Parameter Estimation using Likelihood functions

מנסים להסביר את ה-data set הנתון ע"י מודל הסתברותי. מאפשר לחזות התנהגות של data set דומים. מחפשים את ערך הפרמטר θ שההסתברות לראות את ה-data set בהינתן ערך זה היא המקסימאלית.

דוגמא- הטלת מטבע: $\theta = p(H)$,Heads- $1-p(H)$ .Tails-ה-data set: סדרה של הטלות מטבע, למשל: $D = HHTHTTTHH\ldots$.

הנחת העבודה: כל ניסוי (הטלת מטבע) הוא בלתי תלוי באחרים.

Maximum Likelihood Estimation (MLE): ה-likelihood של המידע בהינתן ערך ספציפי ל-θ $P(D\mid\theta)$. נרצה למצוא ערך של θ שממקסם את ההסתברות הזו (או את הלוג שלה).

בדוגמא של הטלת המטבע: זקוקים רק למידע על מספר ה-H ומספר ה-T. אלו הם סטטיסטים מספיקים.

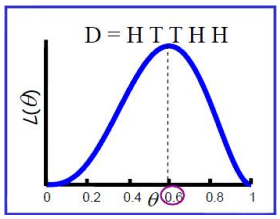
סטטיסטי מספיק: פונקציה המכילה את כל המידע הדרוש לחישוב ה-likelihood.

הערה: $s(D)$ הוא ס"מ אם לשני data-sets D, D' מתקיים: $s(D)=s(D') \Rightarrow L_D(\theta)=L_{D'}(\theta)$.

המשך דוגמת המטבע: $l_D(\theta)=\log\left(L_D(\theta)\right)=N(H)\cdot\log(\theta)+N(T)\cdot\log(1-\theta)$

למציאת המקסימום- גוזרים ומשווים לאפס:

$$\frac{N(H)}{\theta}-\frac{N(T)}{1-\theta}=0 \Rightarrow \boxed{\hat{\theta}=\frac{N(H)}{N(T)+N(H)}}$$



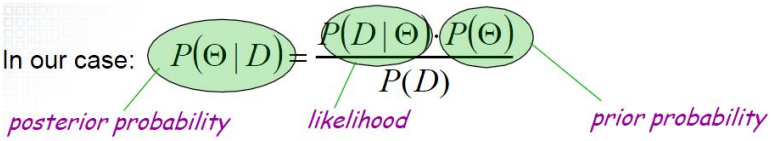
ניסויים מולטינומיים:

סדרה של K ניסויים בלתי תלויים עבורם רוצים למצוא את ערכי הפרמטרים: $\theta_1,\theta_2,\ldots,\theta_k$.

ס"מ: $N_1,N_2,\ldots,N_k$ - מספר הפעמים שנצפה כל אחד מהסוגים.

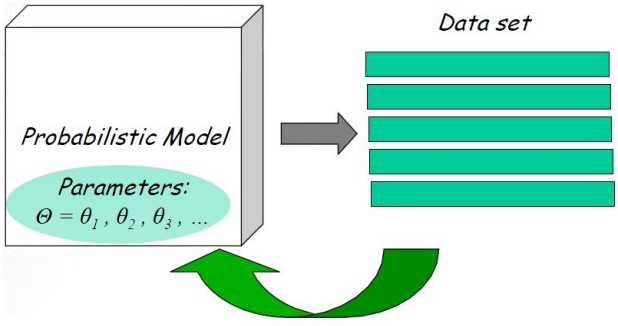
$$\boxed{L_D(\theta_1,\ldots,\theta_k)=\prod_{k=1}^{k}\theta_k^{N_k}} \qquad \boxed{MLE:\hat{\theta}_k=\frac{N_k}{\sum_l N_l}}$$

לפי נוסחת בייס:



Posterior probability: ההסתברות לכך שהמודל המסוים מתאים.

Prior probability: ידע מקדים כלשהו, על שכיחות כלשהי בעולם.



©ענת עציון

Sequence Alignment

למציאת דמיון בין רצפים יש חשיבות בשאלות ביולוגיות רבות:

- מציאת חלבונים דומים- מאפשר לחזות פונקציונאליות של חלבון הדומה לחלבונים מוכרים באורגניזם אחר (שיש לו מקור אבולוציוני משותף).
- איתור תת-רצפים דומים- מאפשר לזהות אלמנטים רגולטורים ברצף.
- מציאת קטעים חופפים- בשביל לרצף את הגנום שוברים אותו לחתיכות קטנות שאותן מרצפים ואח"כ כדי לבנות מחדש את הרצף השלם, מוצאים חפיפות בין הרצפים.

האלמנטים ב-Alignment:

Perfect matches: אותו נוקליאוטיד עומד מול אותו נוקליאוטיד.

Mismatches: נוקליאוטיד אחד יעמוד מול נוקליאוטיד אחר.

Insertion & Deletions (indel): נוקליאוטיד יעמוד מול gap.

הערה: indel בד"כ יקבל score יותר נמוך מאשר mismatches כי הם יותר נפוצים באבולוציה.

Scoring alignments:

אינטואיציה: רצפים דומים מגיעים מאב קדמון משותף. במהלך האבולוציה הרצפים השתנו ע"י מוטציות: Replacements, Deletion, Insertion. ה-Scoring צריך לבטא באיזו מידה מוטציות כאלה חלו ברצף המקורי.

פונקצית scoring פשוטה: Match: +1, Mismatch: -1, Indel: -2.

Score של רצף הוא סכום הניקוד של כל עמדה ברצף.

פונקצית scoring יותר כללית: תלוייה בקונטקסט- יש שינויים שהם יותר סבירים מאשר אחרים, למשל: שינוי לחומצה אמינית בעלת תכונות דומות לעומת כזו בעלת תכונות מנוגדות.

ה-scoring מוגדר ע"י פונקציה: $\sigma:(\sum\cup\{-\})\times(\sum\cup\{-\})\to\mathbb{R}$ כאשר:

$\sigma(x,y)$: הציון על החלפת x ב- y.

$\sigma(x,-)$: הציון על deletion של x.

$\sigma(-,x)$: הציון על insertion של x.

ה-score האופטימאלי: ה-score המקסימאלי מבין כל ה-alignments האפשריים של שני הרצפים.

$$d(S_1,S_2)=\max_{alignment\ of\ S_1\&S_2} score(alignment)$$

Edit distance: העלות של השינויים שיש לעשות כדי להפוך רצף אחד לרצף השני. המטרה היא למזער את העלות של השינויים הללו.

חסם על מספר ה-alignments האפשריים:

בהינתן שני רצפים באורכים m,n:

$$\binom{m+n}{m}<A(m,n)<\binom{m+n}{m}^2$$

חסם תחתון: מיקום m gaps ברצף אחד. חסם עליון: מיקום m gaps בשני הרצפים.

החסמים האלה אינם הדוקים, למשל בחסם העליון ישנן חזרות כאשר ישנו gap מול gap.

©ענת עציון

Global Alignment - אלגוריתם תכנות דינאמי:

Alignment של שני רצפים: $s[1\ldots m+1], t[1\ldots n+1]$.

בעמדה האחרונה של ה-alignment תימצא אחת מ-3 האפשרויות הבאות:

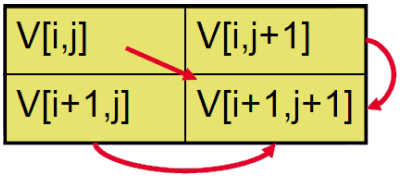
1. $x=\left(s[m+1],t[n+1]\right)$.
2. $x=\left(s[m+1],-\right)$.
3. $x=\left(-,t[n+1]\right)$.

הנוסחה הרקורסיבית: $d\left(s[1\ldots m+1],t[1\ldots n+1]\right)=d\left(s[1\ldots m],t[1\ldots n]\right)+\sigma\left(x\right)$

הגדרות:

$V[i,j]=d\left(s[1\ldots i],t[1\ldots j]\right)$ - ה-score של alignment עד המיקום ה-i ב-s והמיקום ה-j ב-t.

כך נמצא את המטריצה של התכנות הדינאמי.



$$V[i+1,j+1]=\max\begin{cases}V[i,j]+\sigma\left(s[i+1],t[j+1]\right)\\V[i,j+1]+\sigma\left(s[i+1],-\right)\\V[i+1,j]+\sigma\left(-,t[j+1]\right)\end{cases}$$

כלומר: בתא $[i+1,j+1]$ יהיה ה-score האופטימאלי, אליו ניתן להגיע ב-3 מסלולים שונים:

אלכסון: עימוד של $s[i+1]$ מול $t[j+1]$.

הליכה ימינה: עימוד של $s[i+1]$ מול gap.

הליכה למטה: עימוד של $t[j+1]$ מול gap.

אתחול המטריצה (בסיס הנוסחה הרקורסיבית):

S \ T		A	G	C
	0	**1**	**2**	**3**
0	0	-2	-4	-6
A 1	-2			
A 2	-4			
A 3	-6			
C 4	-8			

$V[0,0]=0$ - משבצת ראשונה שבה אין עימוד.

עמודה ראשונה: התאמת רצף של gaps עם רצף s - $V[i+1,0]=V[i,0]+\sigma\left(s[i+1],-\right)$.

שורה ראשונה: התאמת רצף של gaps עם רצף t - $V[0,j+1]=V[0,j]+\sigma\left(-,t[j+1]\right)$.

מציאת המסלול של ה-alignment האופטימאלי:

שומרים מצביע אל המסלול (מבין ה-3 האפשריים) שדרכו הגענו אל כל משבצת.

מהמשבצת האחרונה שבה ה-score של ה-alignment האופטימאלי הולכים אחורה לפי המצביעים למציאת ה-alignment שהביא ל-score זה. לפעמים יש יותר מ-alignment אופטימאלי אחד.

ניתוח סיבוכיות:

סיבוכיות מקום: $O\left(mn\right)$ - גודל המטריצה.

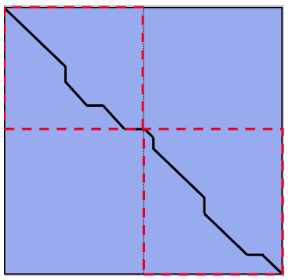
סיבוכיות זמן: $O\left(mn\right)$ - מילוי כל תאי המטריצה, $O\left(m+n\right)$ - backtracking למציאת המסלול (המסלול הכי ארוך הוא $m+n$ שהוא רצף אחד מול gaps ואז הרצף השני מול gaps).

שיפור בסיבוכיות המקום: $O\left(\min\{m,n\}\right)$ - בכל פעם שומרים רק שורה/עמודה אחת. מחשבים את הערכים שורה-שורה (או עמודה-עמודה) ומחזיקים כל פעם רק שתי שורות (או עמודות) בזיכרון בו זמנית. זה לא מאפשר לשחזר אח"כ את המסלול של ה-alignment האופטימאלי.

©ענת עציון

גרסה לאלגוריתם חסכוני במקום:

הרעיון: divide & conquer.



בהינתן שני רצפים: $s[1,m]$ ו-$t[1,n]$ ממלאים את המטריצה עמודה-עמודה ולכל עמודה שומרים את מספר השורה שבה עובר ה-alignment האופטימאלי.

לבסוף מוצאים את השורה שבה המסלול האופטימאלי עובר את עמודה $n/2$, עבור המטריצה כולה. ואז מחשבים את ה-alignment האופטימאלי מההתחלה לנקודה זו, ומנקודה זו לסוף ומשרשרים את ה-alignments.

תיאור האלגוריתם:

- משנים את האלגוריתם הקודם אשר ממלא עמודה-עמודה כך שלכל עמודה יישמר מצביע $c(i,j)$ אל השורה שבה המסלול האופטימאלי עובר.
- אם $n=1$ אז עושים alignment ל-$s[1,m]$ ו-$t[1,1]$.
- אחרת, מוצאים את המיקום $[i,n/2]$ שבו חוצה ה-alignment האופטימאלי את עמודה $n/2$.
- מוצאים את ה-alignments: $A=s[1,i]\ vs\ t[1,n/2]$ $B=s[i+1,m]\ vs\ t[n/2+1,n]$.
- מחזירים את ה-alignment: AB.

ניתוח סיבוכיות:

סיבוכיות זמן: cnm- מציאת השורה שבה המסלול חוצה את עמודה $n/2$ (מילוי מטריצה בגודל mn וביצוע c פעולות קבועות בכל צעד). וחישוב ה-alignment נתון ע"י הנוסחה הרקורסיבית הבאה:

$$\boxed{T(n,m)=cnm+T(n/2,i)+T(n/2,m-i)}$$

טענה: $T(n,m)\le 2cnm$. ובסה"כ: $O(nm)$.

סיבוכיות מקום: $O(\min\{m,n\})$.

Local Alignment – Smith-Waterman algorithm:

מציאת תתי-מחרוזות של s ו-t בעלות ה-scoring המקסימאלי בתכנות דינאמי.

$V[i,j]$ - ה-score המקסימאלי של עימוד סיומת של $s[1\ldots i]$ עם סיומת של $t[1\ldots j]$ (כלומר, ה-score המקסימאלי של local alignment שנגמר בתא זה).

השינוי מהאלגוריתם של global alignment:

הוספת אפשרות חדשה של התחלת עימוד חדש במקום הארכה של העימוד הקודם.

$$V[i+1,j+1]=\max\begin{cases}V[i,j]+\sigma(s[i+1],t[j+1])\\V[i,j+1]+\sigma(s[i+1],-)\\V[i+1,j]+\sigma(-,t[j+1])\\0\end{cases}$$

הרעיון: בכל פעם שמגיעים לערך שלילי "מאפסים" את העימוד ומתחילים אחד חדש. ולבסוף מוצאים את התא המקסימאלי בכל המטריצה.

אתחול המטריצה: שורה ראשונה ועמודה ראשונה- אפסים.

מציאת המסלול: הליכה מהתא בעל הערך המקסימאלי במטריצה לפי המצביעים עד הגעה ל-0.

©ענת עציון

Overlap Alignment:

מציאת החפיפה הכי משמעותית בין שני רצפים.

יחסי חפיפה אפשריים:

a. 

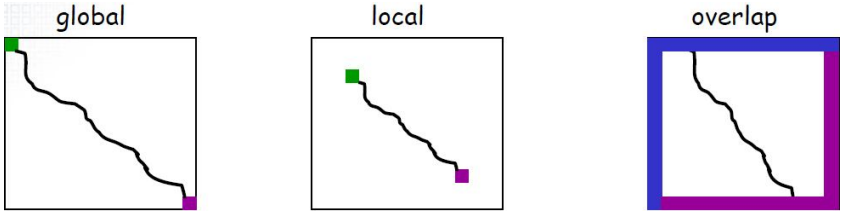
b. 

<u>הערה:</u> בשונה מ-local alignment, כאן מחפשים חפיפה הכוללת קצוות של הרצפים. נרצה שלפחות שני קצוות יימצאו בחפיפה.

<u>שינויים באלגוריתם התכנות הדינאמי:</u>

האלגוריתם יפעל כמו ב-global alignment, נרצה להתחיל מהשורה/העמודה הראשונה ולסיים בשורה/העמודה האחרונה. כלומר, נאפשר gaps בתחילת ה-alignment (אין penalty על gap בהתחלה). ולאחר מילוי המטריצה, את המקסימום נחפש רק על השורה והעמודה האחרונות.

<u>אתחול המטריצה:</u> שורה ראשונה ועמודה ראשונה- אפסים.



Affine Gap Scores:

<u>אבחנה:</u> insertions ו-deletions בדרך-כלל מופיעים בבלוקים באורך הגדול מ-1. פונקציות ה-scoring שראינו עד עכשיו נותנות penalty קבוע לכל יחידה ב-gap. וזה לא מייצג טוב את האבחנה שאלה לא n אירועים בלתי תלויים, אלא אירוע יחיד.

<u>פיתרון:</u> d - penalty על פתיחה של gap. e -penalty על הארכה של gap קיים. בד"כ: $d > e$.

ה-penalty score ל-gap באורך g: $\boxed{\gamma(g) = -d - (g-1)e}$

<u>שינויים באלגוריתם:</u>

מחשבים 3 מטריצות שונות בו-זמנית:

$M[i,j]$ - עימוד של $s[i]$ מול $t[j]$.

$I_s[i,j]$ - עימוד של $s[i]$ מול gap.

$I_T[i,j]$ - עימוד של $t[j]$ מול gap.

מתקדמים במקביל על 3 המטריצות ובוחרים בכל צעד את המקסימאלי ממטריצה כלשהי.

$$M[i,j] = \max\{M[i-1,j-1], I_S[i-1,j-1], I_T[i-1,j-1]\} + \sigma(s[i], t[j])$$
$$I_S[i,j] = \max\{M[i-1,j] - d, I_S[i-1,j] - e\}$$
$$I_T[i,j] = \max\{M[i,j-1] - d, I_T[i,j-1] - e\}$$

<u>הערות:</u>

- מניחים שאין deletion ו-insertion צמודים (כלומר, אין מעבר צמוד בין- I_S ו- I_T).

- הציון ל-mismatch יהיה גבוה יותר מציון לפתיחת gap בשני הגדילים $\sigma(mismatch) > -2e$.

<u>ייעול:</u> שימוש בשתי מטריצות בלבד, אחת ל-mismatches ואחת ל-gaps.

$$M[i,j] = \max\{M[i-1,j-1], I[i-1,j-1]\} + \sigma(s[i], t[j])$$
$$I[i,j] = \max\{M[i-1,j] - d, I[i-1,j] - e, M[i,j-1] - d, I[i,j-1] - e\}$$

©ענת עציון

Breaking Number:

קלט: שני רצפים G ו- E, $|E| < |G|$.

פלט: מספר k הקטן ביותר המקיים: שבירת G ו- E ל-k חלקים כך ש- $G = G_1 G_2 \dots G_k$, $E = E_1 E_2 \dots E_k$ ו $\forall 1 \le i \le k$ $E_i \subseteq G_i$ (כל מקטע ב- E מוכל במקטע מ- G).

