot L_{k'',S''})$$

- Beispiel:



  - Initialisierung:

$$\begin{aligned}
& L_{3,A}=1, L_{3,C}=0, L_{3,G}=0, L_{3,T}=0 \\
& L_{4,T}=1, L_{4,C}=0, L_{4,G}=0, L_{4,A}=0 \\
& L_{5,C}=1, L_{5,A}=0, L_{5,G}=0, L_{5,T}=0 \\
& L_{6,G}=1, L_{6,C}=0, L_{6,A}=0, L_{6,T}=0
\end{aligned}$$

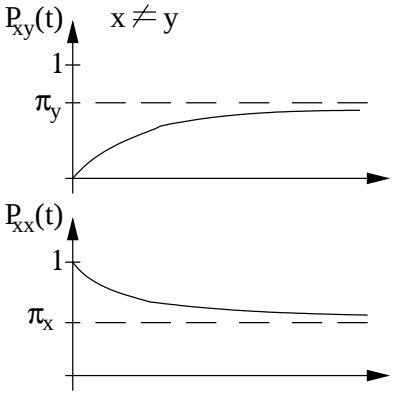
  -

$$\begin{aligned}
L_{1A} &= (\sum_{S'} P_{AS'}(d_3)\cdot L_{3,S'})(\sum_{S''} P_{AS''}(d_4)\cdot L_{4,S''}) \\
&= [P_{AA}(d_3)\cdot 1][P_{AT}(d_4)\cdot 1]
\end{aligned}$$

  - allgemein:

$$\begin{aligned}
L_{1,S_1} &= P_{S_1A}(d_3)\cdot P_{S_1T}(d_4) \\
L_{2,S_2} &= P_{S_2C}(d_5)\cdot P_{S_2G}(d_6) \\
L_{0,S_0} &= (\sum_{S_1} P_{S_0S_1}(d_1)\cdot L_{1,S_1})(\sum_{S_2} P_{S_0S_2}(d_2)\cdot L_{2,S_2})
\end{aligned}$$

- Bestimmung von P(t)



  - Eigenschaften:

    Markoveigenschaft $\rightarrow P(t+s) = P(t) \cdot P(s)$

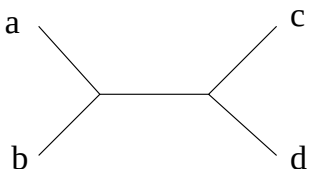
    Grenzwert: $\lim_{t \to 0} P(t) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \mathbf{I}$

    I . . . Einheitsmatrix

2.2.3 Quartet Puzzling

- Kombination der Quartets

Satz: Sei t ein Baum, Q die Menge dieser Quartet trees, die t impliziert. Dann läßt sich t eindeutig aus Q rekonstruieren!
Def.: Sei t=



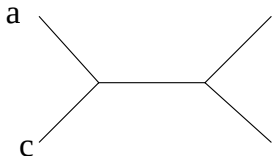
ein Quartet. Dann definiere dies N(a,b| c,d)

- Beipiel: Baum und abgeleitete Quartets

5 taxa...{a,b,c,d,e}

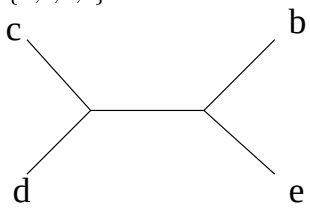
$\binom{5}{4}$ Teilmengen:

1. {a,b,c,d}:



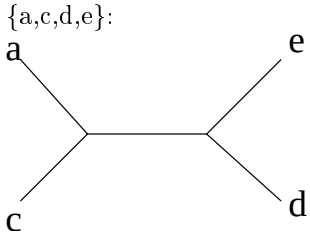
N(a,c|b,d)

2. {b,c,d,e}:



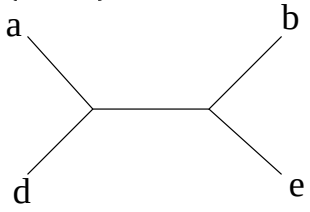
N(c,d|b,e)

