BIOINFORMATIKA

FASTA – tekstualni format za zapis poznate sekvence

```
>CP027599.1 Escherichia coli strain 97-3250 chromosome, complete genome
ATCCCGGCCCCGGCAGAACCGACCTATCGTTCTAACGTAAACGTCAAACACACGTTTGATAACTTCGTTG
AAGGTAAATCTAACCAACTGGCGCGCGCGGCGGCTCGCCAGGTGGCGGATAACCCTGGCGGTGCCTATAA
```

Može biti više ovakvih zapisa (sekvenci) u istoj datoteci, samo se stavi novi header i znamo da krećemo na sljedeću sekvencu.

FASTQ – tekstualni format za zapis očitanja

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

Ovaj niz *čudnih znakova* čije ASCII vrijednosti označuju ocjenu Koliko je pouzdano pročitao koju bazu od ovih. ! označava pouzdanost baze G, i tako svaki znak za njegovog para.

ŠTO SE MOŽE RADITI S GOTOVIM OČITANJIMA?

1. Sastavljanje genoma: ako genom do sada nije poznat možemo sastaviti novi genom, tj do sada nepoznatih sekvenci

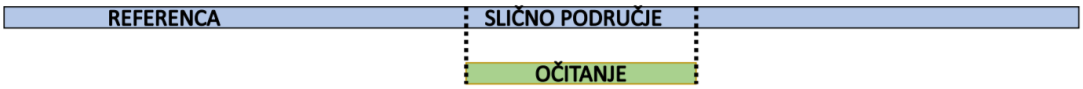
   ➔ određivanje **preklapanja** među očitanjima (vrijedi za 1.)

2. Određivanje varijati: određivanje razlika između sekvenciranog genoma i sličnog gotovog predloška genoma. Ako sekvenciramo genom određene osobe to ne treba onda radit od početka nego uzmemo postojeći ljudski genom (koji je već sastavljen i sekvenciran) i usporedbom tih očitanja s referentnim genomom odrediti gdje se razlikuju i na temelju toga izgraditi genom.
3. Ekspresija gena: ako je sekvenciran RNA koji su geni aktivni u sekvenciranom organizmu, a koji nisu (u organizmu nisu svi geni aktivni, sekvenciranjem RNA se vide se zapravo koji su to aktivni geni jer se iz RNA sekvenciraju proteini), na temelju aktivnih gena se dalje može npr skužit jel netko ima karcinom
4. Metagenomska analiza: sastav uzorka koji ima više organizama; velika primjena u medicinskoj dijagnostici

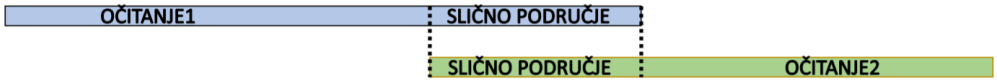
   ➔ **poravnanje** očitanja na jedan(vrijedi za 2. i 3.) ili više(vrijedi za 4.) poznatih referentnih genoma

PORAVNANJE

Poravnanje očitanja na referencu – pronalaženje pozicije na referenci na kojem je referenca „najsličnija”



Preklapanje očitanja – pronalaženje takvih očitanja kod kojih je kraj jednog očitanja „sličan” početku drugog



Levenshtein-ova odaljenost: mjera za udaljenost uređivanja nizova (minimalan broj **zamjena, umetanja i brisanja** potrebnih za transformaciju jednog niza u drugi)

```
GATTACA        -ATTACCC
|||X|||        X|||||XX
GATCACA        GATTAC-A
```

GLOBALNO PORAVNANJE

- Gore je Y, lijevo je X. napiši nizove znakova s lijeva nadesno i odozgo prema dolje
- Napiši redak i stupac zaglavlja tj od 0 do N-1 nadesno i prema dolje
- Za svaki prazan kvadratić gleda se min[(gornji+1), (lijevi+1), (gorelijevi ako su slova X i Y ista odnosno gorelijevi+1 ako su slova X i Y različita] i povuče se strelica prema kvadratiću kojeg smo izabrali iz funkcije min(). To se zove **različitost**, ali može ići i sličnost.
- Ako ima više kvadratića iz koje naš kvadratić može doći, izaberemo bilo koji.

