Biostatistică și Bioinformatică.

Lab 5: Aliniere bazată pe metode euristice (BLAST), alinierea multiplă a secvențelor si arbori filogenetici

---

1. Analiza similarității dintre secvențe folosind algoritmi de tip BLAST

Algoritmul BLAST stă la baza majorității sistemelor de interogare a bazelor de date biologice. Există variante adaptate pentru diferite tipuri de secvențe (nucleotide sau aminoacizi). Datele de intrare pentru un algoritm de tip BLAST sunt:

- Secvența de interogat
- Baza de date sau setul de secvențe unde se realizează căutarea
- Parametrii specifici algoritmului:
    - lungimea șablonului de start (ex: 11 pentru secvențe de nucleotide și 3 pentru secvențe de aminoacizi)
    - scor potrivire (ex: 1), scor nepotrivire (ex:-2)
    - penalizarea pentru inițierea unui gap (ex: 5)
    - penalizarea pentru extinderea unui gap (ex: 2)
    - pragul pentru E-value (valoarea sub care se consideră că similaritatea e semnificativă)
    - scoruri pentru potrivire/nepotrivire (în cazul secvențelor ADN) respectiv matricea de substituție (în cazul secvențelor de aminoacizi)

In marea majoritate a implementărilor se returnează alinierile pentru care valoarea indicatorului E este mai mică decât pragul specificat. Pentru fiecare aliniere se furnizeaza valorile pentru: E-value și bit-Score. Potrivirea este considerată cu atât mai semnificativă cu cât E-value este mai mică (în general valori mai mici decât 0.01 indică o potrivire semnificativă) și bit-Score este mai mare.

Cea mai simplă modalitate de a testa aplicațiile de tip BLAST este utilizarea implementărilor accessibile la http://blast.ncbi.nlm.nih.gov/Blast.cgi

Exercițiu 1. Utilizând varianta adecvată de BLAST de la http://blast.ncbi.nlm.nih.gov/Blast.cgi să se caute in genomul uman (human) secvența aflată în fișierul mitochondrion.fa. Să se realizeze același lucru pentru genomul șoarecelui (mouse) și pentru cel al musculiței de oțet (drosophila melanogaster). Să se analizeze influența lungimii cuvântului de start prin setarea optiunii –W (de exemplu –W 15).

O altă modalitate de a accesa resursele de la NCBI este de a utiliza facilitățile de tip eUtility (API pentru NCBI Entrez). Un exemplu de căutare, bazat pe efetch, în baza de secvențe utilizând BLASTP (variantă de BLAST pentru proteine) este ilustrat în secvența R următoare. Detalii despre NCBI eUtility pot fi găsite la http://www.ncbi.nlm.nih.gov/books/NBK25500/.

Funcțiile R care care permit accesarea programatică a serviciilor Web oferite de către NCBI se află în pachetul RCurl care trebuie încărcat în prealabil.

- Descărcarea unei secvențe specificată printr-un identificator din baza de date “protein” (prin utilizarea e-utilitarului efetch din NCBI):

```
id = "15718680"
seq=getForm("http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi",
db="protein", id=id, retmode="text", rettype="fasta", binary=F)
s=strsplit(seq[1],"\\n")[[1]]
seq=AAString(paste(s[2:length(s)],collapse=""))
```

Exemplu de utilizare BLASTN prin apelare din R:

- Definirea unei funcții de căutare în baza de date “nr” (non-redundant GenBank) care utilizează BLASTN pentru a returna secvențele similare cu secvența specificată ca secevență de interogare

```
blastSeqKK <- function (x, database = "nr", hitListSize = "10",
                        filter = "L", expect = "10", program = "blastn",
                        attempts = 10) {
  baseUrl <- "http://www.ncbi.nlm.nih.gov/blast/Blast.cgi"
  query <- paste("QUERY=", as.character(x), "&DATABASE=", database,
                 "&HITLIST_SIZE=", hitListSize, "&FILTER=", filter, "&EXPECT=",
                 expect, "&PROGRAM=", program, sep = "")
  url0 <- sprintf("%s?%s&CMD=Put", baseUrl, query)
  results <- tempfile()
  Sys.sleep(5)
  require(XML)
  post <- htmlTreeParse(url0, useInternalNodes = TRUE)
  x <- post[["string(//comment()[contains(., \"QBlastInfoBegin\")])"]]
  rid <- sub(".*RID = ([[:alnum:]]+).*", "\\1", x)
  rtoe <- as.integer(sub(".*RTOE = ([[:digit:]]+).*", "\\1",
                         x))
  url1 <- sprintf("%s?RID=%s&FORMAT_TYPE=XML&CMD=Get", baseUrl,
                  rid)
  Sys.sleep(rtoe)
  .tryParseResult <- function(url, attempts){
    for (i in 1:(attempts+1)) {
      result <- tryCatch({
        xmlTreeParse(url, useInternalNodes=TRUE,
                     error = xmlErrorCumulator(immediate=FALSE))
      }, error=function(err) NULL)
      if (!is.null(result)) return(result)
      Sys.sleep(10)
    }
    stop(paste("no results after ", attempts,
               " attempts; please try again later", sep = ""))
  }
  result <- .tryParseResult(url1, attempts)
  return(result)
}
```