דוגמא:

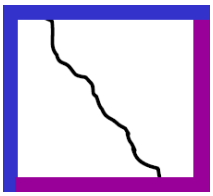
G=AAAATTTAAATTTA
E=AATTATA

G_1=AAAATTT	G_2=AAATT	G_3=A	AAAATTTAAATTTA
E_1= AATT	E_2= AT	E_3=A	--AATT---AT--A

מוטיבציה: בהינתן תעתיק mRNA, נרצה למצוא את חלקי הרצף בגנום (האקסונים).

האלגוריתם:

כמו global alignment אבל עם השינויים הבאים:



- אין Penalty על gaps בצדדים.
- ב-G (הרצף הגדול) אין gaps.
- לא מאפשרים mismatch (score = $-\infty$).
- על gap פנימי יש penalty קבוע שאינו תלוי באורכו.

$$M[i,j] = \max\{M[i-1,j-1], I[i-1,j-1]\} + \sigma(s[i], t[j])$$
$$I[i,j] = \max\{M[i-1,j]-d, I[i-1,j]-e\}$$

פונקציית ה-scoring: Match:0, Mismatch: $-\infty$, Gap intr(d):-1, Gap elong(e):0.

breaking number = -score of the alignment + 1.

הערות:

- alignment בעל score= $-(k-1)$ מתאים לחלוקה של G ו- E ל-k תת-רצפים.
- כאשר ל-alignment שמתקבל יש score= $-\infty$, אין חלוקה חוקית.
- כאשר ל-alignment שמתקבל יש score=-k, יש חלוקה חוקית ל-k+1 בלוקים ואין חלוקה חוקית למספר בלוקים קטן מזה.

ניתוח סיבוכיות:

סיבוכיות זמן: $O(|G| \cdot |E|)$ - עבור אלגוריתם תכנות דינאמי רגיל.

©ענת עציון

Alignment in Real Life

ישנם יישומים לאלגוריתמים הקודמים המשמשים למציאת alignment טוב (אבל לא בהכרח האופטימאלי) כאשר רצפי הקלט הם מאוד ארוכים (למשל- חיפוש ב-DB).

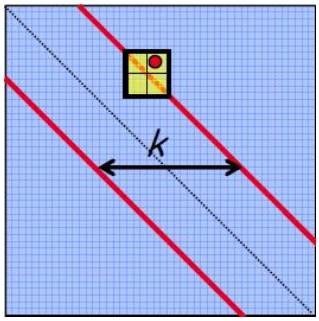
שיטות היוריסטיות:

שיטות שהן מהירות יותר אך לא בהכרח מוצאות את ה-alignment הטוב ביותר.

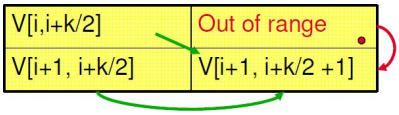
אבחנה: בהתאמות הטובות ביותר יש בדרך-כלל רצף ארוך שאין בו gaps בכלל.

השיטות ההיוריסטיות מנסות למצוא רצפים משמעותיים ללא gaps ולהרחיב אותם.

Banded DP for Global Alignment:



אם אין יותר מידי gaps ב-global alignment האופטימאלי נצפה שההתאמה הטובה ביותר תהיה קרובה לאלכסון הראשי של המטריצה, ולכן אין טעם למלא את כל המטריצה, ולכן מגדירים "שרוול" בגודל-k מסביב לאלכסון הראשי.



את ערכי המטריצה ממלאים רק עבור השרוול ורק לפי ערכים אחרים שבתוכו.

ניתוח סיבוכיות:

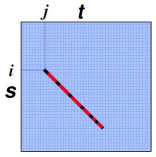
סיבוכיות זמן: $O(kn)$ - עבור מילוי ערכי שרוול בגודל-k. (יותר מהיר מאשר $O(n^2)$).

Banded DP for Local Alignment:

הבעיה: לא ידוע סביב איזה מאלכסוני המטריצה יהיה ה-alignment (וגם כאשר מחפשים global alignment ואורכי s ו-t שונים).

הפיתרון: מוצאים בשיטה היוריסטית אלכסונים פוטנציאלים, ועליהם עושים Banded DP. מניחים כי ישנו אזור מסוים שאין בו gaps. הוא מורכב מאזורי Match ו-Mismatch המצויים על אחד האלכסונים.

Seeds: אזורי Match.




```
S=****AGCGCCATGGATTGAGCGA*
T=**TGCGACATTGATCGACCTA**
```

FASTA:

קלט: הרצפים s ו-t, ופרמטר: ktup- אורך ה-Seeds שנרצה לחפש. הוא נבחר כך שקטעי Match יהיה משמעותיים אך לא נדירים מידי.

פלט: local alignment בעל ציון גבוה.

האלגוריתם:

1. מציאת כל ה-Seeds שגודלם ktup בין שני הרצפים $s[i \ldots i+ktup-1] = t[j \ldots j+ktup-1]$.

הנחה: $s \gg t$. מייצרים טבלת אינדקסים עבור המחרוזת s (שהיא למשל ה-DB שבו מחפשים) של כל ה-seeds בגודל ktup. כל seed ייוצג ע"י (i,j) - תחילת ה-seed ב-s ותחילת ה-seed ב-t. משתמשים ב-index-table: עוברים על s וממלאים בו כל רצף באורך ktup. ואז עוברים על t ולכל עמדה בודקים האם הרצף שמתחיל בה הופיע גם ב-s ומחזירים את כל זוגות האינדקסים שהם Match.

Index Table (ktup =2)
AA -
AC -
AG 5, 19
AT 11, 15
CA 10
CC 9,21
CG 7
...
TT 16

©ענת עציון

ניתוח סיבוכיות:

סיבוכיות זמן: ליניארית- $O(|s|+|t|)$ (האם מתייחסים ל-ktup כקבוע?)

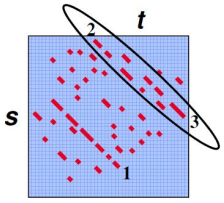
הערות:

- הגודל המקסימאלי של ה-index table הוא: $|\sum|^{ktup}$ כאשר $|\sum|$ הוא גודל הא"ב (4 או 20). כאשר ktup קטן, כל הטבלה שמורה בזיכרון. כאשרktup גדול, שומרים בזיכרון רק את הכניסות עבור רצפים שנמצאו ב-DB ואז משתמשים ב-hashing.
- ערכי ktup טיפוסיים: 1-2 עבור חלבונים, 4-6 עבור DNA. ערכים גבוהים יותר לא ייתנו מספיק אלכסונים פוטנציאליים ולכן זה מגדיל את הסיכוי לפספס local alignment אופטימאלי.

2. זיהוי אלכסונים פוטנציאליים- הרחבת ה-seeds

מוצאים קבוצות של seeds על אותו אלכסון ע"י חיסור $i-j$ של אינדקסי ההתחלה. עבור אלכסונים שבהם יש הרבה seeds, מחפשים באופן חמדני gap less match גדול ע"י הרחבת אזור ה-ktup כל עוד הציון של הרצף משתפר (ונמצא מעל score מסויים).

3. הפעלת ה-Banded DP



מריצים את ה-Banded DP על האלכסונים הטובים ביותר. ייתכן שהאלגוריתם ישלב שני אלכסונים ביחד ל-gaped match.

BLAST (Basic Local Alignment Search Tool):

כיום כבר לא משתמשים ב-FASTA אלא ב-BLAST. בשונה מ-FASTA מרשים mismatch בהגדרת ה-seeds, אבל לא gaps וישנו thresholdלציון מינימאלי ל-seed.

קלט: רצפים s ו-t ופרמטר $T = treshold$.

פלט: local alignment בעל ציון גבוה.

High scoring pairs: שתי חרוזות u ו-v באורך k המקיימות: $d(u,v) > T$.

האלגוריתם:

1. מוצאים high scoring pairs- ה-seeds.

בונים index table עבור s. עוברים על מחרוזת t ולכל עמדה-i: מוצאים מהן המחרוזות מתוך s שה-alignment שלהן עם המחרוזת בגודל k במתחילה ב-i, ה-score גדול מ-T. עושים זאת ע"י בנייה מראש של כל המחרוזות באורך K שיש להן score>T עם הרצף הנוכחי ב-t ומחפשים אותך ב-Index table של s. (K=4 לחלבונים ו-11 ל-DNA).

2. זיהוי אלכסונים פוטנציאליים- הרחבת ה-seeds

באופן דומה ל-FASTA (מציאת אלכסונים ע"י הרחבת seeds באופן חמדני, אשר מפסיקה כאשר ה-score מגיע לסף תחתון הנקבע מראש ע"י האלגוריתם).

3. הפעלת ה-Banded DP

כמו ב-FASTA.

©ענת עציון

מודל הסתברותי:

הנחה: ב-alignment אין gaps.

מודל R (Random):

מודל שבו שני הרצפים אינם קשורים. ההסתברות לקבל את האות ה-i ברצף היא ללא תלות במיקומה וללא תלות באותיות האחרות ומיוצגת ע"י הפונקציה- $q(\bullet)$.

$$P(s[1\ldots n],t[1\ldots n]\mid R)=\prod_i q(s[i])q(t[i])$$

מודל M (Match):

כל זוג של עמדות ב-alignment, מקורו מאב אבולוציוני משותף. ההסתברות המשותפת מוגדרת ע"י הפונקציה $p(a,b)$, והיא ההסתברות לכך שאות הפכה במהלך האבולוציה לאות אחרת.

$$P(s[1\ldots n],t[1\ldots n]\mid M)=\prod_i p(s[i],t[i])$$

Odds-Ratio test:

$$Q=\frac{P(s,t\mid M)}{P(s,t\mid R)}=\frac{\prod_i p(s[i],t[i])}{\prod_i q(s[i])q(t[i])}=\prod_i \frac{p(s[i],t[i])}{q(s[i])q(t[i])}$$

$Q>1$: המודל הסביר יותר הוא M. $Q<1$: המודל הסביר יותר הוא R.

Log Odds-Ratio test:

$$\log\frac{P(s,t\mid M)}{P(s,t\mid R)}=\log\prod_i \frac{p(s[i],t[i])}{q(s[i])q(t[i])}=\sum_i \log\frac{p(s[i],t[i])}{q(s[i])q(t[i])}$$

$Q>0$: המודל הסביר יותר הוא M. $Q<0$: המודל הסביר יותר הוא R.

הגדרת פונקצית ה-scoring: מוגדרת לפי המבחן $\sigma(a,b)=\log\frac{p(a,b)}{q(a)q(b)}$

הערכת פונקציות ההסתברות:

הפונקציה q: שימוש ב-maximum likelihood על הרצף הנתון $s[1\ldots n]$:

$$L(q\mid s)=\prod_{i=1}^{n} q(s[i])=\prod_a q(a)^{N_a}\Rightarrow\begin{cases}q(a)=\frac{N_a}{n} & \text{Maximum Likelihood -ML}\\ q(a)=\frac{N_a+1}{n+|\Sigma|} & \text{Maximum Aposteriori Probability -MAP}\end{cases}$$

הערה: בשיטת ה-MAP, אף אות לא תהיה בהסתברות 0, גם אם היא לא הופיעה ברצף הנתון.

הפונקציה p: מתוך data נתון של זוגות מחרוזות דומות, ההסתברות היא השכיחות של הימצאות הזוגות אותיות ביחד במחרוזות דומות $p(a,b)=\frac{N_{a,b}}{n}$

©ענת עציון

מטריצות החלפה

הגדרת הפונקציה p לחלבונים:

יש להגדיר לכל שתי חומצות אמינו (a,b) את ההסתברות לכך ש-a יימצא ב-alignment מול b בהינתן שאלה הם שני רצפים שיש ביניהם קשר אבולוציוני.

מחשבים זאת ע"פ data set נתון מראש של חלבונים הקרובים אבולוציונית. נסתכל רק על המקרים בהם זוג אותיות שונות אחת מהשנייה. אוסף זוגות לא סדורים של מוטציות- **accepted mutations.**

הגדרות:

$f_{ab}=f_{ba}$: המספר של מוטציות $\{a,b\}$ באוסף ($a\neq b$).

f : מספר המוטציות (זוגות לא סודרים) באוסף כולו.

f_{ab}/f : ההסתברות המותנית, השכיחות של $\{a,b\}$ באוסף:

$$\frac{f_{ab}}{f}=P\left(\{x,y\}=\{a,b\}\,|\,\{x,y\}\,is\,a\,mutation\right)=\boxed{P\left(\{x,y\}=\{a,b\}\,|\,x\neq y,M\right)}$$

$p(a,b)$ במודל M עבור $a\neq b$ נתונה ע"י:

$$p(a,b)=P\left(\{x,y\}=\{a,b\}\,|\,M\right)=$$

$$P\left(\{x,y\}=\{a,b\}\,|\,x\neq y,M\right)\cdot P\left(x\neq y\,|\,M\right)=\boxed{\frac{f_{ab}}{f}\cdot P\left(x\neq y\,|\,M\right)}$$

Relative mutability: פרמטר של המודל- $r=P\left(x\neq y\,|\,M\right)$, ההסת' שזוג רנדומאלי הוא מוטציה.

מטריצת PAM-1 (1-Percent Accepted Mutations):

הגדרת p: $r=1\%\rightarrow\boxed{p(a,b)=\frac{f_{ab}}{f}\cdot\frac{1}{100}=\frac{f_{ab}}{100f}}$. מכך נובע כי ישנם $100f$ זוגות ח.אמינו בכל ה-data set (או $200f$ ח.אמינו).

הגדרת q: הערכת $p(a,a)$ (ההסתברות לזוג $\{a,a\}$ במודל M) תיעשה ע"י $\boxed{q_a=n_a/n}$- השכיחות היחסית של ח.האמינו a באוסף (n_a-מספר הופעות a, n-מספר כל ח.האמינו).

הערה: q_a זו השכיחות של a בשני המודלים: M ו-R.

מספר הזוגות $\{a,a\}$:

$$f_{aa}=\frac{\#\left(a's\,in\,the\,collection\right)-\#\left(a's\,in\,mutations\right)}{2}=\frac{(200f)q_a-\sum_{b\neq a}f_{ab}}{2}=\boxed{(100f)a_a-\frac{1}{2}\sum_{b\neq a}f_{ab}}$$

Odds-Ratio: $Q(a,b)=\frac{P\left(\{a,b\}\,|\,M\right)}{P\left(\{a,b\}\,|\,R\right)}=\frac{f_{ab}/100f}{q_aq_b}=\frac{f_{ab}}{100fq_aq_b}$.

תאי מטריצה PAM1 הם ה-log Odds-Ratio: $\boxed{[PAM-1]_{ab}=\log\frac{f_{ab}}{100fq_aq_b}}$

הערה: ערכי המטריצה האמיתיים הם הערך הלוגריתמי כפול 10, מעוגל למספר שלם.

©ענת עציון

Evolutionary Distance:

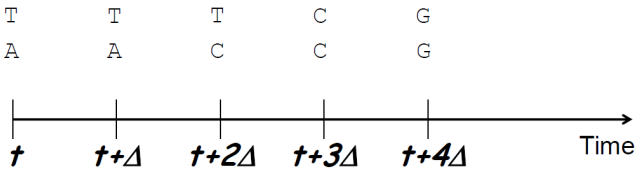
הבחירה של $r=1\%$ היא ארביטרארית. זו היא יחידה של **evolutionary change** ולא של **evolutionary time**. כיצד מגידים מטריצת החלפה עבור k יחידות של **evolutionary time**?

Model of Evolution:

הנחות המודל:

- כל עמדה משתנה ללא תלות בעמדות האחרות.
- ההסתברות למוטציות היא זהה בכל העמדות.

לאבולוציה אין "זיכרון".



שרשרת מרקוב:

שרשרת מוגדרת ע"י **הסתברות מעבר**: $P\left(X^{t+\Delta}=b \mid X^{t}=a\right)$, ההסתברות שהמצב הבא הוא b בהינתן שהמצב הנוכחי הוא a. ההסתברויות מתוארות ב**מטריצת מעברים** $P[\Delta]$.

ניתן לחשב את ההסתברות למעבר ב-n יחידות זמן: $\boxed{P[n\Delta]=P[\Delta]^n}$ (כפל מטריצות).

מטריצת PAM-250: מטריצה מאוד פופולארית, מתארת זמן אבולוציוני ארוך יחסית.

בעיות בבניית מטריצות החלפה:

- ישנן שתי דרכים שונות לחישוב $M[4\Delta]$: $M[\Delta]^4$ של PAM1 או $M'[2\Delta]^2$ של K=50.
- עד כמה רחוק האב הקדום המשותף? אם ההתפצלות נעשתה בשלב מוקדם יש מעט דמיון ברצפים ואם היא נעשתה בשלב מאוחר יש דמיון גבוה בין הרצפים.
  (כלומר, $M[250\Delta]$ ידועה ככזו שלא משקפת טוב שינויים בתקופות ארוכות של זמן).
- האם אות אחת עברה מוטציה לשנייה או ששתיהן עברו מוטציה מאות שלישית?

מטריצת BLOSUM:

מטריצות PAM לא מבדילות בין מרחקים קצרים וארוכים של time mutations.

BLOck-**SU**btitution-**M**atrices מנסות להתמודד עם הבעיה.

רעיון: שימוש ב-alignment חסרי gaps של משפחות חלבונים ולא רק של רצף בודד. הרעיון דומה ל-PAM אך כך מושגות תוצאות סטטיסטיות טובות יותר.

הערה: מטריצות פופולאריות הן: Blosum50 ו-Blosum62.

©ענת עציון

Multiple Sequence Alignment

עימוד של k רצפים. <u>קלט:</u> $S_1, S_2, \ldots, S_k$ מעל אותו הא"ב. <u>פלט:</u> k רצפים עם gaps באורך שווה: $S'_1, S'_2, \ldots, S'_k$ המקיימים: $|S'_1| = |S'_2| = \cdots = |S'_k|$.

משתמשים במטריצה לייצוג ה-alignment: כל שורה היא רצף ומניחים שאין עמודה המכילה רק gaps. ה-scoring function הרגילה נותנת ציון לכל עמודה, כאשר סדר האיברים בתוך העמודה לא משנה, הפונקציה היא סימטרית.