Minimalnu udaljenost nađemo tako da očitamo **najdonji najdesniji broj**.

Put se određuje praćenjem strelica od najdonjeg najdesnijeg kvadratića prema najgornjem najlijevijem kvadratiću možemo dobiti jedan od mogućih poravnanja na način da s desna nalijevo zapisujemo nizove:

- Ako je sljedeći korak **ulijevo** pišemo u Y slovo Yi i u X minus (-)
- Ako je sljedeći korak prema **gore** pišemo u Y minus (-) i u X slovo Xi
- Ako je sljedeći korak prema gorelijevo (u **koso**) pišemo u Y slovo Yi i u X slovo Xi

Globalno poravnanje nam može dat neki rupičast rezultat, nama je bolje *preklapanje*

PREKLAPANJE; POLU-GLOBALNO PORAVNANJE

Preklapanje je poravnanje u kojemu su praznine na početku i na kraju zanemarene (=ne utječu negativno na sličnost).

Promjene u algoritmu:

- Gore je Y, lijevo je X. napiši nizove znakova s lijeva nadesno i odozgo prema dolje
- Napiši redak i, stupac zaglavlja tj od ~~0 do N-1~~ **sve nule** nadesno i prema dolje, jer poravnanje može početi bilo gdje
- Gleda se **sličnost** a ne različitost: **max**[(gornji-1), (lijevi-1), (gornjilijevi+1 ako su slova X i Y ista, gornjilijevi-1 ako su slova X i Y različita)]
- Ako ima više kvadratića iz koje naš kvadratić može doći, izaberemo bilo koji.

Najveću sličnost nalazimo t.d. nađemo **najveći broj u najdesnijem stupcu ili najdonjem redu.** (jer poravnanje može završiti bilo gdje)

Put je od najdesnijeg najdonjeg, kroz *najveću sličnost* do neke gornje „0“ pa samo lijevo do najlijevije najgornje „0“.

Očitavanje je isto:

- Ako je sljedeći korak **ulijevo** pišemo u Y slovo Yi i u X minus (-)
- Ako je sljedeći korak prema **gore** pišemo u Y minus (-) i u X slovo Xi
- Ako je sljedeći korak prema gorelijevo (u **koso**) pišemo u Y slovo Yi i u X slovo Xi

LOKALNO PORAVNANJE

Promjene u algoritmu:

- Gore je Y, lijevo je X. napiši nizove znakova s lijeva nadesno i odozgo prema dolje
- Napiši redak i, stupac zaglavlja **sve nule** nadesno i prema dolje, jer poravnanje može početi bilo gdje
- Gleda se sličnost: **max**[(gornji-2), (lijevi-2), (gornjilijevi+2 ako su slova X i Y ista, gornjilijevi-1 ako su slova X i Y različita)], **ako rezultat poprimi negativnu vrijednost vraćamo ga na 0**
- Ako ima više kvadratića iz koje naš kvadratić može doći, izaberemo bilo koji.

Najveću sličnost nalazimo t.d. nađemo **najveći broj u bilogdje u matrici** (jer želimo odrediti lokalno poravnanje)

Put se određuje od tog maksimalnog po strelicama kojima vodi dok ne dođe do neke „0“ koja ne treba biti u headeru (to provjeriti još!).