Apelul funcției definite anterior

```
x = read.fasta("ex1.fasta")
seqCollapse = paste(toupper(as.character(x)), collapse = "")
blastRes = blastSeqKK(x = seqCollapse, database = "nr", hitListSize = 3,
attempts = 20)
```

2. Alinierea multiplă

Alinierea multiplă se referă la alinierea globală a mai multor secvențe cu scopul detectării unor regiuni conservate de-a lungul evoluției, identificării unor grupări de proteine similare sau construirii arborilor filogenetici.

Deși din punct de vedere teoretic tehnica programării dinamice (care stă la baza algoritmilor Needleman-Wunsch și Smith-Waterman) poate fi extinsă la cazul alinierii a mai mult de două secvențe, din punct de vedere practic tehnica nu este eficientă în cazul a mai mult de 3 secvențe. In cazul mai multor secvențe se preferă tehnici aproximative. Dintre acestea cele mai frecvent folosite sunt cele de tip progresiv (de exemplu ClustalW sau TCoffee). Ideea principală a acestor algoritmi este de a parcurge următoarele etape:

- Se realizează alinierea tuturor perechilor de secvențe (folosind un algoritm de aliniere globală – de exemplu algoritmul Needleman-Wunsch)
- Se construiește o matrice de similaritate (sau o matrice de distanțe): elementul de pe linia i coloana j este raportul dintre numărul de potriviri din alinierea secvențelor i și j și lungimea alinierii. Cu cât secvențele i și j sunt mai similare cu atât va fi mai mare valoarea corespunzătoare din matrice (în matricea de distanțe cu cât disimilaritatea e mai mare cu atât mai mare va fi valoarea din matrice).
- Se construiește un arbore de ghidare într-o manieră similară construirii dendrogramelor pe baza algoritmilor ierarhici aglomerativi (vezi lab 6).
- Se realizează alinierea succesivă a câte două secvențe (fie secvențe inițiale fie secvențe de tip consens obținute printr-o aliniere anterioară) în ordinea sugerată de arborele de ghidare (se aliniază prima dată cele mai similare secvențe)

2.1. Instrumente software pentru alinierea multiplă a secvențelor

Alinerea multiplă poate fi realizată folosind implementări ale diferiților algoritmi progresivi de aliniere. Cele mai cunoscute și utilizate implementări sunt:

- Suita Clustal (www.clustal.org) conține implementări ale algoritmului Clustal (ClustalW – varianta clasică, ClustalX – varianta cu interfață grafică, ClustalOmega (varianta eficientă care folosește aliniere bazată pe modele de tip HMM – Hidden Markov Models = modele Markov cu stări ascunse). Poate fi utilizată în diferite moduri:
  - Ca serviciu web (ex: http://www.ebi.ac.uk/Tools/msa/clustalw2/) – dimensiunea maximă a fișierului ce conține secvențele de aliniat este 1Mb
  - Ca aplicație caracterizată prin interfață text (ClustalW) sau prin interfață grafică (ambele variante pot fi descărcate de la http://www.clustal.org/download/current/) - versiuni pentru Unix si Windows
  - Prin apel din alte aplicații (ex: R)

- Muscle (http://www.drive5.com/muscle/). Poate fi utilizat ca aplicație Unix, Windows. Există pachet R (muscle)