X^1	M	Q	_	I	L	L	L
X^2	M	L	R	-	L	L	-
X^3	M	K	_	I	L	L	L
X^4	M	P	P	V	L	I	L

מספר ה-alignments האפשרי לעמודה של k רצפים מעל א"ב בגודל L: $\boxed{\binom{L+k}{L}-1}$.

<u>הסבר קומבינטורי:</u> סידור k כדורים זהים (המיקום בעמודה) ב-$L+1$ תאים (אותיות א"ב ו-gap). ומפחיתים 1 כי עמודה של gaps היא לא חוקית.

<u>Sum-of-pairs (SP):</u> ה-score ל-MSA הוא סכום ה-score של כל הזוגות בכל $\binom{k}{2}$ הקומבינציות.

נשים-לב כי יש צורך בהגדרת הציון: $(-,-)$ כי בעמודה אחת ייתכנו מספר gaps כל עוד היא לא כולה ריקה. מגדירים אותו להיות 0 כי הכניסות הללו לא משפיעות על ה-pairwise alignment בין שני הרצפים שעבורם מחשבים את הציון.

M	Q	_	I	L	L	L
M	L	R	-	L	L	-
M	K	_	I	L	L	L
M	P	P	V	L	I	L

<u>הכללת אלגוריתם DP (Dynamic Programming) למציאת SP alignment אופטימאלי:</u>

נתונים k רצפים באורך n_i.

- במקום מטריצה דו-מימדית, מטריצה בעלת k מימדים.
- כל מימד הוא באורך: $n_i + 1$.
- לכל תא יש $2^k - 1$ תאים סמוכים שעל-פיהם הוא מחושב.

<u>ניתוח סיבוכיות:</u>

$O\left(n^k 2^k\right)$ - ישנם n^k תאים וחישוב אחד מהם לוקח $2^k - 1$ תאים סמוכים.

$O\left(k^2\right)$ - חישוב SP לכל עמודה, לחישוב ה-score של ה-alignment כולו.

בסה"כ: $O\left(k^2 2^k n^k\right)$. זו בעיה שהיא **NP-hard** (אין אלגוריתם בזמן פולינומיאלי).

<u>אלגוריתם מקורב (היוריסטי):</u>

שימוש ב-cost במקום ב-score, כלומר: מוצאים alignment בעל cost מינימאלי.

<u>הנחה:</u> ה-cost function היא למעשה פונקצית מרחק-

- $\delta(x,x) = 0$
- $\delta(x,y) = \delta(y,x) \geq 0$
- $\delta(x,y) + \delta(y,z) \geq \delta(x,z)$ (אי-שוויון המשולש)

<u>הערה:</u> מכך גם נובע שה-cost של mismatch נמוכה יותר מה-cost של שני indels.

©ענת עציון

D(S,T): ה-cost של minimum global alignment בין הרצפים S ו-T.

The 'star' algorithm:

בהינתן: Γ - סט של k רצפים $S_1, S_2, \ldots, S_k$

- לכל זוג $i < j$ מחשבים: $D(S_i, S_j)$.
- מוצאים את הרצף S' שהכי "קרוב" לשאר הרצפים, שממזער את: $\sum_{S \in \Gamma \setminus \{S'\}} D(S', S)$

  מסמנים: $S_1 = S'$ ואת שאר הרצפים מסמנים: $S_2, \ldots, S_k$.
- בצורה איטרטיביות מוסיפים את הרצפים $S_2, \ldots, S_k$ ל-alignment באופן הבא:
  - נניח שכל הרצפים הקודמים $S_1, \ldots, S_{i-1}$ כבר יש עבורם עימוד כ- $S'_1, \ldots, S'_k$.
  - עושים alignment ל-S_i עם S'_1 שמביא כפלט: S'_i ו- S''_1.
  - מתאימים את $S'_2, \ldots S'_{i-1}$ ע"י הוספת gaps במקומות שבהם הוספו gaps ל-S''_1.
  - מסמנים את S''_1 כ- S'_1 ומתחילים שוב את האיטרציה עבור הרצף ה-$i+1$.

ניתוח סיבוכיות:

סיבוכיות זמן:

$O(k^2 n^2)$ - ביצוע DP לכל זוגות הרצפים האפשריים למציאת S_1.

$O(i \cdot n^2)$ - הוספת S_i ל-alignment, ביצוע DP ל-S_i ו-S'_1 (בשלב ה-i, אורכו של S'_1 יכול להיות לכל היותר $i \cdot n$). ולהוספת כל הרצפים ל-alignment נקבל: $\sum_{i=1}^{k-1} O(i \cdot n^2) = O(k^2 n^2)$.

בסה"כ: $O(k^2 n^2)$.

מידת הקירוב:

M^* - ה alignment האופטימאלי, M - ה alignment שנוצר ע"י האלגוריתם.

$d(i,j)$ - המרחק המושרה על הזוג S_i, S_j ע"י M. $\boxed{v(M) = \sum_{i=1}^{k} \sum_{i \neq j=1}^{k} d(i,j) = 2 \sum_{i<j} d(i,j)}$

לכל i: מבצעים alignment אופטימאלי עם S_1 ולכן: $d(1,i) = D(S_1, S_i)$.

אי-שוויון המשולש $v(M) = \sum_{i=1}^{k} \sum_{i \neq j=1}^{k} d(i,j) \leq \sum_{i=1}^{k} \sum_{i \neq j=1}^{k} \left(d(i,1) + d(1,j) \right)$

$= 2(k-1) \sum_{l=2}^{k} d(1,l) = \boxed{2(k-1) \sum_{l=2}^{k} D(S_1, S_l)}$

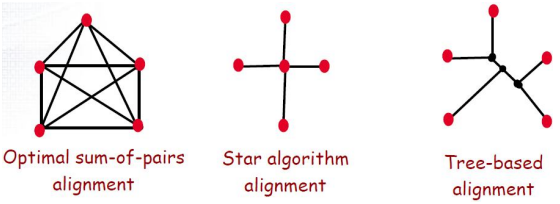
לפי הגדרת S_1: $v(M^*) = \sum_{i=1}^{k} \sum_{i \neq j=1}^{k} d^*(i,j) \geq \sum_{i=1}^{k} \sum_{i \neq j=1}^{k} D(S_i, S_j) \geq \sum_{i=1}^{k} \sum_{j=2}^{k} D(S_1, S_j)$

$= \boxed{k \sum_{j=2}^{k} D(S_1, S_j)}$

Star algorithm לכל היותר גרוע פי-2 מהאופטימאלי: $\boxed{\frac{v(M)}{v(M^*)} \leq \frac{2(k-1)}{k} \leq 2}$

©ענת עציון

Tree Alignment:

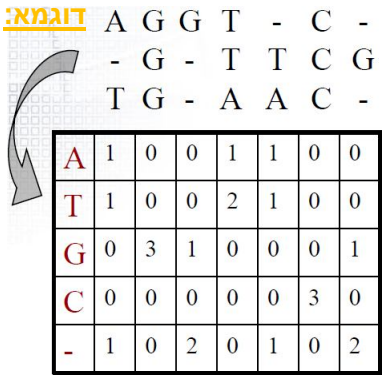


בעיה: MSA קונבנציונאליים לא ממדלים בצורה נכונה יחסים אבולוציוניים.

קלט: X - סט של רצפים. T-עץ פילוגנטי שהעלים שלו הם רצפי X.

פלט: ערכים על הצמתים הפנימיים של T כך שסכום ה-costs של כל הקשתות הוא מינימאלי.
הערכים על הצמתים הללו הם: רצפים או פרופילים.

דוגמא:

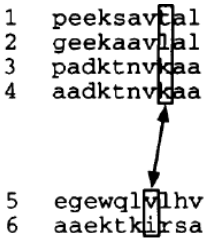


A	1	0	0	1	1	0	0
T	1	0	0	2	1	0	0
G	0	3	1	0	0	0	1
C	0	0	0	0	0	3	0
-	1	0	2	0	1	0	2

פרופיל: פרופיל של MSA באורך n מעל א"ב $\sum$ הוא טבלה בגודל: $(|\sum|+1)\times n$. כל עמודה מייצגת שכיחות האותיות ב-MSA וכל שורה מייצגת אות אחרת.

עימוד של רצף לפרופיל: ציון של עימוד אות ברצף מול עמודה בפרופיל הוא ממוצע ממושקל של הציונים של הנוקליאוטיד עם כל אחת מהאותיות בעמודה. עימוד כזה ניתן לעשות ע"י אלגוריתם DP סטנדרטי עם פונקציית ציון מתאימה.

דוגמא ל-alignment של שתי עמודות:



ניתן לעמד שני פרופילים זה מול זה ע"י נתינת ציון ל-alignment של שתי עמודות. הציון של עימוד שתי העמודות המסומנות הוא:

$$score = \begin{pmatrix} M(t,v)+M(t,i)+M(l,v)+ \\ M(l,i)+2M(k,v)+2M(k,i) \end{pmatrix} / 8$$

הערה: מציאת ההשמה האופטימאלית של רצפים לצמתים הפנימיים זו בעיית NP-hard.

אלגוריתם Clustal:

- מחזיקים את הפרופילים של ה-alignment לכל העלים.
- בוחרים שני עלים שכנים (צמתים בעלי אב משותף ב- T) הכי קרובים ועבורם:
  - עושים alignment לפרופילים שלהם לקבלת "פרופיל האב".
  - שמים אותו בצומת של האב המשותף.
- ממשיכים כך למעלה בעץ עד שמקבלים פרופיל המייצג את כל העץ.

ClustalW: כל רצף/פרופיל גם ממושקל בחישוב הציון לפי מידת האמינות שלו.

בניית העץ הפילוגנטי: ע"י pairwise alignment בין הרצפים בסט (נדון על כך יותר מאוחר בקורס).

Lifted tree alignment:

ייצוג העץ ע"י רצף מבין הרצפים הנתונים במקום ע"י פרופיל. כל אחד מהצמתים הפנימיים בעץ מיוצג ע"י רצף של אחד הבנים שלו. כדי לבחור את הרצף מבין הבנים משתמשים בתכנות דינאמי.

קלט: X - סט של k רצפים שאורכת לכל היותר n. T-עץ פילוגנטי על X.

פלט: lifted labels על הצמתים הפנימיים כך שסכום ה-costs של כל הקשתות הוא מינימאלי.

©ענת עציון

עיקרון מנחה: מחשבים לכל צומת v ב-T ורצף $S \in X$ את: $d(v,S)$ - הcost האופטימאלי של תת-העץ של v כאשר הוא מסומן ברצף S.

ה-cost של העץ האופטימאלי: $\min\left\{ \underset{S\in X}{d}\left(root,S\right)\right\}$

המטריצה: ציר-x: הצמתים בעץ, ציר-y: הרצפים ב-X. כל תא מכיל: $d(v,S)$.

אתחול המטריצה: לכל עלה v אשר מסומן ב-S_v (העץ הנתון מכיל את ערכי העלים):

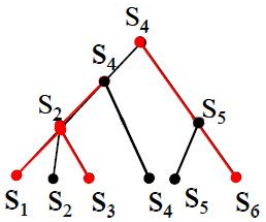
$$d(v,S)=\begin{cases}0 & S=S_v\\ \infty & S\neq S_v\end{cases}$$

הנוסחה הרקורסיבית: לצומת פנימי v שבניו הם: $u_1,\ldots,u_l$ -

$$d(v,S)=\sum_{i=1}^{l}\min_{S'\in X}\left\{D(S,S')+d(u_i,S')\right\}$$

הערה: D - פונקציה למרחק בין שני רצפים.

דוגמא:



ה-cost של העץ:

$$D(S_1,S_2)+D(S_3,S_2)+D(S_2,S_4)$$
$$+D(S_6,S_5)+D(S_5,S_4)$$

ניתוח סיבוכיות:

סיבוכיות זמן:

$O(k^2n^2)$ - חישוב מרחקים בין כל זוגות הרצפים ע"י k^2 pairwise alignment.

$O(k^3)$ - התכנות הדינאמי: $O(k^2)$ לכל צומת וישנן k-1 איטרציות (צמתים פנימיים).

בסה"כ: $O\left(k^2\left(n^2+k\right)\right)$.

טענה: פלט האלגוריתם הינו לכל היותר פי-2 מהפתרון האופטימאלי.

הוכחה:

T^* - עץ עם תוויות אופטימאליות. S_v^* - התווית של הצומת v ב-T^*.

T^L - העץ שהאלגוריתם Lifted Tree בונה. S_v^L - התווית של הצומת v ב-T^L.

לחלק מהקשתות ב-T^L יש cost=0 (בגלל שיש אותו הרצף משני צידי הקשת). נתבונן בקשתות (v,u) שה-cost שלהן גדול מ-0: $P(v,u)$ - המסלול ב-T^* מ-v לעלה אשר מסומן ב-S_u.

$$\boxed{D(S_v,S_u)\overset{(1)}{\leq}D(S_v,S_v^*)+D(S_u,S_v^*)\overset{(2)}{\leq}2D(S_u,S_v^*)\overset{(3)}{\leq}2D(P(v,u))}$$

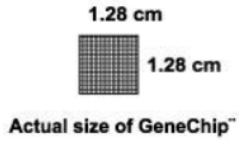
$(1),(3)$: אי-שוויון המשולש. (2): אופן בחירת S_v בתור הרצף מבין כל הסט הקרוב ביותר ל-S_V^*.

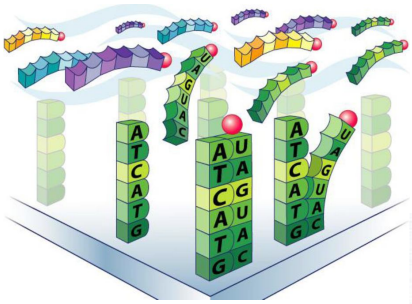
$$\boxed{D(T^L)=\sum D(S_v,S_u)\leq 2\sum D(P(v,u))\leq 2D(T^*)}$$

©ענת עציון

Gene Expression

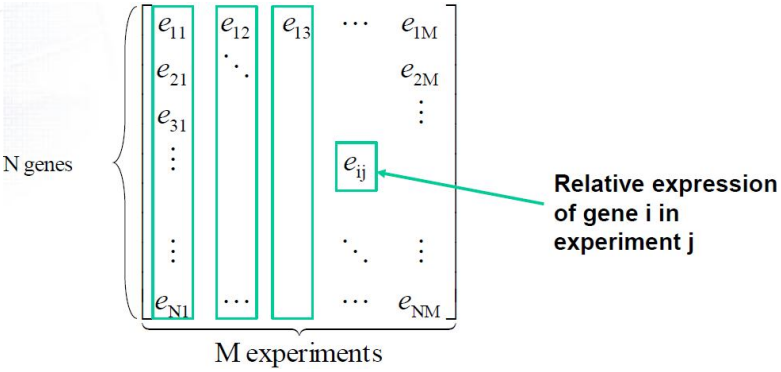
Microarray Technology:





צ'יפ שיש עליו גדילי DNA ברצף ידוע ובמיקום ידוע. כל נקודה בצ'יפ- גן אחד. ניתן לבדוק את רמת הביטוי ברקמה מסויימת ע"י לקיחת RNA מהרקמה, הפיכתו ל-cDNA וע"י קומפלמנטציה ל-DNA שעל הצ'יפ ניתן לראות את רמת הביטוי של הגן (מסמנים באופן שונה את הדוגמה הנבדקת ואת דוגמת הבקרה שכלפיה משווים ומנתחים את הקרינה הנפלטת מהרצפים באופן ממוחשב).

פרופיל expression: מטריצה שבה התא ה-$[i,j]$- רמת ביטוי של גן i בניסוי j.



Classified Gene Expression Data:

- ידע מוקדם באופן בינארי לגבי מקור ה-RNA: חולה לעומת בריא, או שלבים שונים במחלה וכו'.
- זיהוי באופן סטטיסטי של הגנים אשר מפרידים בין שתי הקבוצות- **Informative genes**.

צריך לבנות מודל סטטיסטי שמתאר את העולם ולמצוא את הגנים ש"מפתיעים" את המודל.

p-value: ההסתברות למציאת גן עם ציון נתון במודל רנדומאלי. נרצה שיהיה כמה שיותר נמוך.

למשל: תחת "הנחת-האפס" מה הסיכוי לקבל הפרש כזה (או מפתיע יותר ממנו) בין ממוצע הביטוי באנשים החולים לממוצע הביטוי באנשים בריאים.

t-test: מבחן t לשני מדגמים בגודל שונה ובעלי שונות לא זהה:

$$S_{X_1X_2} = \sqrt{\frac{(n_1-1)S_{X_1}^2 + (n_2-1)S_{X_2}^2}{n_1+n_2-2}}$$

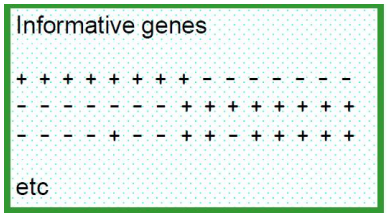
$$t = \frac{\bar{X}_1 - \bar{X}_2}{S_{X_1X_2}\sqrt{\frac{1}{n_1}+\frac{1}{n_2}}}$$

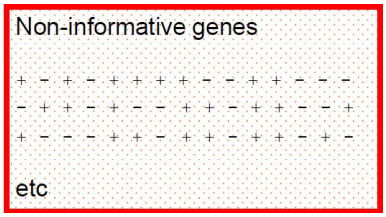
יתרונות: קל לחישוב, קל ליישום, סיבוכיות ליניארית.

חיסרון: ההנחה ששני המדגמים מהתפלגות נורמאלית לא ממש נכונה.

The non-parametric approach: הערך עצמו (ההפרש בין הממוצעים של ה-expression) לא מעניין, מה שמעניין זה האם הגן מפריד בין שתי הקבוצות (למשל- בין חולים לבריאים).

דוגמא: + בריאים, - חולים. מסדרים אותם ממויינים לפי רמת ביטוי.