Očitavanje je isto:

- Ako je sljedeći korak **ulijevo** pišemo u Y slovo Yi i u X minus (-)
- Ako je sljedeći korak prema **gore** pišemo u Y minus (-) i u X slovo Xi
- Ako je sljedeći korak prema gorelijevo (u **koso**) pišemo u Y slovo Yi i u X slovo Xi

Ako želimo zapisati cijele odsječke (a ne samo slične dijelove) onda nadodamo desno i lijevo preostala slova od X i Y, ali pritom moramo **boldati slične dijelove**, a neslične dijelove ostaviti tankim slovima.

PORAVNANJE VIŠE SEKVENCI

Sekvencranje i signali

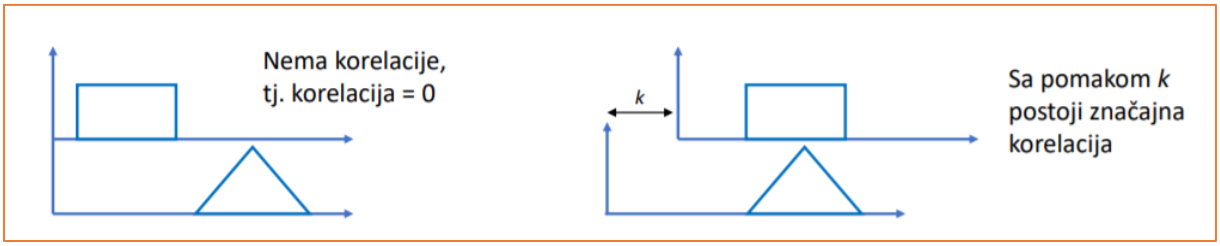
Iz sekvence dobijemo signal na način da vrijednosti nukleotida zamijenimo brojevima, a t smatramo nekim pomakom u prostoru (umjesto kao inače u vremenu)

```
Seq   = GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
X(t)  = G  A  T  T  T  G  G  G  G  T  T  C  A  A  A  G  C  A  G  T  …
X'(t) = 1  2  3  3  3  1  1  1  1  3  3  4  2  2  2  1  4  2  1  3  …
t     = 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20  …
```

MAFFT, Multiple sequence Alignment with Fast Fourier Transform

Promatra sekvencu kao signal, sličnost dvije sekvence opisuje kao korelaciju signala.

Korelacija signala je mjera njihove sličnosti kao funkcija pomaka između njih



Teorem o korelaciji: korelaciju dva signala možemo dobiti tako da izračunamo DFT oba signala (prebacimo ih u frekvencijsku domenu), jedan DFT konjugiramo, pomnožimo ih, te rezultat vratimo u originalnu domenu (vremensku, prostornu …) pomoću IDFT.

Ako pri tome koristimo FFT za sve tri operacije (računamo DFT oba signala te za rezultat računamo IDFT), složenost postupka pada s O(N^2) u O(NlogN)

Računanje korelacije: imamo zadane signale **a** i **b**, te pomak **k.**

- Napišemo signale **a** i **b** jedan ispod drugog pa **b** pomaknemo udesno za **k** mjesta. (ako je k<0 pomičemo se ulijevo). Na preostala mjesta nadodamo nule tako da **a** i **b** imaju jednako znamenaka
- Za svaku znamenku *i* množimo **a***i* * **b***i* i dodajemo je rezultatu Cor(k)

```
k = -3
a = [0 0 0 1 2 3 4]
b = [4 3 2 1 0 0 0]
Cor(-3) = 1
```

```
k = -2
a = [0 0 1 2 3 4]
b = [4 3 2 1 0 0]
Cor(-3) = 2+2 = 4
```

```
k = -1
a = [0 1 2 3 4]
b = [4 3 2 1 0]
Cor(-3) = 3+4+3 = 10
```

```
k = 0
a = [1 2 3 4]
b = [4 3 2 1]
Cor(-3) = 4+6+6+4 = 20
```

Vrijednost **k** je element od [-len(**b**)+1, len(**a**)]
broj nula koji dodajemo signalu **b** je (len(**a**)-1) i obrnuto za signal **a**.