- TCoffee (http://www.tcoffee.org/). Poate fi utilizat ca:
  - Serviciu web (http://tcoffee.crg.cat/) – rezultatul alinierii este trimis prin e-mail. Numărul de secvențe care pot fi aliniate este 150.
  - Aplicație Unix

Exercițiu 2:
(a) Să se alinieze secvențele de aminoacizi stocate în fișierul multiseq.fasta utilizând serviciile web bazate pe Clustal și TCoffee
Indicatie: Se folosesc serviciile web de la http://www.ebi.ac.uk/Tools/msa/ și http://tcoffee.crg.cat/

(b) Să se alinieze secvențele de nucleotide din fișierul exMSA.fasta utilizând pachetul R muscle (face parte din Bioconductor; necesită R 3.2.0).

Indicație:
Instalarea și încărcarea prealabilă a pachetului Muscle:

```
>source("http://bioconductor.org/biocLite.R")
> biocLite("muscle")
```

Construirea unui set de secvențe ADN pornind de la tablou cu șiruri:

```
>seqs=DNAStringSet(c("GATCA"," GATATTA","GTTGA"," GTCAGCTCA"))
```

Construirea alinierii:

```
>aliniere=muscle(seqs)
```

3. Analiza filogenetică

Analiza filogenetică permite identificarea relațiilor de evoluție între diferite specii (reprezentate prin secvențe ADN sau de aminoacizi). Rezultatul analizei este o diagramă ierarhică (similară dendrogramelor utilizate în gruparea bazată pe algoritmi ierarhici) numita *arbore filogenetic*. Ramurile în arbore exprimă descendența sau ascendența în procesul de evoluție. De regulă speciile analizate sunt asociate cu noduri frunză ale arborelui iar nodurile interne corespund unor specii ipotetice care ar reprezenta strămoși comuni.

Principalele etape ale analizei filogenetice sunt:

- Alegerea secvențelor pe baza cărora se face analiza (biologii folosesc secvențe prezente la toate speciile – de exemplu ARN ribosomal sau gene mitocondriale).
- Stabilirea similarităților (sau a distanțelor) dintre secvențe prin aliniere (simplă sau multiplă).
- Construirea propriu-zisă a arborelui filogenetic. Există mai multe categorii de metode:
  - Metode bazate pe distanțe – necesită determinarea similarității/distanței între oricare două secvențe
  - Metode bazate pe principiul maximizării simplității (maximum parsimony)
  - Metode bazate pe principiul verosimilității maxime

3.1. Construirea unui arbore filogenetic folosind metode bazate pe distanțe

Metodele bazate pe distanțe presupun că pentru fiecare pereche de secvențe se poate calcula distanța dintre ele.

3.1.1. **Algoritmul UPGMA** (Unweighted Pair Group Method using arithmetic Averages)

UPGMA se bazează pe aceeași idee ca și algoritmii aglomerativi de clustering. Secvențele (speciile) vor corespunde nodurilor frunză. Fiecare nod are asociat un nivel (înălțime) și fiecare muchie are asociată o etichetă (eticheta muchiei care unește nodul i cu nodul j este diferența dintre înălțimile corespunzătoare celor două noduri).

Date de intrare: set de secvente (x_1, x_2,…)
Ieșire: arbore cu rădăcină în care toate nodurile frunză se află pe același nivel (similar unei dendrograme)

Structura algoritm:

Inițializare:

- Construiește lista inițială cu noduri (fiecare nod corespunde unei secvențe x_i care definește clusterul propriu C_i) . Acestea sunt nodurile frunză ale arborelului. Pentru fiecare nod se asignează înălțimea: $h(C_i)=0$ și se reține numărul de secvențe care fac parte din clusterul corespunzător nodului (la inițializare acesta este 1)
- Construiește matricea de distanțe (aceasta poate fi reprezentată ca o listă – pentru fiecare element din listă se specifică perechea de noduri corespunzătoare și valoarea distanței)

Prelucrare repetitivă:

- Determină cele mai apropiate două noduri (corespunzătoare clusterilor C_i *și* C_j) adică perechea de noduri pentru care distanța dintre ele (d_{ij}) este minimă
- Grupează nodurile i și j și construiește nodul k ce corespunde reuniunii dintre clusterele corespunzătoare nodurilor i și j ($C_k = C_i \cup C_j$). Nodul k va avea înălțimea $h(C_k)= d_{ij}/2$; etichetează muchia (C_k,C_i) cu $h(C_k)$-$h(C_i)$ și muchia (C_k,C_j) cu $h(C_k)$-$h(C_j)$
- Actualizează lista cu noduri prin marcarea nodurilor C_i *și* C_j ca inactive și adăugarea lui C_k ca nou nod activ
- Calculează distanța de la C_k la nodurile active și actualizează matricea (lista) cu distanțe