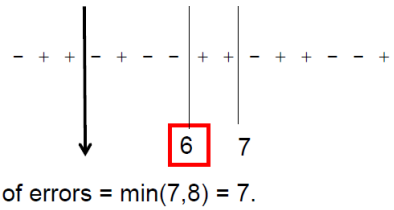
©ענת עציון

TNoM- Threshold Number of Misclassification Score:

מחפשים את המיקום בסדרה שמשאיר כמה שיוצר (+) וכמה שיותר (-) בשני צדדים מנוגדים.

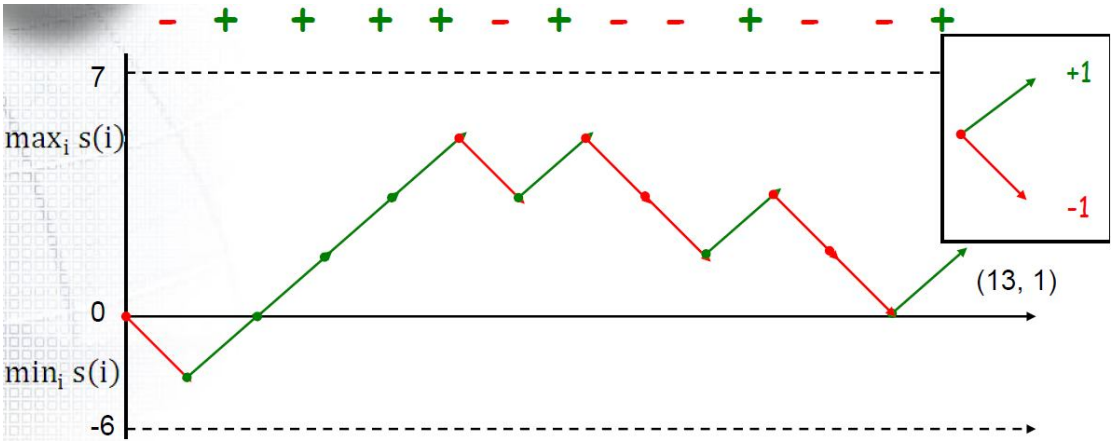
מחשבים את מספר ה-errors ומוצאים את המיקום האופטימאלי בו מספר טעויות מינימאלי.

דוגמא:



הערה: gene classifier אופטימאלי הוא כזה שיש לו ציון טעויות של: 0.

חישוב TNoM באופן יעיל:



הנוסחה לחישוב #errors: $s(i)=\sum_{j=1}^{i} v(j)$ - לכל עמדה i בסדרה סכום העמדות שעד אליה.

$TNoM(v)=\min\{7-\max_i s(i), 6+\min_i s(i)\}=4$, כאשר 7-# ה-(+), 6-# ה-(-).

ניתוח סיבוכיות:

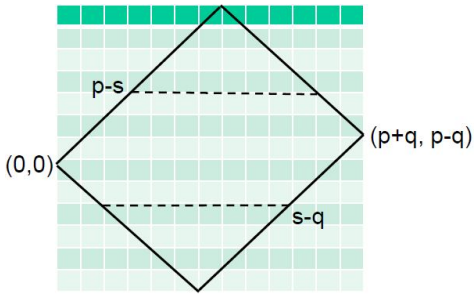
$O(n\log n)$ - מיון הווקטור של פלוסים ומינוסים לפי רמות ביטוי.

$O(n)$ - חישוב TNoM.

TNoM p-value: כמה מפתיע למצוא TNoM כזה באופן רנדומאלי?

בהינתן וקטור v עם p '+' ו '-' אשר מושמים באופן רנדומאלי, מהי ההסתברות ש-$TNoM(v)\le t$?

אבחנה: לווקטור v יש TNoM $\le s$ אם המסלול בחישובו חוצה את אחד מהצירים $Y_1=p-s$ או $Y_2=s-q$.



מחשבים כמה ווקטורים באמת עוברים את הצירים האלה מתוך $\left|\{+,-\}^{p,q}\right|=\binom{p+q}{p}$ הווקטורים האפשריים.

$v(i,j)$ - מספר המסלולים מ-$(0,0)$ ל-(i,j) שלא עוברים דרך Y_1 או Y_2.

©ענת עציון

$$v(0,0)=1$$

נוסחת התכנות הדינמי:

$$v(i,j)=\begin{cases}0 & not\ in\ space \\ 0 & if\ \ j\geq Y_1\ \ or\ \ j\leq Y_2 \\ v(i-1,j-1)+v(i-1,j+1) & otherwise\end{cases}$$

ערך ה-TNoM p-value:

$$TNoM\ \ p-value(s)=1-\frac{v(p+q,p-q)}{\binom{p+q}{p}}$$

שזוהי ההתסברות לכך שווקטור אקראי אינו עובר את הצירים הללו.

סיבוכיות: $O(n^2)$.

The reflection principal:

דרך נוספת לחישוב מספר המסלולים שלא חוצים אחד מן הצירים Y_1, Y_2 ע"י נוסחה קומבינטורית

סגורה: מספר המסלולים מ- $(0,0)$ ל- (n,k) הוא: $\binom{n}{(n+k)/2}$.

המסלולים מ- $(0,0)$ ל- (n,k) שנוגעים ב- $Y=A$ שווה ל-# המסלולים מ $(0,2A)$ ל- (n,k).

מכיוון שמעוניינים במספר המסלולים שחוצים אחד משני הצירים משתמשים בנוסחת "הכלה והפרדה" למניעת כפילויות.

סיבוכיות: $O(\log(p+q))$.

©ענת עציון

שרשראות מרקוב

בשיעורים הקודמים הנחנו שכל אות ברצף גנומי יוצרה באופן רנדומאלי מהתפלגות כלשהי מעל הא"ב $\{A,C,T,G\}$. אבל זה לא לגמרי מדוייק, למשל יש רצפים מיוחדיפ בגנום כמו TATA באזור ה-upstream של גנים, ודוגמא נוספת- רצפי CG פחות נפוצים בגנום מכפי שמצופה באופן רנדומאלי.

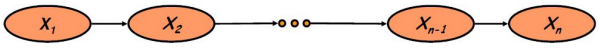
Finite Markov Chain:

תהליך סטוכסטי הסתברותי, בעל יחידות זמן שלמות שבכל אחת מהן עוברים ממצב אחד למצב אחר מתוך $m>1$ מצבים- $\{s_1,\ldots,s_m\}$ שהם הדומיין D. ובנוסף:

1. ווקטור בגודל m ובו ההסתברויות להתחיל מכל מצב $\left(p(s_1),\ldots,p(s_m)\right)$
2. מטריצה בגודל $m\times m$ של הסתברויות מעבר בין המצבים- $M=\left(m_{s_is_j}\right)$.

דוגמא: $D=\{A,C,T,G\}$, $p(A)$-ההסתברות שהרצף יתחיל באות A ו-m_{AG} - ההסתברות ש-G יהיה אחרי A ברצף.

ההסתברות לשרשרת מעברים באורך n:



$$p\left((x_1,x_2,\ldots,x_n)\right)=p(X_1=x_1)\prod_{i=2}^{n}p(X_i=x_i\mid X_{i-1}=x_{i-1})=\boxed{p(x_1)\prod_{i=2}^{n}m_{x_{i-1}x_i}}$$

ייצוג המטריצה:

מטריצה סטוכסטית: מטריצה שהסכום בכל שורה בה הוא: 1 $\sum_t m_{st}=1$.

	A	B	C	D
A	0.95	0	0.05	0
B	0.2	0.5	0	0.3
C	0	0.2	0	0.8
D	0	0	1	0

מטריצת הסתברויות המעבר $M=\left(m_{st}\right)$.

ווקטור ההתפלגות ההתחלתי $\left(u_1,\ldots,u_m\right)$ מגדיר את ההתפלגות של X_1, $p(X_1=s_i)=u_i$.

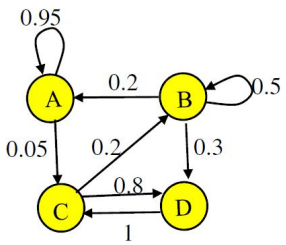
לאחר מעבר אחד, ההתפלגות הינה: $X_2=X_1M$.

דוגמאות:

- אם $X_1=(0,1,0,0)$ אז $X_2=(0.2,0.5,0,0.3)$ (רק השורה השנייה במטריצה)
- אם $X_1=(0,0,0.5,0.5)$ אז $X_2=(0,0.1,0.5,0.4)$ (ממוצע שורות 3 ו-4 במטריצה).

ההתפלגות ה-i: $\boxed{X_i=X_1M^{i-1}}$

ייצוג שרשרת מרקוב כגרף:

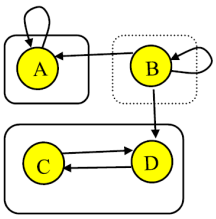


הגרף מימין לפי המטריצה בדוגמא הקודמת.

כל קשת מכוונת $A\to B$ היא הסתברות מעבר **חיובית** מ-A ל-B.

©ענת עציון

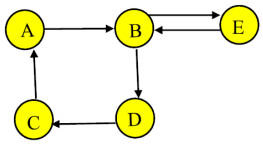
<u>מצבים ותכונות של שרשרת מרקוב מסווגים לפי תכונות הגרף:</u>



<u>Recurrent state</u>: צומת ברכיב קשיר היטב בגרף אשר מהווה "בור"- כשנכנסים אליו לא ניתן לצאת ממנו. צמתים: A, C, D.

<u>הגדרה נוספת</u>: צמות s הוא recurrent אם ניתן להגיע אליו מכל צומת שהוא ישיג מ-s.

<u>Transient state</u>: צומת שאינו מהווה "בור"- ניתן לחזור ממנו לכל צומת שממנו מגיעים. צמתים: B.



<u>Irreducible graph</u>: גרף קשיר היטב שכל צמתיו הם recurrent (כי הם שייכים לאותו רק"ה שהוא הגרף כולו).

<u>Period</u>: ה-GCD של כל המעגלים שצומת נמצא בהם. ה-period של A =2.

הערה: למשל אם יש לולאה עצמית אז ה-period=1.

<u>טענה:</u> בגרף קשיר היטב ה- period של כל הצמתים הוא זהה.

<u>Periodic Graph/Chain</u>: גרף שלכל הצמתים יש period>1. אחרת זה aperiodic.

<u>Ergodic Markov chains</u>: שרשרת מרקוב היא ארגודית אם:

1. הגרף התואם הוא קשיר היטב.
2. הגרף אינו periodic (כלומר, ה-period של כל הצמתים הוא-1).

<u>הערה:</u> זה מבטיח שהתהליך המרקובי מתכנס להתפלגות יחידה שבה לכל מצב הסתברות גדולה מ-0, שלא משתנה יותר.

<u>Stationary Distribution</u>: תהא M שרשרת מרקוב בעלת m מצבים ו-$V=(v_1,\ldots,v_m)$ ווקטור של התפלגות ההסתברויות מעל m המצבים. V נקרא **התפלגות סטציונרית** עבור M אם $VM=V$ (כלומר, צעד נוסף בתהליך לא משנה את ההתפלגות).

V ווקטור הוא התפלגות סטציונרית ↔ V הוא ווקטור עצמי שמאלי של M עם ערך עצמי-1.

<u>משפט:</u> לכל מטריצה סטוכסטית יש ו"ע (שמאלי וימני) של 1-ים עם ע"ע-1.

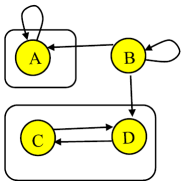
<u>הערה:</u> לכל שרשרת מרקוב יש התפלגות סטציונרית.

<u>שרשרת מרקוב "טובה":</u>

שרשרת מרקוב היא טובה כאשר ההתפלגות של X_i כאשר $i\to\infty$:

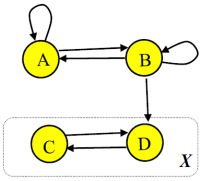
1. מתכנסת להתפלגות יחידה שאינה תלויה בהסתברויות ההתחלה.
2. בהתפלגות זו, לכל מצב יש הסתברות חיובית ממש.

<u>המשפט היסודי של Finite Markov Chains</u>: שרשרת מרקוב היא טובה אם"ם הגרף שלה ארגודי.



<u>דוגמא-1:</u> אם $p(X_1=A)=1$ אז נישאר ב-A תמיד. אם $p(X_1=C)=1$ אז נישאר ב-$\{C,D\}$ תמיד.

<u>מסקנה-1:</u> אם בגרף יש שני צמתים שאינם ישיגים אחד מהשני, אז $\{X_i\}$ לא יכול להתכנס להתפלגות שהיא ב"ת תלוייה במצב ההתחלתי.

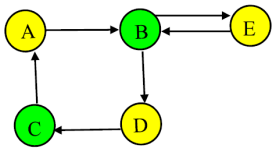


<u>דוגמא-2:</u> ברגע שעוברים פעם ראשונה מ- B ל-D, לא ניתן לחזור יותר אחורה.

<u>מסקנה-2:</u> לכל התפלגות התחלתית, מצב transient-י יופיע במסלול מספר סופי של פעמים.

⇐ **שרשרת מרקוב טובה היא irreducible.**

©ענת עציון



דוגמא-3: בשרשרת זו ה-period הוא-2. בהינתן התפלגות התחלתית $p(B)=1$, נגיע למצבים $\{B,C\}$ בצעדים אי-זוגיים בלבד ולמצבים $\{A,D,E\}$ בצעדים זוגיים בלבד.

מסקנה-3: בשרשרת מרקוב פריודית ישנן התפלגויות התחלה שבעקבותיהן המצבים יופיעו רק במספר צעדים מחזורי. עבור התפלגויות התחלה אלה, X_i לא מתכנס כאשר $i \to \infty$.

$\Leftarrow$ שרשרת מרקוב טובה היא לא פריודית.

Modeling CpG Islands:

בגנום האנושי הזוג CG הופך פעמים רבות ל-(CH_3-C)G שהופך פעמים רבות ל-TG, ולכן CG מופיע בגנום פחות מהמצופה. תהליך המתילציה לפעמים מדוכא בקטעים קצרים בגנום, למשל באזורי תחילת הגן של גנים רבים. האזורים הללו נקראים: **CpG islands** (C-phosphate-G).

בעיה-1: בהינתן רצף קצר מהגנום, האם הוא הגיע מאזור CpG island?

פותרים את הבעיה ע"י שני מודלים של שרשראות מרקוב מעל המצבים $\{A,C,T,G\}$ בעלי הסתברויות מעבר שונות:

מודל (+): מטריצת מעברים $M^+ = \left(m_{st}^+\right)$.

מודל (-): מטריצת מעברים $M^- = \left(m_{st}^-\right)$.

ויש לקבוע מאיזה מהמודלים יותר סביר שהרצף הגיע.

את מטריצות המעבר והסתברויות ההתחלה לומדים מתוך מידע קיים ע"פ maximum likelihood estimation ע"י חישוב השכיחות של הופעת אות אחת אחרי אות אחרת.

$$p_+\left(X_i = a \mid X_{i-1} = b\right) = \frac{N_{ba,+}}{\sum_x N_{bx,+}}$$

$$p_+\left(X_1 = a\right) = \frac{N_{a,+}}{\sum_x N_{x,+}}$$

$N_{x,+}$ - מספר הפעמים שהאות x הופיעה ב-CpG data set.

$N_{bx,+}$ - מספר הפעמים שהאות x הופיעה אחרי האות b ב-data set.

מודל (+):

X_{i-1} \ X_i	A	C	G	T
A	0.18	0.27	0.43	0.12
C	0.17	$p_+(C \mid C)$	0.274	$p_+(T \mid C)$
G	0.16	$p_+(C \mid G)$	$p_+(G \mid G)$	$p_+(T \mid G)$
T	0.08	$p_+(C \mid T)$	$p_+(G \mid T)$	$p_+(T \mid T)$

מודל (-):

X_{i-1} \ X_i	A	C	G	T
A	0.3	0.2	0.29	0.21
C	0.32	$p_-(C \mid C)$	0.078	$p_-(T \mid C)$
G	0.25	$p_-(C \mid G)$	$p_-(G \mid G)$	$p_-(T \mid G)$
T	0.18	$p_-(C \mid T)$	$p_-(G \mid T)$	$p_-(T \mid T)$

קלט: רצף $x = \left(x_1, \ldots, x_L\right)$ מחשבים את ה-ratio:

$$RATIO = \frac{p\left(x \mid (+)model\right)}{p\left(x \mid (-)model\right)} = \frac{\prod_{i=1}^{L-1} p_+\left(x_{i+1} \mid x_i\right)}{\prod_{i=1}^{L-1} p_-\left(x_{i+1} \mid x_i\right)}$$

הערה: $p_+\left(x_1 \mid x_0\right)$ מוגדר לשם הנוחות כ: $p_+\left(x_1\right)$ וכך גם לגבי ה (-).

ליתר דיוק מחשבים את ה-log ratio:

$$\log Q = \log \frac{p\left(x_1 \ldots x_L \mid +\right)}{p\left(x_1 \ldots x_L \mid -\right)} = \sum_i \frac{p_+\left(x_i \mid x_{i-1}\right)}{p_-\left(x_i \mid x_{i-1}\right)}$$

Log Q > 0: מודל ה (+) יותר סביר.

Log Q < 0: מודל ה (-) יותר סביר.

©ענת עציון

בעיה-2: מהו מיקום מקטע CpG ברצף נתון?

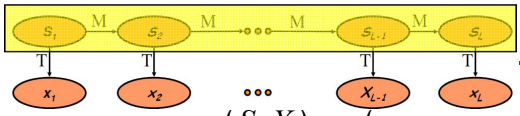
$A^+ \; C^+ \; G^+ \; T^+$
$A^- \; C^- \; G^- \; T^-$

מגדירים מודל ובו ישנם 8 מצבים שכולם רכיב קשיר היטב.
הבעיה היא שלא ידועה סדרת המצבים אלא רק רצף האותיות.
ולכן משתמשים ב-Hidden Markov Model.