3. {a,c,d,e}:



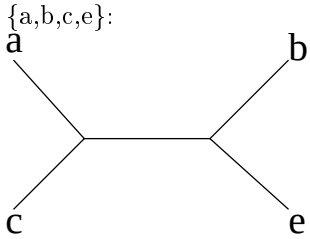
N(a,c|b,e)

4. {a,b,d,e}:



N(a,d|b,e)

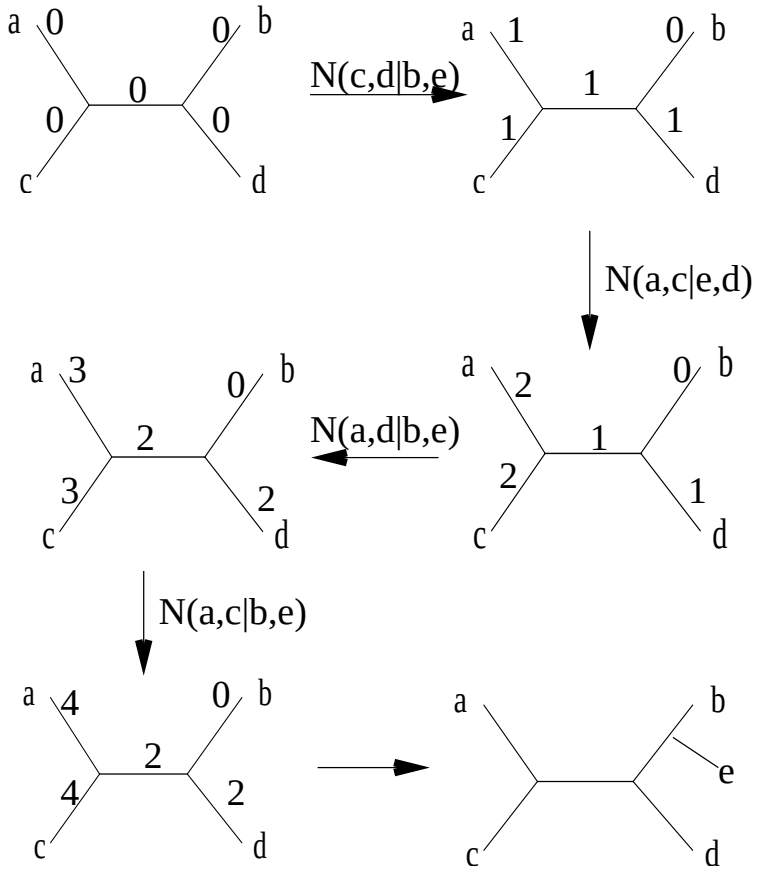
5. {a,b,c,e}:



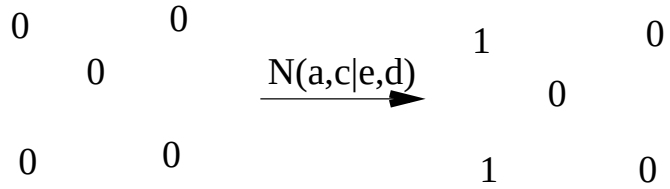
N(a,c|b,e)

- Quartet Puzzling Algo:
  - Beginne mit 1 Quartet:

    (zufällige Auswahl)
  - Füge der Reihe nach neue Taxa hinzu.
    - Setze alle Kantenmarkierung auf 0.
    - Betrachte alle Kanten an die neues taxon x hinzugefügt werden kann.
    - Betrachte alle N(...)-Relationen,die x enthalten.
    - Addiere 1 zu Kantenbeschreiftung, falls N(,) verletzt.
    - Wähle minimale Kante.
  - Beispiel:



– Beispiel 2:



– Bemerkung:

1. Satz gilt nur für Quartetts aus Bäume ⇒ Quartett Puzzling ohne Kante mit 0
2. Quartetts über Maximum Likelihood

Kapitel 3

Struktur

3.1 Monte-Carlo-Methoden

- ESS - Algorithmus (Elementary Simple Sampling):

```
w(0) := 0;
i := 0;
i := i + 1;
IF i = N THEN
   RETURN(w);
ELSE
   w(i) = Random({neighbour of w(i-1)});
   IF w(i) = w(j) THEN                    //fuer ein j < i
      GOTO1;
   ELSE
      GOTO2;
   END;
END;
```

- Probleme:

  1. uniformes Samplen von Konformationen → ESS
     neue/ initiale Konformation
     Kettenwachstumsalgorithmus → gelöst
  2. geg: Energiefunktion E(w)
     ges: native Konformation w mit E(w) minimal

  ↪ L"osung: Monte-Carlo-Methoden mit konstanter Kettenlänge und simulated Annealing

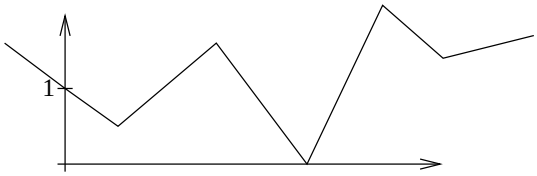
  2 Klassen:

  - lokale Bewegungen
  - Pivot-Algorithmen

- allgemeiner Algorithmus:

```
w := Konformation aus ESS (MSAW);
count := 0;
T := 50 Mio;
WHILE count < Grenze                          //z.B. 50 Mio
   w := Random_change(w);
   T := T - 1;
   count := count + 1;
END;
```

- Beispiel:
  - Suche nach geringsten Wert einer Funktion



Random_change(x):

```
initiale Werte: x := 0; f(x) := 1;
BEGIN
   y := Random[x - 0.1, x + 0.1];
   IF f(y) < f(x) THEN
      RETURN(y);
   ELSE
      RETURN(x);
END;
```

↪ hill-climbing

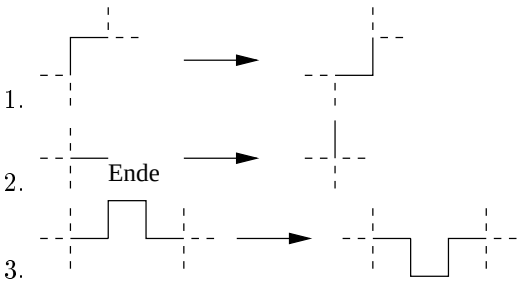
  - Problem: nur lokales Optimum
  - Verbesserung: Random_change'(x)

```
y := Random[x - 0.1, x + 0.1];
IF f(y) < f(x) THEN
   RETURN(y);
ELSE
   accept := Random_accept(T);      //T...Temperatur
   IF accept THEN
      RETURN(y);
   ELSE
      RETURN(x);
```

  - je höher T → desto größer die Wahrscheinlichkeit, dass accept = true

↪ besseres Verfahren für Konformation

  - lokale Bewegungen:



– Random_change(w):

```
w' := lokale Bewegung auf w angewandt
IF w' selfavoiding THEN
   IF E(w') < E(w) THEN
      RETURN(w);
   ELSE
      accept := Random_accept(E(w),E(w'),T);
      IF accept THEN
         RETURN(w');
      ELSE
         RETURN(w);
ELSE
   RETURN(w);
```

– Random_accept(E(w),E(w'),T):

Energie als Wahrscheinlichkeiten
↪ BOLTZMANN-Wahrscheinlichkeiten

$$Pr[w,T] = \frac{e^{-\frac{E(w)}{k_b T}}}{Z}$$

k_b ... BOLTZMANN-Konstante
Z ... Partitionsfunktion
$Z = \sum_w e^{-\frac{E(w)}{k_b T}}$ (Maximum-Likelihood unter minimalen Annahmen)

```
dE := E(w) - E(w#);
x := Random[0,1];
IF x <= e^(dE/(k_bT)) THEN
   RETURN(1);
ELSE
   RETURN(0);
```