HMM:

שרשרת מרקוב מעל סט של מצבים, כך שלכל מצב s קיימת הסתברות מותנית לפליטת האות x $p(X_i = x \mid S_i = s)$. זה נקרא emission probability.

סימון: $e_s(b)$ - ההסתברות לפליטת האות b כשנמצאים במצב s.



ההסתברות לשרשרת מצבים S הפולטת סדרת אותיות X:

$$p(S,X) = p(s_1,\ldots,s_L,x_1,\ldots,x_L) = \prod_{i=1}^{L} p(s_i \mid s_{i-1})\, p(x_i \mid s_i) = \boxed{\prod_{i=1}^{L} m_{s_{i-1},s_i} e_{s_i}(x_i)}$$

הערה: כאמור $m_{s_0,s_1} = p(S_1 = s_1)$.

טענה: הפונקציה $p(S,X)$ היא פונקצית הסתברות תקינה, כלומר:

$$\sum_{(s_1,\ldots,s_L,x_1,\ldots,x_L)} p(s_1,\ldots,s_L,x_1,\ldots,x_L) = \sum_{(s_1,\ldots,s_L,x_1,\ldots,x_L)} \left(\prod_{i=1}^{L} m_{s_{i-1},s_i} e_{s_i}(x_i) \right) = 1$$

תכונות חוסר-תלות ב-HMM:

1. ההתפלגות של המצב ה-S_k תלויי ה רק במצב של השלב הקודם- s_{k-1}.
2. ההתפלגות של האות X_k תלוייה רק במצב הנוכחי- s_k.

בעיה-א: מהי סדרת המצבים הכי סבירה עבור סדרת האותיות? (שקול לבעיה-2)

קלט: סדרת פלט $x = (x_1,\ldots,x_L)$.

פלט: סדרת המצבים הסבירה ביותר $s^* = (s_1^*,\ldots,s_L^*)$ היא כזו שממקסמת את $p(s \mid x)$.

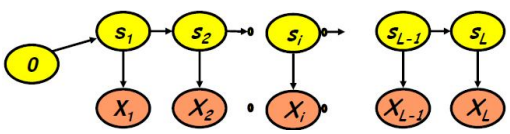
ולפי נוסחת בייס: $p(s \mid x) = \dfrac{p(s,x)}{p(x)}$ ומכיוון ש-x נתון אז נרצה למקסם את $p(s,x)$.

אלגוריתם Viterbi:

מעבר על כל סדרות המצבים האפשריות הוא בסיבוכיות אקספוננציאלית ולכן משתמשים ב-DP.

הרעיון: לכל מיקום $1,\ldots,L$ ולכל מצב l (מתוך m מצבים) מחשבים:

$v_l(i)$ - ההסתברות $p(s_1,\ldots,s_i,x_1,\ldots,x_i \mid s_i = l)$ של המסלול הסביר ביותר באורך i שמסתיים במצב l. הנוסחה הרקורסיבית: (כאשר מוסיפים מצב התחלתי מיוחד-0)



$$\boxed{\begin{aligned} v_l(i) &= e_l(x_i) \cdot \max_k \{ v_k(i-1) \cdot m_{kl} \} \\ v_0(0) &= 1,\ v_k(0) = 0 \ \ for\ k > 0 \end{aligned}}$$

ובנוסף: $ptr_i(l) = argmax_k \{ v_k(i-1) m_{kl} \}$ ששומר את המצב הקודם בשביל שחזור המסלול.

©ענת עציון

שלב ה-Trackback (מציאת סדרת המצבים):

$$Base: S_N^* = argmax_{S'}\left\{v_N\left(S'\right)\right\}$$
$$step: S_i^* = ptr_{i+1}\left(s_{i+1}^*\right)$$

פלט: $\boxed{p\left(s_1^*,\ldots,s_L^*,x_1,\ldots,x_l\right)=\max_k\left\{v_k\left(L\right)\right\}}$.

סיבוכיות: $O\left(L\times S^2\right)$.

בעיה-ב: מהי ההסתברות לקבל את סדרת התווים?

ההסתברות לכך שסדרת התווים התקבלה ממודל ה-HMM: $p\left(x\right)=\sum_S p\left(X,S\right)$.

Forward algorithm:

הרעיון: לכל מיקום $1,\ldots,L$ ולכל מצב l (מתוך m מצבים) מחשבים $F_l\left(i\right)=p\left(x_1,\ldots,x_i,s_i=l\right)$ - ההסתברות לקבל את סדרת התווים הזו כאשר המצב ה-i הוא: l.

הנוסחה הרקורסיבית:

$$\boxed{\begin{aligned}&F_l\left(i\right)=e_l\left(x_i\right)\sum_k F_k\left(i-1\right)m_{kl}\\&F_0\left(0\right)=1,\quad f_k\left(0\right)=0\ \ for\ k>0\end{aligned}}$$

פלט: (דומה ל-Viterbi רק שזה סכום במקום מקסימום)

$$\boxed{p\left(x_1,\ldots,x_L\right)=\sum_k F_k\left(L\right)}$$

בעיה-ג: מהי ההתפלגות של מצב i בהינתן סדרת התווים?

חישוב $p\left(s_i=k\mid x_1,\ldots,x_L\right)$ ל-i ולכל מצב k:

$p\left(s_i\mid x_1,\ldots,x_L\right)=\frac{p\left(s_i,x_1,\ldots,x_L\right)}{p\left(x_1,\ldots,x_L\right)}$ נרצה למקסם את המונה.

$$p\left(x_1,\ldots,x_L,S_i=l\right)=\underbrace{p\left(x_1,\ldots,x_i,S_i=l\right)}_{F_l(i)}\cdot p\left(x_{i+1},\ldots,x_L\mid x_1,\ldots,x_i,S_i=l\right)$$

Backward algorithm:

$B_l\left(i\right)=p\left(x_{i+1},\ldots,x_L\mid x_1,\ldots,x_i,S_i=l\right)$ - ההסתברות להמשך סדרת התווים בהינתן תחילת הסדרה והמצב ה-i. הנוסחה הרקורסיבית:

$$\boxed{\begin{aligned}&step: B_l\left(i\right)=\sum_k m_{lk}e_k\left(x_{i+1}\right)B_k\left(i+1\right)\\&base: B_L\left(l\right)=1\end{aligned}}$$

פלט: $\boxed{p\left(x_1,\ldots,x_L,S_i=l\right)=F_l\left(i\right)\cdot B_l\left(i\right)}$

- כדי לחשב זאת לכל המיקומים i ברצף: מריצים פעם אחת את ה-forward וה-backward ומאחסנים $F_l\left(i\right)and\ B_l\left(i\right)$ לכל l ו-i. ואז מחשבים את המכפלה: $F_l\left(i\right)B_l\left(i\right)$ לכל l ו-i.

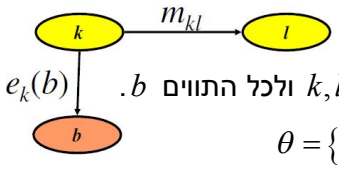
ניתוח סיבוכיות:

זמן: $O\left(m^2L\right)$ - m^2 אפשרויות למעבר ממצב אחד לאחר.

מקום: $O\left(mL\right)$ - שתי מטריצות בגודל $m\times L$ לשמירת כל ערכי ה-forward וה-backward.

©ענת עציון

Parameter Estimation for HMM



מודל HMM מוגדר ע"י הפרמטרים: m_{kl} ו-$e_k(b)$ לכל המצבים k,l ולכל התווים b.

$$\theta = \{m_{kl} \mid k,l \text{ are states}\} \cup \{e_k(b) \mid k \text{ is a state}, b \text{ is a letter}\}$$

Training Set: $\{x^1,\ldots,x^n\}$ כאשר כל x^j הוא רצף שמניחים שמתאים למודל. ה-Training set ייתכן שכולל גם את סדרות המצבים המתאימות $\{s^1,\ldots,s^n\}$.

מקרה 1- ML כאשר סדרות המצבים ידועות:

בשיטת ה-Maximum Likelihood אנחנו מחפשים פרמטרים θ^* אשר ממקסמים את ההסתברות:

$$p(x^1,\ldots,x^n,s^1,\ldots,s^n \mid \theta^*) = \max_\theta p(x^1,\ldots,x^n,s^1,\ldots,s^n \mid \theta)$$

הגדרות:

- $M_{kl} = \left|\{i \mid s_{i-1}=k, s_i=l\}\right|$ - מספר הפעמים ב-Training set שהיה מעבר מ-k ל-l.
- $E_k(b) = \left|\{i \mid s_i=k, x_i=b\}\right|$ - מספר הפעמים ב-Training set שמצב k פלט את האות b.

ההסתברות לקבלת את סדרת התווים x_j בהינתן המודל: $p(x^j \mid \theta) = \prod_{(k,l)} m_{kl}^{M_{kl}} \cdot \prod_{(k,b)} (e_k(b))^{E_k(b)}$.

מהנחת האי-תלות בין סדרות המצבים והאותיות, אם נשנה את הגדרות M_{kl} ו-$E_k(b)$ כך שייספרו עבור כל הרצפים ב-Training set אז נקבל: $p(x^1,\ldots,x^n \mid \theta) = \prod_{(k,l)} m_{ml}^{M_{kl}} \cdot \prod_{(k,b)} (e_k(b))^{E_k(b)}$.

<u>זוהי ההסתברות שנרצה למקסם.</u>

כתיבת הביטוי באופן שונה: $\prod_k \left(\prod_l m_{ml}^{M_{kl}} \right) \cdot \prod_k \left(\prod_b (e_k(b))^{E_k(b)} \right)$ מראה שניתן למקסם כל אחד משני חלקי הביטוי עבור כל k <u>בנפרד</u> (צמצום הבעיה למקסום של שני ביטויים נפרדים), כל עוד לכל מצב k מתקיימים שני התנאים: $\sum_l m_{kl} = 1$ וגם $\sum_b e_k(b) = 1$.

נפתור באופן דומה לבעיות ML שכבר ראינו: $e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$ $\quad$ $m_{kl} = \frac{M_{kl}}{\sum_{l'} M_{kl'}}$

כעת כאשר כל הפרמטרים ידועים, זהו ה-θ^* שממקסם את ה-likelihood.

מקרה 2- סדרות המצבים לא ידועות:

נרצה למקסם את $p(x \mid \theta) = \sum_s p(x,s \mid \theta)$ - כאשר הסכימה היא על כל סדרות המצבים s אשר מייצרות כפלט את הרצף x. מציאת θ^* הממקסמת את הביטוי זו **בעיית NP-Hard**.

©ענת עציון

אלגוריתם להערכת הפרמטרים- Expectation Maximization (EM):

1. נתחיל מ-θ התחלתי כלשהו.
2. נמצא θ^* כך ש- $p\left(x\mid\theta^*\right)>p\left(x\mid\theta\right)$,
3. נסמן $\theta\leftarrow\theta^*$.
4. נחזור על התהליך עד שנתכנס לפיתרון (הגעה לקריטריון כלשהו להתכנסות).

זהו אלגוריתם כללי. עבור המקרה הפרטי של HMM משתמשים באלגוריתם הבא:

Baum-Welch Training:

מתחילים מערכים כלשהם של m_{kl} ו-$e_k\left(b\right)$ אשר מוגדרים ע"י המודל ההתחלתי θ. ואז משתמשים באלגוריתם האיטרטיבי שמנסה להחליף את θ ב-θ^*.

זה נעשה ע"י "חיקוי" הצעדים במקרה-1 שבו סדרות המצבים ידועות. כאשר סדרות המצבים לא ידועות, לא ניתן לספור ולכן עושים ממוצע משוקלל על-פני כל סדרות המצבים האפשריות עבור רצף:

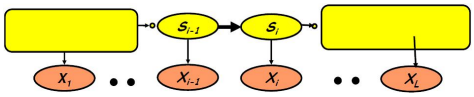
$$E_k\left(b\right)=\sum_s E_k\left(b\right)p\left(s\mid x,\theta\right) \qquad M_{kl}=\sum_s M_{kl}^s p\left(s\mid x,\theta\right)$$

(הכפלת מספר הפעמים בסדרת המצבים בהסתברות לסדרה עצמה, וכך זהו ממוצע משוקלל).

חישוב הממוצע: מכיוון שמספר סדרות המצבים הוא אקספוננציאלי ב-L משתמשים ב-DP:

DP:

Step E (Estimation):



ספירת הממוצע למעברים בין מצבים:

לכל שני מצבים $\left(k,l\right)$ ולכל קשת $s_{i-1}\rightarrow s_i$ מחשבים את הממוצע של מספר מעברי $k\rightarrow l$ על-פני הקשת, ואז M_{kl} הוא הסכום של הממוצעים על כל הקשתות.

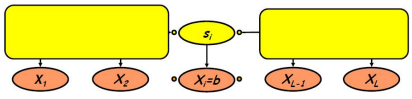
לכל אינדקס i ומצבים k,l מחשבים: $$p\left(s_{i-1}=k,s_{i-1}=l\mid x,\theta\right)=\frac{F_k\left(i-1\right)m_{kl}e_l\left(x_i\right)B_l\left(i\right)}{p\left(x\mid\theta\right)}$$

(ע"י ה-forward וה-backward). ואז סוכמים על-פני כל L הקשתות:

$$M_{kl}=\frac{1}{p\left(x\mid\theta\right)}\sum_{i=1}^{L}p\left(s_{i-1}=k,s_i=l,x\mid\theta\right)=\frac{1}{p\left(x\mid\theta\right)}\sum_{i=1}^{L}F_k\left(i-1\right)m_{kl}e_l\left(x_i\right)B_l\left(i\right)$$

טענה: כאשר ישנם n רצפים ב"ת $\left(x^1,\ldots,x^n\right)$ באורכים: $L^1,\ldots,L^n$ אז:

$$M_{kl}=\sum_{j=1}^{n}\frac{1}{p\left(x^j\right)}\sum_{i=1}^{L^j}p\left(s_{i-1}=k,s_i=l,X^j\mid\theta\right)=\sum_{j=1}^{n}\frac{1}{p\left(x^j\right)}\sum_{i=1}^{L^j}F_k^j\left(i-1\right)m_{kl}e_l\left(x_i\right)B_l^j\left(i\right)$$



ספירת המספר המצופה של פליטת אותיות:

לכל קשת $s_i\rightarrow b$ ולכל מצב k , מחשבים את הממוצע של מספר הפעמים בהן $s_i=k$. זהו ה-expected number של העברת $k\rightarrow b$ על הקשת. ו-$E_k\left(b\right)$ הוא סכום הממוצעים על כל הקשתות.

לכל אינדקס i בו $x_i=b$ ומצב k מחשבים: $$p\left(s_i=k\mid x_1,\ldots,x_L\right)=\frac{F_k\left(s_i\right)B_k\left(s_i\right)}{p\left(x_1,\ldots,x_L\right)}$$

©ענת עציון

ואז סוכמים על-האינדקסים שבהם ישנה האות b: $\boxed{E_k(b)=\frac{1}{p(x|\theta)}\sum_{i:x_i=b}F_k(i)B_k(i)}$

<u>טענה:</u> כאשר ישנם n רצפים ב"ת $(x^1,\ldots,x^n)$ אז: $\boxed{E_k(b)=\sum_{j=1}^{n}\left(\frac{1}{p(x^j)}\sum_{i:x_i^j=b}F_k^j(i)B_k^j(i)\right)}$

<u>הערה:</u> הערכים האלה מחושבים ע"י הפרמטרים של ה-θ הנוכחי באיטרציה.

<u>Step M (Maximization):</u>

שלב זה זהה לחישוב ה-ML כאשר סדרות המצבים ידועות.

$\boxed{e_k(b)=\frac{E_k(b)}{\sum_{b'}E_k(b')}}$ $\boxed{m_{kl}=\frac{M_{kl}}{\sum_{l'}M_{kl'}}}$ פרמטרים אלה מגדירים את θ^*.

מנכונות ה-EM נובע שאם $\theta^*\neq\theta$ אז: $p(x^1,\ldots,x^n|\theta^*)>p(x^1,\ldots,x^n|\theta)$.

<u>תיאור כללי של אלגוריתם ה-EM:</u>

זהו מודל עם פרמטרים- θ מעל מרחב הסתברויות- M.

<u>Observed Data:</u> $x\subseteq M$ - אוסף של מאורעות פשוטים מתוך כל מרחב ההסתברויות.

<u>הערה:</u> ב-HMM מדובר בסדרת תווים (ללא סדרת מצבים).

שיטת ה-ML מחפשת פרמטרים θ^* אשר ימקסמו את ה-likelihood של ה-Observed Data:
$p(x|\theta)=\sum_y p(x,y|\theta)$ (y - hidden data, זוהי מנייה על כל סדרות המצבים האפשריות).

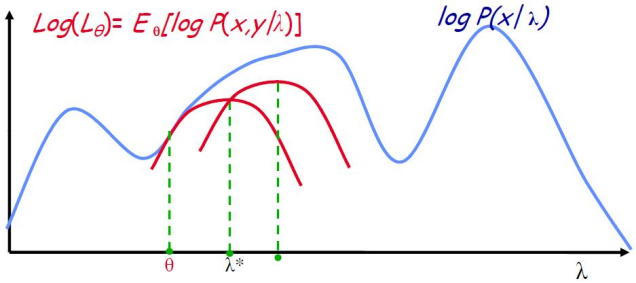
<u>קלט:</u> x,θ כאשר: $x=\{(x,y_1),\ldots,(x,y_k)\}$ (כל סדרות המצבים שמייצרות את הרצף x).

<u>פלט:</u> פרמטרים חדשים λ^* כך ש- $p(x|\lambda^*)\geq p(x|\theta)$ (שוויון רק כאשר $\lambda^*=\theta$).

האלגוריתם משתמש בפרמטרים הנוכחיים- θ, וממקסם את: $\boxed{L_\theta(\lambda)=\prod_y p(x,y|\lambda)^{p(y|x,\theta)}}$,

כלומר: בהינתן x,θ מוצאים את ה-λ שימקסם את הביטוי.

ומתקיים: אם $L_\theta(\lambda)>L_\theta(\theta)$ אז: $p(x|\lambda)>p(x|\theta)$



מסתכלים על ה-log של הביטוי:

$Q_\theta(\lambda)=\log(L_\theta(\lambda))=\sum_y p(x,y|\lambda)\log(x,y|\lambda)=E_\theta(\log p(x,y|\lambda))$.

©ענת עציון

<u>הוכחת נכונות:</u>

נסמן: $p_i = p(y_i \mid x,\theta),\ \ q_i = p(y_i \mid x,\lambda^*)$.

לפי כלל בייס: $p(x,y_i \mid \lambda^*) = q_i \cdot p(x \mid \lambda^*)$ $\qquad$ $p(x,y_i \mid \theta) = p_i \cdot p(x \mid \theta)$.

לפי הנחת ה-EM כי $L_\theta(\lambda^*) > L_\theta(\theta)$:

$$\prod_{i=1}^{k}\left(q_i \cdot p(x \mid \lambda^*)\right)^{p_i} = L_\theta(\lambda^*) > L_\theta(\theta) = \prod_{i=1}^{k}\left(p_i \cdot p(x \mid \theta)\right)^{p_i}$$

ומכיוון שנכניס את הסכימה לחזקה ו-$\sum_{i=1}^{k} p_i = \sum_{i=1}^{k} q_i = 1$ הרי שנקבל:

$$\left(\prod_{i=1}^{k} {q_i}^{p_i}\right)\cdot p(x \mid \lambda^*) > \left(\prod_{i=1}^{k} {p_i}^{p_i}\right)\cdot p(x \mid \theta)$$

נחלק את הביטוי ב- $\prod_{i=1}^{k} {q_i}^{p_i}$ ונקבל:

$$\boxed{p(x \mid \lambda^*) > \frac{\prod_{i=1}^{k} {p_i}^{p_i}}{\prod_{i=1}^{k} {q_i}^{p_i}}\, p(x \mid \theta) > p(x \mid \theta)}$$

מ.ש.ל

אי השוויון השמאלי הינו בגלל שהשבר גדול מ-1, זאת כי הפרמטרים הממקסמים את ה-likelihood של החזקות הם רק כאשר הבסיס זהה להם (באופן כללי). ולכן הביטוי במונה גדול מזה שבמכנה.

<u>דוגמא- אלגוריתם EM עבור משתנה אקראי יחיד (הטלת קוביה):</u>

מאורע פשוט נקבע ע"י אוסף הטלות של קוביה אחת, בעלת הסתברויות: $\lambda_1,\ldots,\lambda_m$.

$N_k(y)$ - מספר הפעמים שהקוביה נפלה על הפאה ה-k באירוע-y.

ההתסברות לארוע ה-y: $p(y \mid \lambda) = \prod_{k=1}^{m} {\lambda_k}^{N_k(y)}$.

N_k - מספר הפעמים שנצפה שהקוביה תיפול על הפאה ה-k:

$$N_k = E\left(N_k(y) \mid x,\theta\right) = \sum_y p(y \mid x,\theta)\cdot N_k(y).$$

<u>יישום ה-EM:</u> $L_\theta(\lambda) = \prod_y p(y \mid \lambda)^{p(y \mid x,\theta)}$, ביטוי זה ימוקסם כאשר: $\boxed{\lambda_k = \frac{N_k}{\sum_{k'} N_{k'}}}$

ואלה יהיו הפרמטרים עבור האיטרציה הבאה.

<u>Baum-Welch כאלגוריתם EM:</u>

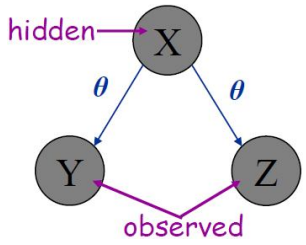
ה-observed data, אוסף של סיגנאלים: $X_1,\ldots,X_L$, שכל אחד מהם הוא סדרה של הלטות קוביה.

לכל מצב ישנן שתי הטלות קוביה: אחת לקביעת המצב הבא ואחת לקביעת הסיגנאל הנפלט.

<u>E-step:</u> מציאת ה-expected number של מספר הפעמים שכל קוביה נפלה על כל אחת מהפאות.

©ענת עציון

EM באבולוציה:



- כל זן מיוצג ע"י ווקטור בינארי $\{0,1\}^n$.
- שני זנים Y,Z התפתחו מאב משותף- Z
- כל ביט ב- X נבחר מהתפלגות יוניפורמית ויכול בהסתברות θ להשתנות לכיוון- Y או Z (לצורך הפשטות, מדובר באותו θ עבור שניהם).

בהינתן Y,Z, מהי ההסתברות- θ הסבירה ביותר, הממקסמת את הנראות של Y ו- Z.

ס"מ: מספר השינויים בין הווקטורים Y ו- Z $n_0=\left|\{i\,|\,y_i=z_i\}\right|,\ n_1=\left|\{i\,|\,y_i\neq z_i\}\right|$.

$$L(\theta)=p(Y,Z\,|\,\theta)=\prod_{1\le i\le n}p(Y_i,Z_i\,|\,\theta)=\underbrace{\left(\tfrac{1}{2}\left(\theta^2+(1-\theta)^2\right)\right)^{n_0}}_{probability\ for\ complementary}\cdot\underbrace{\left(\theta(1-\theta)\right)^{n_1}}_{prob.\ for\ non-complementary}$$

חישוב ההסתברות לשוויון או לאי-שוויון בין הביטים:

דוגמא- השורה הראשונה בטבלה:

1/2- בחירת הביט ל- X ; $(1-\theta)^2$- ההסתברות לכך שהביט לא השתנה פעמיים.

Y_i	Z_i	X_i	$\Pr[X_i,Y_i,Z_i]$
b	b	b	½(1-θ)²
		1-b	½θ²
b	1-b	b	½ θ(1-θ)
		1-b	½ θ(1-θ)

$b\in\{0,1\}$

Flip- ביט התהפך=קרתה מוטציה.

E-step: חישוב תוחלת מספר הפעמים שקרתה מוטציה.

$$E(\#flips)=\sum_{i=1..n}\left(\Pr[x_i\neq y_i]+\Pr[x_i\neq z_i]\right)=2n_0\cdot\left(\frac{\theta^2}{(1-\theta)^2+\theta^2}\right)+n_1$$

#flips = sum of indicator variables

(כאשר יש אי-שוויון- flip אחד, וכאשר יש שוויון- 2 או 0).

M-step: $\theta'=E(\#flips)/2n$

פתרון אנליטי:

- $L(\theta)=\left(\tfrac{1}{2}(\theta^2+(1-\theta)^2)\right)^{n_0}\cdot(\theta(1-\theta))^{n_1}$
- $l(\theta)=\log(L(\theta))=n_0\cdot\log\left(\tfrac{1}{2}(\theta^2+(1-\theta)^2)\right)+n_1\cdot\log(\theta(1-\theta))$
- Find extreme-points of log-likelihood:

$$l(\theta)'=\frac{n_0(2\theta-1)}{\tfrac{1}{2}(\theta^2+(1-\theta)^2)}+\frac{n_1(1-2\theta)}{\theta(1-\theta)}=$$

$$\frac{(1-2\theta)}{\tfrac{1}{2}(\theta^2+(1-\theta)^2)\theta(1-\theta)}\left(-n_0\cdot\theta(1-\theta)+n_1\cdot\tfrac{1}{2}(\theta^2+(1-\theta)^2)\right)=$$

$$\frac{(1-2\theta)\left((n_1+n_0)\theta^2-(n_1+n_0)\theta+\tfrac{1}{2}n_1\right)}{\tfrac{1}{2}(\theta^2+(1-\theta)^2)\theta(1-\theta)}=\frac{(1-2\theta)n\left(\theta^2-\theta+n_1/2n\right)}{\tfrac{1}{2}(\theta^2+(1-\theta)^2)\theta(1-\theta)}=0$$

$$\theta=\tfrac{1}{2},\tfrac{1}{2}\pm\frac{\sqrt{1-2n_1/n}}{2}$$

minimum — maxima

©ענת עציון

לוקוס ה-ABO:

זהו לוקוס הקובע את סוג הדם.

6 גנוטיפים: $\{a/a, a/o, b/o, b/b, a/b, o/o\}$.

4 פנוטיפים: $\{A, B, AB, O\}$.

בהינתן פנוטיפים של אוכלוסייה, האם ניתן לשחזר ממנה את התפלגות הגנוטיפ?

הרדי-ויינברג: שכיחות האללים בין שני כרומוזומים שונים היא בלתי-תלוייה.

ההסתברויות עבור הפנוטיפים:

$$\begin{aligned}
p(P=A\mid\Theta) &= \theta_{a/a}+\theta_{a/o}=\theta_a^2+2\theta_a\theta_o\\
p(P=B\mid\Theta) &= \theta_{b/b}+\theta_{b/o}=\theta_b^2+2\theta_b\theta_o\\
p(P=AB\mid\Theta) &= \theta_{a/b}=2\theta_a\theta_b\\
p(P=O\mid\Theta) &= \theta_{o/o}=\theta_0^2
\end{aligned}$$

מחפשים פרמטרים של הסתברות לאללים שימקסמו את הנראות של המידע מהאוכלוסייה.

ס"מ: מספר הפעמים שהופיע כל אחד מסוגי הפנוטיפים באוכלוסייה.

מגדירים מהם כל הגנוטיפים, ולכל אחד: מה ההסתברות, מה הפנוטיפ (ה-observed) ומהי התרומה שלו לס"מ (gene count):

genotype	prob	pheno-type	gene count		
			a	b	o
a/o	$2\theta_a\theta_o$	A	$1^*\frac{2\theta_o}{2\theta_o+\theta_a}$ $+2^*\frac{\theta_a}{2\theta_o+\theta_a}$	0	$1^*\frac{2\theta_o}{2\theta_o+\theta_a}$
a/a	θ_a^2				
b/o	$2\theta_b\theta_o$	B	0	$1^*\frac{2\theta_o}{2\theta_o+\theta_b}$ $+2^*\frac{\theta_b}{2\theta_o+\theta_b}$	$1^*\frac{2\theta_o}{2\theta_o+\theta_b}$
b/b	θ_b^2				
a/b	$2\theta_a\theta_b$	AB	1	1	0
o/o	θ_o^2	O	0	0	2

result(s) of die tosses — observed outcome of "experiment" — Expected count of die-toss results

E-step:

$$\begin{aligned}
E[\#a] &= n_A\cdot\frac{2(\theta_o+\theta_a)}{(2\theta_o+\theta_a)}+n_{AB}\cdot 1\\
E[\#b] &= n_B\cdot\frac{2(\theta_o+\theta_b)}{(2\theta_o+\theta_b)}+n_{AB}\cdot 1\\
E[\#o] &= n_A\cdot\frac{2\theta_o}{(2\theta_o+\theta_a)}+n_B\cdot\frac{2\theta_o}{(2\theta_o+\theta_b)}+n_O\cdot 2
\end{aligned}$$

M-step:

$$\theta_a'=\frac{E[\#a]}{2n}\quad;\quad\theta_b'=\frac{E[\#b]}{2n}\quad;\quad\theta_b'=\frac{E[\#b]}{2n}$$

©ענת עציון

עצים פילוגנטיים

התיאוריה של האבולוציה:

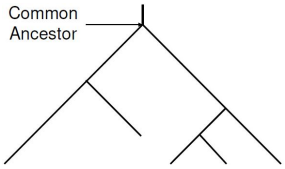
Speciation: ארועים מובילים ליצירה של זנים חדשים. היא נגרמת ע"י הפרדה לקבוצות שונות כאשר שונות גנטית מסויימת נעשית דומיננטית (שלטת).

- לכל שני זנים ישנו אב קדמון משותף (יכול להיות גם מאוד רחוק מהם).

עץ פילוגנטי: עץ המתאר את אוסף אירועי ה-speciation (הפיצולים בעץ) שהובילו ליצירת סט של זנים הקיימים כיום.

עלים: זנים הקיימים היום

צמתים פנימיים: אבות קדמונים משותפים

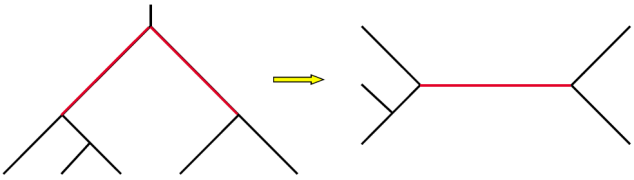


אורך קשת: "זמן" מאירוע speciation אחד לאחר.

Rooted tree: מודל שטבעי לחשוב עליו-

LUCA: Last Universal Common Ancestor.

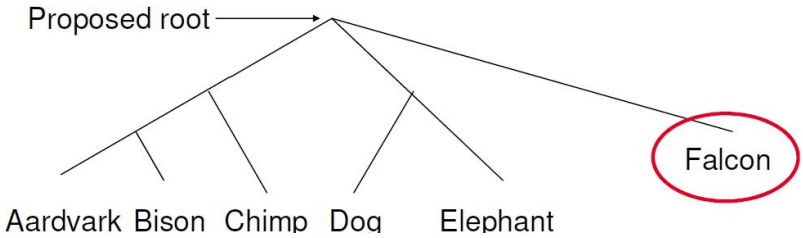
Unrooted tree: מייצג את אותו הדבר ללא השורש



הערה: ניתן להפוך Unrooted ל-rooted ע"י בחירת מיקום השורש.

Outgroup: זן שהוא בוודאות יותר רחוק משאר הזנים הנבדקים בעץ.

ואז ניתן לקבוע את ה-root החדש כמקום החיבור של זן זה לשאר העץ.



שוויון בין עצים:

T_1, T_2 מעל אותם הזנים נחשבים זהים אם הם מייצגים את אותה האבולוציה.

Tree isomorphism: פונקציה $h: T_1 \to T_2$ שממפה כל קודקוד בעץ אחד לקודקוד בעץ השני.

אם קיימת פונקציה כזו ← העצים שקולים.

בניית עץ פילוגנטי:

- פילוגנזה קלאסית: לפי מאפיינים מורפולוגיים: מספר רגלים, אורך הרגליים וכו'.
- פילוגנזה מודרנית: לפי מאפיינים מולקולאריים: רצפי הגנים ורצפי החלבונים.

בדרך-כלל נסתכל על אוסף של **רצפים הומולוגיים**.

אלה יהיו רצפים שהם significant characters אשר מבדילים בין זנים שונים.

אורתולוגים: רצפים שנוצרו מאותו המקור בעקבות **speciation event**.

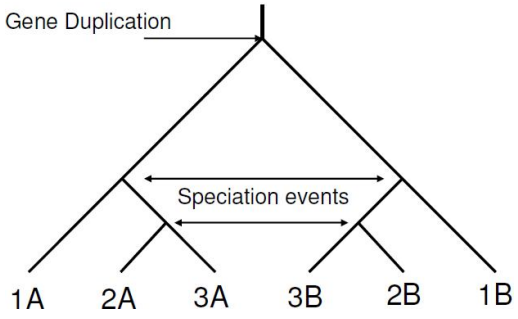
פרלוגים: באותו אב קדמון, הגן השתכפל בגנום- **duplication event**, וכל עותק התחיל לעבור שינויים בנפרד ורק אז היה speciation event.

©ענת עציון

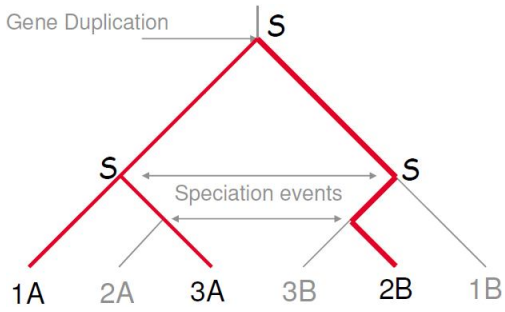
סכנה בהסתכלות על פרלוגים: יובילו לעץ שגוי מבחינת טופולוגיה של זנים.

דוגמא:

העץ המקורי:



אם נסתכל בעץ על הרצפים: 1A, 2B, 3A בתור הרצפים המייצגים את 3 הזנים, נקבל עץ לא נכון:



Duplication events לא יוצרים זנים חדשים.

הנחה: כל הרצפים הנתונים עבור הזנים הם אורתולוגים- נוצרו מאב משותף ע"י speciation events.

הבעיה הגדולה: בניית עץ פילוגנטי בהינתן n רצפים לעלים.

הבעיה הקטנה: בהינתן עץ פילוגנטי וערכי העלים, מציאת השמת רצפים לצמתים הפנימיים, וחישוב מידת ה-likelihood של העץ.

שילוב הבעיות: כדי לפתור את הבעיה הגדולה, ניתן לפתור את הבעיה הקטנה לכל הטופולוגיות עצים האפשריות של עצים בעלי n עלים, ולהוציא כפלט את העץ בעל הציון likelihood הגבוה ביותר.

Character-based methods:

בניית עצים ע"י השוואת characters של הרצפים המתאימים. כל אות היא תכונה משמעותית אשר תבדיל בין זנים שונים. שתי שיטות: "Perfect Phylogeny" ו-"Maximum Parsimony".

הנחות:

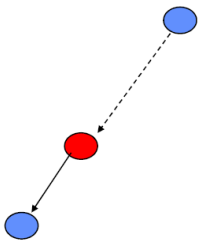
- חוסר תלות בין תכונות.
- עץ טוב: מספר מינימאלי של שינויים על-פני ענפי העץ.

Character: תכונה שהיא סיגניפיקנטית ומבדילה בין זנים שונים.

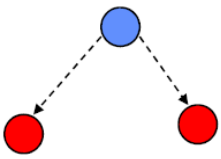
Character state: הערך של התכונה.

הנחה: לא סביר ש-state כלשהו של תכונה ייווצר פעמיים בעץ אבולוציוני, כלומר, אין **reversal** ואין **convergence**. עץ שמקיים את ההנחה נקרא- Homoplasy free.

Reversal: מוטציה ש"חזרה אחורה".



Convergence: תכונה שהתפתחה במקביל ב-2 מקומות שונים



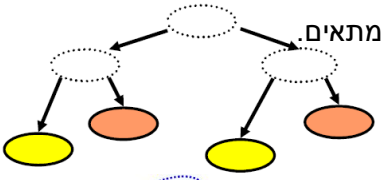
©ענת עציון

Characters=coloring:

Coloring: של עץ T הוא מיפוי $C:V\to[set\ of\ colors]$.

Partial coloring: של עץ T מוגדר על קבוצת צמתים $U\subseteq V$ - $C:U\to[set\ of\ colors]$.

כל תכונה (character) מגדירה partial coloring של העץ הפילוגנטי המתאים.

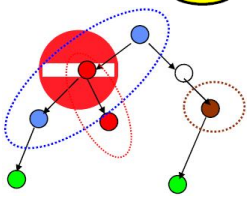


Convex coloring:

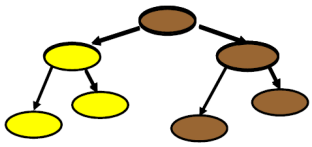
יהי $T=(V,E)$ עץ שהוא צבוע חלקית. ויהי d אחד הצבעים.

d-carrier: תת-עץ מינימאלי של T המכיל צמתים שכולם צבועים ב-d.

טענה: צביעה (מלאה או חלקית) של העץ היא קונווקסית אם"ם כל ה-d carriers לא נחתכים.



טענה: תכונה היא homoplasy free אם"ם הצביעה (החלקית) המתאימה היא קונווקסית.



The Perfect Phylogeny:

קלט: מטריצה של species-characters.

פלט: עץ בעל n עלים המתאים לרצפים הניתנים כקלט שהוא homoplasy free (אם קיים כזה), כלומר, קיימת השמה של צביעה קונווקסית לכל צמתי העץ.

הערה: הבעיה הזו היא NP-hard.

פישוט ההגדרות לווקטורים בינאריים:

0- התכונה לא קיימת

1- התכונה קיימת.

נתחיל משורש ובו ווקטור של אפסים. ניתן להניח שבכל פיצול משתנה תכונה אחת.

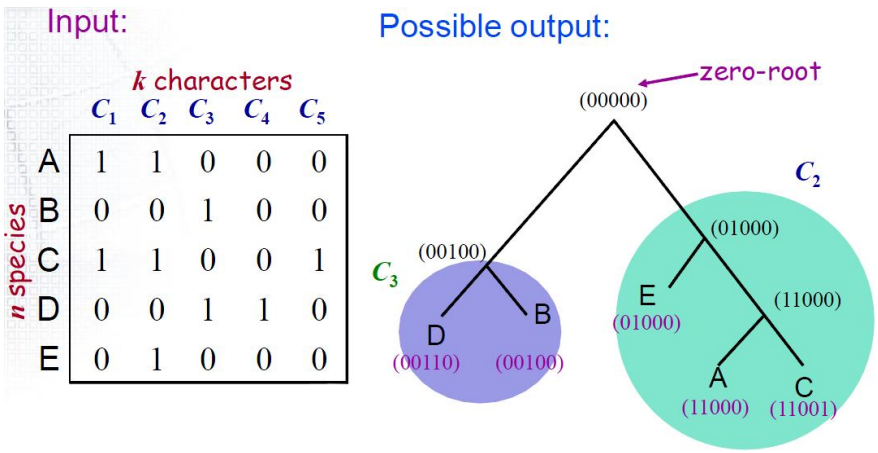
קלט: k צביעות (ווקטורים) בינאריות $C_1,\ldots,C_k$ עבור כל הזנים הנתונים.

פלט:

1. עץ פילוגנטי rooted עבור על הזנים הנתונים.
2. k צביעות בינאריות $C'_1,\ldots,C'_k$ של צמתי T אשר משרות ווקטור אפסים בשורש ומתאימות לצביעה התונה של העלים.

(או הודעה כי לא קיים כזה עץ).

דוגמא:



אבחנה: עץ הינו perfect phylogeny אם"ם ניתן למפות כל תכונה לצומת שבו התכונה "נדלקה".

©ענת עציון

פתרון בזמן פולינומי- מטריצות למינריות:

O_t: תת-קבוצה של זנים בעלי התכונה C_t ($O_t = \{i \mid M_{it} = 1\}$).

קבוצה למינרית: קבוצת של קבוצות שכל שתיים הן זרות או שאחת מכילה את השנייה.

טענה: למטריצה בינארית M יש עץ שהוא perfect phylogeny אם"ם אוסף הקבוצות $\{O_1,\ldots,O_k\}$ הוא קבוצה למינרית.

דוגמא:

Laminar

	C_1	C_2	C_3	C_4	C_5
A	1	1	0	0	0
B	0	0	1	0	0
C	1	1	0	0	1
D	0	0	1	1	0
E	0	1	0	0	0

Not Laminar

	C_1	C_2	C_3	C_4	C_5
A	1	1	0	0	0
B	0	0	1	0	1
C	1	1	0	0	1
D	0	0	1	1	0
E	0	1	0	0	1

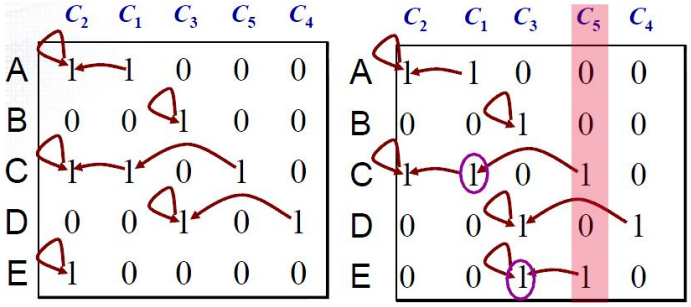
המטריצה הימנית אינה למינרית כי $O_2 = \{A,C,E\}$ ו- $O_5 = \{B,C,E\}$ וקבוצות אלה אינן שוות ואף-אחת לא מכילה את השנייה.

מימוש יעיל:

1. ממיינים את עמודות המטריצה (ה-characters) לפי ערך בינארי יורד.

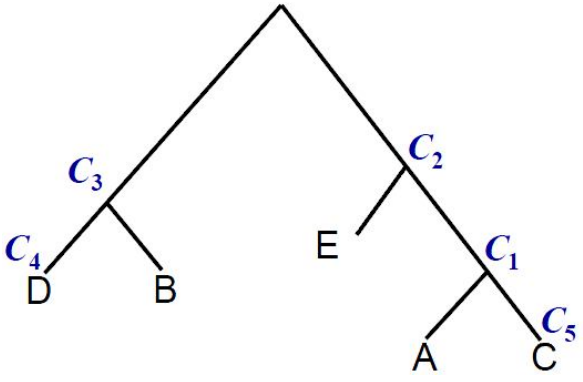
טענה: אם הערך הבינארי של עמודה i גדול מהערך הבינארי של עמודה t, O_i לא יהיה מוכל ב- O_t. (הוכחה: $C_i > C_t \leftarrow$ לא כל ה 1-ים ב- C_i מכוסים ע"י ה 1-ים ב- C_t).

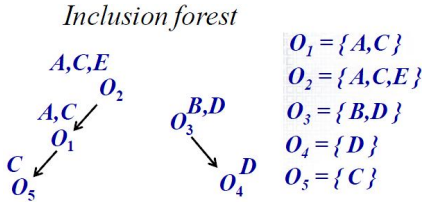
2. מייצרים מצביעים מכל 1 במטריצה, ל- 1 הקרוב ביותר המצוי לפניו באותה השורה.



טענה: אם העמודות ממויינות, הן מהוות קבוצה למינרית אם"ם לכל עמודה i, כל ה 1-ים בעמודה זו מצביעים לאיברים באותה העמודה.

3. אם המטריצה למינרית- הופכים את המצביעים לקבלת inclusion tree. מוסיפים לו עלים- כל עלה יחובר לצומת הקבוצה הכי תחתונה שהוא מופיע בה. ומוסיפים שורש שיחובר לכל מקורות הגרף.





סיבוכיות: 1. $O(nk)$ - מיון העמודות ע"י bucket sort. 2. $O(nk)$ - הוספת מצביעים מכל ה 1-ים.

3. $O(n)$ - בניית העץ מה-inclusion tree. בסה"כ: $O(nk)$.

©ענת עציון

Maximum Parsimony:

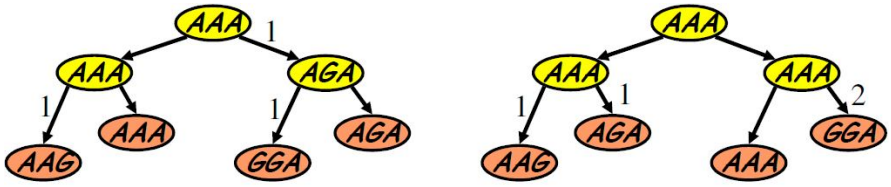
גישה זו היא הגישה הכי מקובלת כיום בתחום. (Perfect Phylogeny לא רק קשה לחישוב, לפעמים עץ כזה לא קיים). בגישה זו מוצאים עץ ובו מספר מינימאלי של שינויים על-פני הענפים.

קלט: h רצפים עבור הזנים השונים, כולם באורך m.

פלט: עץ ובו רצפי הקלט מופיעים בעלים, וההשמה לצמתים הפנימיים מקיימת שהמספר הכולל של החלפות על-פני הקשתות הוא מינימאלי.

Parsimony score: מספר ההחלפות הכולל על פני קשתות העץ.

דוגמא:



The left tree is preferred over the right tree.

הבעיה הקטנה: מציאת המספר המינימאלי של החלפות עבור עץ מסוים.

הבעיה הגדולה: מציאת עץ שממזער את מספר השינויים.

אלגוריתם Fitch:

קלט: rooted tree ורצפי העלים שלו.

פלט: השמת ערכי רצפים לצמתים הפנימיים, לכל מיקום ברצף באופן נפרד.

- מסיירים על העץ ב-post order. לכל צומת מגדירים קבוצה של מצבים אפשריים. לכל עלה מוגדר מראש מצב אחד אפשרי, לפי רצף הקלט.

הגדרת קבוצת המצבים האפשריים R_i עבור צומת פנימי i לפי קבוצות המצבים של הבנים שלו:

$$R_i = \begin{Bmatrix} R_j \cup R_k & if\ R_i \cap R_k = \varnothing \\ R_j \cap R_k & otherwise \end{Bmatrix}$$

טענה: מספר ההחלפות בפיתרון האופטימאלי = מספר פעולות האיחוד.

- מסיירים על העץ ב-pre order (מהשורש לעלים) ולכל צומת פנימי j שהאב שלו הוא i בוחרים מצב r_j באופן הבא:

$$r_j = \begin{Bmatrix} r_i & if\ r_i \in R_j \\ arbitrary\ state \in R_j & otherwise \end{Bmatrix}$$

כלומר: 0 החלפות הינה אפשרות המועדפת.

הערה: ניתן לבחור בכל-פעם עבור השורש מתוך קבוצת המצבים שלו, ולראות איזו בחירה ממזערת את מספר ההחלפות. מהרגע שהשורש נבחר, כל שאר ההשמות ייקבעו לפי ההגדרה.

ניתוח סיבוכיות:

$O(nk)$ - כאשר n הוא מספר העלים ו- k הוא מספר המצבים האפשריים. ישנם $O(n)$ צמתים ו- $O(k)$ עבר פעולת איחוד בצומת.

התהליך הזה מתבצע בנפרד לכל אחת מ- m ה-characters (המיקומים ברצף).

ובסה"כ: $O(nmk)$.

©ענת עציון

האלגוריתם של Sankoff (Weighted Parsimony):

לכל מוטציה ישנה עלות שונה. מוטציה $a \leftrightarrow b$, עלותה נתונה ע"י: $S(a,b)$.

Bottom-up phase: מגדירים $R_i(s)$ - העלות האופטימאלית של תת-העץ של i כאשר הצומת מקבל את ההשמה למצב- s.

Top-down phase: בוחרים מצבים אופטימאליים לכל אחד מהצמתים הפנימיים.

הערה: נעשה בדומה לאלגוריתם של lifted tree alignment מתרגול-5.

חישוב $R_i(s)$:

אתחול: $R_i(s)=\begin{cases}0 & if\ s_i=s\\ \infty & otherwise\end{cases}$ (כלומר-0 לעלים ו- ∞ לצמתים הפנימיים).

צעד רקורסיבי: עוברים על העץ ב-post order ומחשבים לצומת i שילדיו הם j,k :

$R_i(s)=\min_{s'}\{R_j(s')+S(s',s)\}+\min_{s''}\{R_k(s'')+S(s'',s)\}$ - סכום הציונים המינימאליים של הבנים בתוספת חישוב עלות המוטציה על הקשתות $(i,j),(i,k)$. תוך כדי המעבר מחזיקים מצביעים אל המצב בצמתי הבנים שהביא לציון המינימאלי.

Backtracking של המסלול: בוחרים את המצב המינימאלי עבור השורש- $\min_s\{R_{root}(s)\}$. מבצעים סיור Pre-order ובחירת המצבים לפי המצביעים שנשמרו קודם.

ניתוח סיבוכיות:

מימדי טבלת ה-DP הם $m\times n$ (שמירת $R_i(s)$ לכל צומת $1\le i\le n$ ולכל מצב $1\le s\le m$).

כל תא ממלאים ב-$O(m)$, כי עוברים על כל מצבי שני הבנים הרלוונטיים.

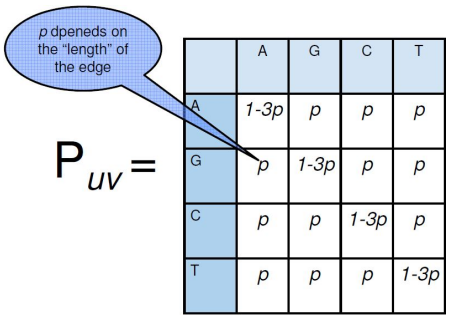
התהליך מתבצע לכל k הנוקליאוטידים.

בסה"כ: $O(m^2nk)$.

האם Maximum Parsimony טוב לבניית עצים פילוגנטיים?

כאשר התכונות הן DNA, מודלים מקובלים מניחים שמוטיציות הן אירועים רנדומאליים.

האם Maximum Parsimony זו שיטה טובה במודלים כאלה?



Jukes Cantor- מודל הסתברותי לתיאור האבולוציה:

1. מוטציות במקומות שונים ברצף הן ב"ת וש"ה (i.i.d).
2. בכל קשת בעץ הפילוגנטי, לכל מוטציה יש אותה הסתברות.

הערה: ההסתברות למוטציה כלשהי פרופורציונית לאורך ה"ענף" בעץ.

פלט המודל מכיל:

- עץ מכוון $T=(V,E)$.
- התפלגות של אותיות DNA עבור השורש (המודל מניח התפלגות יוניפורמית).
- השמה של מטריצות המעבר של JC לעץ.

קונסיסטנטיות של שיטות בנייה:

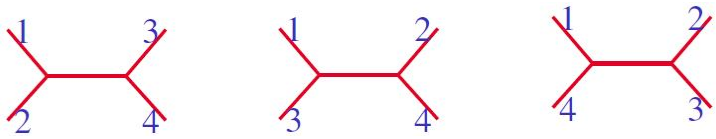
שיטת בניית עצים נחשבת קונסיסטנטית עם מודל מסויים אם מתקיים: כאשר נפעיל את השיטה הזו על רצפים שאורכם שואף ל- ∞, העץ שייבנה הוא בהסתברות גבוהה העץ הנכון.

©ענת עציון

הערה: ניתן לבדוק זאת ע"י עץ נתון, שממנו בונים רצפים ואז משתמשים בשיטה לשחזור העץ.

דוגמא- בניית עץ ל-4 מינים:

ישנן 3 אפשרויות לטופולוגיות שונות.

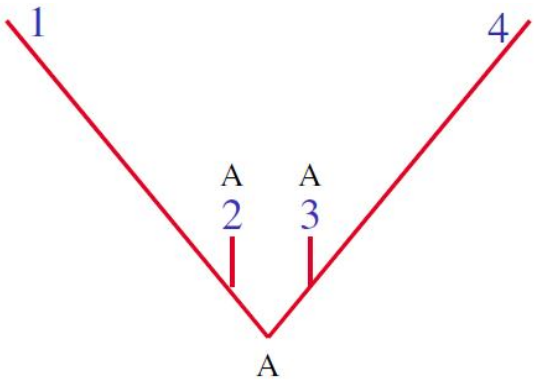


האם השיטה Maximum Parsimony תדע למצוא את הנכונה? לא תמיד...

Maximum Parsimony לא קונסיסטנטית עבור מודל ה-JC, כי עבור חלק מהמקרים באבולוציה, העץ הפרסימוני ביותר הוא בהסתברות גבוהה שונה מהעץ האמיתי.

עץ שקשה יהיה לשחזר ע"י השיטה:

- אין מוטציה- (1 ו-4 הם גם A), לכל 3 הטופולוגיות יש ציון 0.
- מוטציה אחת (רק ב-4 יש G), לכל 3 הטופולוגיות יש ציון 1.
- שתי מוטציות שונות (ב-1 יש C וב-4 יש G), לכל 3 הטופלוגיות יש ציון 2.
- שתי מוטציות זהות (1 ו-4 הם C), טופולוגיה לא נכונה מקבלת ציון 1 והאחרות ציון 2.



כדי שהאלגוריתם יצליח למצוא את הטופולוגיה הנכונה של העץ, 2 ו-3 צריכים להיות בעלי אותיות שונות, מה שפחות סביר שיקרה אבולוציונית, עקב אורך הענפים הקצר.

Maximum Parsimony זו שיטה נפוצה ומשתמשים בה בעיקר כשיודעים שאורכי הענפים שווים.

בעיות של Maximum Likelihood:

ML-The big problem: בהינתן רצפים עבור העלים, למצוא עץ ומשקלים עלך הקשתות שימקסמו את הנראות: $P(data \mid tree \,\&\, edge\ weights)$.

ML-The small problem: בהינתן רצפים עבור העלים ועץ, למצוא את משקלי הקשתות שימקסמו את הנראות.

ML-The tiny problem: בהינתן רצפים עבור העלים, עץ ומשקלי הקשתות, למצוא מה ה-likelihood של המידע.

ML-The tiny problem:

סימונים: T - העץ הפילוגנטי, D - אוסף של n רצפים, θ -פרמטר להתפלגות על הקשתות.

שתי וריאציות:

- Tiny ML: חישוב הנראות של המידע בהתחשב בכל ההשמות האפשריות לצמתים הפנימיים $P(D \mid T,\theta) = \sum_A P(D,A \mid \theta)$.
- Tiny AML: מציאת ההשמה לצמתים הפנימיים אשר ממסקמת את הנראות של המידע $\arg\max_A P(D,A \mid T,\theta)$.

©ענת עציון

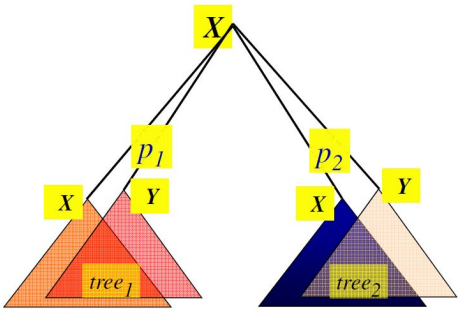
לצורך פשטות: נניח שב-DNA ישנם שני מצבים בלבד: X ו-Y. ולכל קשת e יש התפלגות p_e להחלפה בין האותיות.

ניתן לחשב רצף מורכב יותר על הקשת e: $XXYXY \Leftrightarrow YXYXX$ - $p_e^2(1-p_e)^3$.

בהינתן העץ השלם, ה-likelihood נקבע ע"פ ערכי ה-p_e של הקשתות.

מניחים שכל מיקום (position) ברצף התפתח באופן בלתי-תלוי באחרים ולכן ניתן לחשוב בנפרד.

$$P(D|T,\theta)=\prod_i P\left(D^{(i)}|T,\theta\right)$$



אלגוריתם Felsenstein – Tiny problem-ל DP:

לכל צומת בעץ מחשבים את $L_X(D|T,\theta)$, $L_Y(D|T,\theta)$ - ה-likelihood של התת-עץ של צומת זה עם האות X או Y בהתאמה.

בעזרת ערכים אלה מחשבים את ה-likelihood של העץ כולו:

$$L(D|T,\theta)=L_X(D|T,\theta)+L_Y(D|T,\theta)$$

חישוב L_X:

$$L_X(D|T,\theta)=\left[L_X(D|Tree_1,\theta)(1-p_1)+L_Y(D|Tree_1,\theta)p_1\right]\cdot\left[L_X(D|Tree_2,\theta)(1-p_2)+L_Y(D|Tree_2)p_2\right]$$

חישוב L_Y:

$$L_Y(D|T,\theta)=\left[L_Y(D|Tree_1,\theta)(1-p_1)+L_X(D|Tree_1,\theta)p_1\right]\cdot\left[L_Y(D|Tree_2,\theta)(1-p_2)+L_X(D|Tree_2)p_2\right]$$

אתחול: אם T הוא עלה שמכיל X אז: $L_X(D|T,\theta)=1, L_Y(D|T,\theta)=0$.

האלגוריתם מתחיל מהעלים וממשיך למעלה לכיוון השורש, ולכל צומת שומרים את שני הערכים הללו מה שמאפשר חישוב הערכים הבאים במעלה העץ ע"י 3 הכפלות ושתי פעולות חיבור- $O(1)$.

ולחישוב עבור כל הצמתים הפנימיים- $O(n)$.

הערות:

- בעצים שאינם בינאריים יהיו יותר מכפלות לחישוב בכל צומת.
- אם העץ הוא unrooted ניתן לבחור אקראית שורש לעץ.
- בא"ב בגודל 4 יש לשמור 4 ערכים לכל צומת בעץ: L_A, L_C, L_G, L_T.

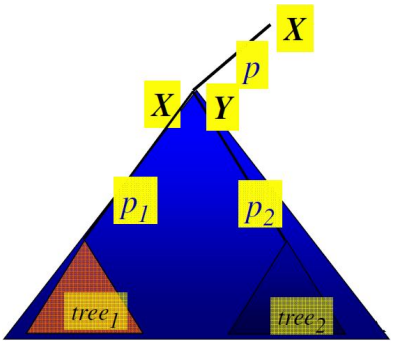
Kimura's K2P model:

מודל JC לא לוקח בחשבון שהסתברות ההחלפה בין פורינים $A \leftrightarrow G$ ובין פירימידינים $C \leftrightarrow T$ הן שונות מאלה שבין פורין לפירימידין. במודל זה משתמשים במטריצות החלפות שונה:

α - החלפה בין פורינים/פירימידינים לבינם. β - החלפה בין שני הסוגים השונים.

$$\text{Subst. Prob.} = \begin{array}{c} \\ A \\ C \\ G \\ T \end{array}\begin{array}{c} \begin{array}{cccc} A & C & G & T \end{array} \\ \begin{bmatrix} 1-2\beta-\alpha & \beta & \alpha & \beta \\ \beta & 1-2\beta-\alpha & \beta & \alpha \\ \alpha & \beta & 1-2\beta-\alpha & \beta \\ \beta & \alpha & \beta & 1-2\beta-\alpha \end{bmatrix} \end{array}$$

©ענת עציון



Tiny AML:

$L^E(D|T,\theta)$ - ה-"ancestral likelihood" של T עם E (X או Y) בצומת האב של T.

לפי ההגדרה של ML (מקסום) חישוב L_X:

$$L^X(D|T,\theta)=\max\begin{Bmatrix}(1-p)L^X(D|tree_1,\theta)\cdot L^X(D|tree_2,\theta),\\ pL^Y(D|tree_1,\theta)\cdot L^Y(D|tree_2,\theta)\end{Bmatrix}$$

<u>בשורש:</u> בוחרים את האות $E\in\{X,Y\}$ שממקסמת את הביטוי

$L^E(D|tree_1,\theta)L^E(D|tree_2,\theta)$

<u>בעלים:</u> $L^X(D|T,\theta)=1-p$ אם העלה מסומן כ- X ו- p אחרת.

<u>סיבוכיות:</u> יורדים למטה בעץ ולכל צומת בוחרים את האות שתמקסם את ה-likelihood בהינתן האות שבצומת האב. לכל הצמתים הפנימיים: $O(n)$.

<u>הערה:</u> צריך לשמור את הערך שממנו הגענו (מצביע) כדי לדעת איזה ערך נתן את המקסימום ומהי ההשמה בסוף האלגוריתם לכל הצמתים הפנימיים.

ML-The small problem:

<u>Hill climbing:</u> שיטה היוריסטית, שבה בהינתן פתרון התחלתי, מוסיפים/מורידים ε מההסתברות למוטציה בכל אחת מהקשתות בנפרד ובודקים האם זה שיפר את ה-fitness.

לבסוף מגיעים למקסימום מקומי.

<u>Consistency & Efficiency:</u>

method	consistency	efficiency
MP	No	No (NP complete)
AML	No	No (NP complete)
ML	Yes !	No (NP complete)

<u>Distance-based methods לבניית עצים פילוגנטיים:</u>

<u>הרעיון:</u> ההבדלים בין מינים מומרים למרחקים ועץ בעל משקלים על הקשתות נבנה על-סמך המרחקים הללו. המרחקים מיוצגים ע"י מטריצה המקיימת מטריקה של מרחק.

<u>מטריקת מרחק:</u> פונקציה המוגדרת על סט M של L אובייקטים $d:M\times M\to R^+$ שמקיימת:

- $d(i,i)=0$ ועבור $i\neq j$, $d(i,j)>0$.
- $d(i,j)=d(j,i)$ (ולכן המטריקה מיוצגת ע"י מטריצה סימטרית שהאלכסון שלה הוא אפסים).
- לכל i,j,k מתקיים אי-שוויון המשולש: $d(i,k)\le d(i,j)+d(i,k)$

©ענת עציון

האם מובטח שלכל מטריצה המקיימת מטריקה של מרחק קיים עץ? לא.

אינטואיציה: במטריצה ישנם n^2 ערכים ובעץ ישנן $O(n)$ קשתות ולכן לא כל מטריצה מתאימה לעץ.

מטריקה אולטרמטרית: מתאימה לעץ פילוגנטי שבו המרחק מהשורש לכל אחד מהעלים הוא שווה.

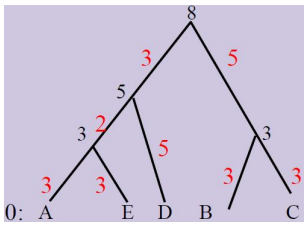
מטריקה אדטיבית/עץ: מתאימה לעץ פילוגנטי שהמרחק בין כל שני עלים שווה לסכום המשקל בין שני הצמתים.

שעון מולקולארי- עץ אולטרמטרי:

הנחה: קצב השינויים על קשת בעץ הוא פרופורציונלי למשך הזמן שעבר בין שני המינים.

מסקנה: המרחק מכל צומת פנימי לעלים בתת-העץ שלו הוא זהה.

עץ אולטרמטרי: עץ בעל שורש ומשקלים על הקשתות שכל העלים שלו באותו מרחק מהשורש.

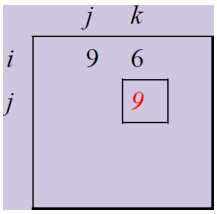


הגדרת הגובה בעץ: עלים מוגדרים כגובה-0 והמרחקים על הקשתות מתאימים לגובה הצמתים הפנימיים בעץ.

Least common ancestor (LCA): האב המשותף הקרוב ביותר של שני עלים בעץ.

$\boxed{height(LCA(i,j)) = 0.5dist(i,j)}$ - המרחק של ה-LCA משני העלים, שווה לחצי המרחק בין שני העלים.

מטריצה אולטרמטרית: מטריצת מרחק U בגודל $L \times L$ שבה לכל 3 אינדקסים i,j,k מתקיים:



$U(i,j) \le \max\{U(i,k), U(j,k)\}$. **כלל 3 הנקודות**

טענה: U היא מטריצה אולטרמטרית אם"ם קיים עץ אולטרמטרי כך שהמטריצה במקום (i,j) מייצגת את ה-LCA שלהם. $\boxed{U(i,j) = height(LCA(i,j)) = 0.5dist(i,j)}$

אלגוריתמים לבניית עצים אולטרמטריים:

קלט: מטריצת מרחק מעל הסט S.

פלט: עץ אולטרמטרי מעל האובייקטים ב-S.

תכונות נדרשות:

- קונסיסטנטיות: אם המטריצה היא אולטרמטרית אז האלגוריתם צריך להחזר את העץ המתאים.
- רובסטיות: אם המטריצה האינה אולטרמטרית, האלגוריתם צריך להחזיר עץ אולטרמטרי "קרוב" למטריצה.

UPGMA-Neighbor Joining:

הרעיון: בוחרים שני צמתים i,j מחליפים אותם בצומת אב v. מורידים אותם מהעץ ומחשבים ל-v את המרחק מכל שאר צמתי העץ.

אתחול: כל אובייקט (מין) הוא קלסטר נפרד, וכולם ממוקמים בגובה-0.

איטרציה: איחוד שני קלסטרים i,j "קרובים ביותר" לקלסטר חדש v (שיופיע במקומם).

המרחק של v לכל קלסטר k הוא המרחק הממשוקל של k מ-i ו-j:

$$\boxed{D(v,k) = \alpha D(i,k) + (1-\alpha) D(j,k)}, \quad \boxed{\alpha = \frac{|C_i|}{|C_i| + |C_j|}}$$

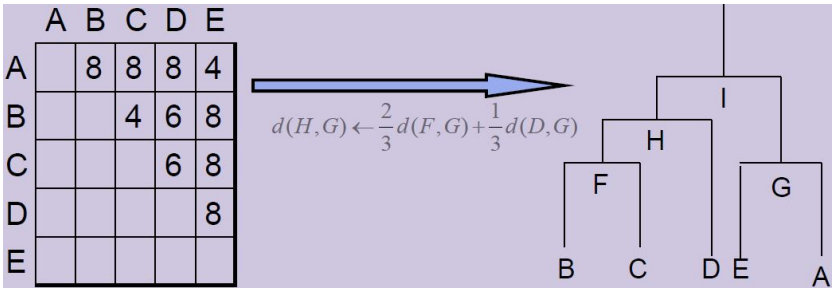
הערה: זהו למעשה המרחק הממוצע בין כל הצמתים בקלסטר אחד לכל הצמתים בקלסטר השני.

$$d(C_i, C_j) = \frac{1}{|C_i| \, |C_j|} \sum_{p \in C_i} \sum_{q \in C_j} d(p,q)$$

©ענת עציון

דוגמא:

הפעלת UPGMA על מטריצה אולטרמטרית



טענה: הפעלת UPGMA על מטריצה אולטרמטרית תוציא כפלט עץ אולטרמטרי נכון.

(כלומר: האלגוריתם קונסיסטנטי עבור מטריצות אלוטרמטריות).

הערה: UPGMA לא מצליח עבור מטריצות שאינן אולטרמטריות-בונה עת אולטרמטרי לא נכון.

ניתוח סיבוכיות:

כל קלסטר מתוחזק כערימה. ישנן n ערימות שכל אחת שומרת את המרחקים של קלסטר אחד לכל האחרים. ואז באיחוד קלסטרים ניתן לעדכן כל אחד מהמרחקים ב-$O(\log n)$. ולכן עדכון הערימות כולן עבור הקלסטר החדש הינו- $O(n\log n)$ בכל איטרציה. ישנן n איטרציות.

ובסה"כ: $O(n^2 \log n)$.

הערה: יתרון המימוש בערימות הוא שניתן ב-$O(1)$ לבדוק לכל קלסטר מהו הקלסטר הכי קרוב אליו.

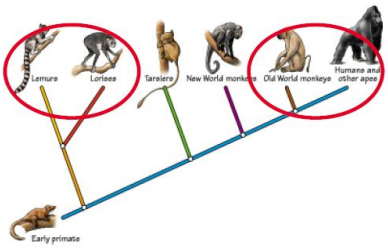
אלגוריתם ה-Nearest Neighbor:

אלגוריתם בעל זמן אופטימאלי: $O(n^2)$.

קלט: מטריצת מרחק D.

Nearest Neighbor (NN): j הוא NN של i אם $j \neq i$ וגם $d(i,j)=\min\{d(i,k)|k\neq i\}$.

Mutual Nearest Neighbors: i,j שהם NN אחד של השני.



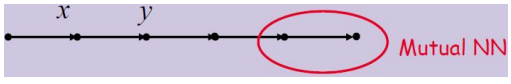
כלומר: $d(i,j)\leq \min\{d(i,k),d(j,k)\mid k\equiv i,j\}$.

האלגוריתם: זהה ל-UPGMA רק שהקלסטרים שיאוחדו בכל איטרציה יהיו mutual nearest neighbors.

הערה: חישוב מרחק הקלסטר החדש מהשאר- באופן זהה.

מימוש האלגוריתם ב-$O(n^2)$:

יוצרים chain מכל קלסטר לקלסטר הכי קרוב אליו ואז עוצרים כאשר מגיעים לזוג שמצביע אחד על השני. החישוב בפעם הראשונה אמנם יכול לקחת- $O(n^2)$, אבל כל ה-chain שהיה לפני שני הקלסטרים, נותר נכון גם לאחר איחודים (כי המרחק לקלסטר החדש מכל קלסטר אחר ב-chain הוא ממוצע מרחקי הקלסטרים שאוחדו ואם אף-אחד מהם לא היה המינימאלי (ה-NN) אז גם המרחק החדש גדול יותר).



©ענת עציון

מטריקת עץ- מרחקים אדטיביים:

מטריצה אדיטיבית: מטריצת מרחק כך שקיים עץ T עם משקלים חיוביים על הקשתות המקיים: לכל i,j , $d(i,j)=d_T(i,j)$ שזהו אורך המרחק מ- i ל- j ב- T.

כלל 4 הנקודות: מטריקה היא אדיטיבית אם"ם לכל 4 מינים קיימת חלוקה לשני זוגות שסכומם קטן מסכום כל הזיווגים האפשריים האחרים (ל-4 מינים יש כאמור 3 דרכים לחלק לזוגות, ואחת הדרכים תיתן סכום קטן או שווה לשתיים האחרות שתהיינה שוות):

$$\boxed{d(i,j)+d(k,l)\le d(i,l)+d(j,k)=d(i,k)+d(j,l)}$$

קלט: מטריצת מרחק D.

פלט: אם D מטריצה אדיטיבית, הפלט הוא עץ התאים למרחקים שהיא מייצגת.

האלגוריתם של Saitou & Nei:

- הקריטריון לבחירת הזוג לאיחוד: i,j שמרחקיהם לשאר הצמתים הוא מקסימאלי וביניהם מינימאלי: $\boxed{Q(i,j)=\sum_r D(r,i)+\sum_r D(r,j)-(n-2)D(i,j)}$ כלומר: i,j הממקסמים ביטוי זה.
- מחברים את i,j לצומת אב חדש- v שהמשקלים על הקשתות הם:

$$w(i,v)=\frac{1}{2}\left(D(i,j)+\frac{r(i)-r(j)}{n-2}\right),\quad w(j,v)=\frac{1}{2}\left(D(i,j)+\frac{r(j)-r(j)}{n-2}\right)$$

- מסירים את i ו- j מהגרף ושמים את v במקומם. הקשת מ- v לכל צומת k אחר בגרף תהיה במשקל: $D(v,k)=\frac{1}{2}\left(D(i,j)+D(j,k)-D(i,j)\right)$.
- חוזרים על התהליך עד אשר נשארים בעץ שני מינים ומחברים אותם בקשת במשקל $D(i,j)$.

טענה: האלגוריתם הוא קונסיסטנטי, רובסטי וסימטרי.

ניתוח סיבוכיות:

בכל איטרציה צריך לחשב את $r(i)=\sum_{k\ne i}D(i,k)$ לכל מין. ואת ערכי Q. זה נעשה ב- $O(n^2)$.

ולכן בסה"כ: $O(n^3)$.

©ענת עציון