Condiție de terminare:

- Când se ajunge la un singur nod activ (corespunde clusterului ce conține toate secvențele)

3.1.2. **Algoritmul Neighbor-Joining**

Are ca scop tot construirea unui arbore într-o manieră similară algoritmilor aglomerativi de grupare, însă se urmărește minimizarea sumei lungimilor tuturor muchiilor ce unesc nodurile. Din acest motiv la alegerea nodurilor care se unesc se determină perechea (i,j) pentru care distanța dintre ele este minimă, iar distanța medie față de celelalte noduri este maximă. Prin urmare se alege perechea (i,j) care minimizează o valoare ajustată a distanței:

$$D_{ij} = d_{ij} - (r_i + r_j)$$

$$r_i = \frac{1}{N-2}\sum_{k \in L} d_{ik}$$

Ajustarea distanței se bazează pe scăderea din distanța initială dintre nodurile i și j a sumei distanțelor medii dintre nodul i (respectiv j) și celelalte (N-2) noduri.

3.2. Analiza filogenetică în R

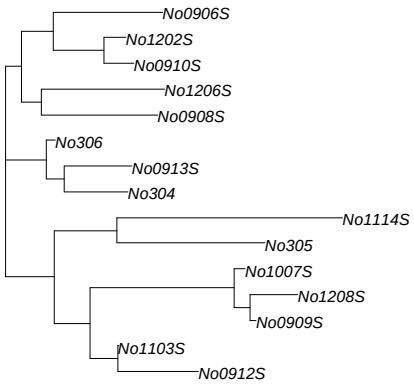
Pachetul ape (necesită instalare și încărcare) conține o serie de funcții care permit construirea și vizualizarea arborilor filogenetici. Printre funcțiile implementate se află și cea corespunzătoare algoritmului Neighbor-Joining: nj care se apelează prin: nj(matriceDistante) și returnează un obiect din clasa phylo.

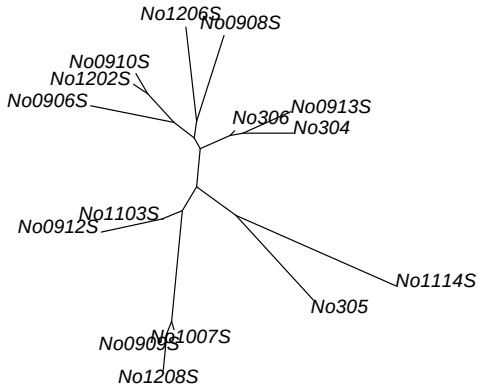
Matricea de distanțe se poate construi explicit pornind de la secvențe ADN sau folosind funcția dist.dna.

Arborii filogenetici construiți prin nj pot fi vizualizați prin funcția plot. Un apel de forma plot(arbore) permite vizualizarea unui arbore cu rădăcină, iar un apel de forma plot(arbore, "u") vizualizează un arbore fără rădăcină.

Exercițiul 1.

```
# incarcarea unui set de date (predefinit in pachetul ape)
data(woodmouse)
# construirea arborelui filogenetic folosind algoritmul Neighbor Joining
arbore =  nj(dist.dna(woodmouse))
# vizualizarea arborelui filogenetic (cu rădăcină)
plot(arbore)
# vizualizarea arborelui filogenetic (fără rădăcină)
 plot(arbore, "u")
```





(a) Arbore filogenetic cu rădăcină (b) Arbore filogenetic fără rădăcină

Teme (la alegere).

Varianta 1. Să se implementeze o variantă simplificată de algoritm euristic de căutare caracterizată prin:

- primește ca date de intrare două secvențe de aminoacizi (una scurtă q – max 20 aminoacizi- si una mai lunga s – cca 200 aminoacizi); secvența scurtă reprezintă șablonul de interogare
- pentru fiecare subsecvență de câte k=3 aminoacizi din secvența de interogare q se identifică prezențele (potriviri exacte) în secvența de referință s și pentru fiecare dintre aceste prezente se realizează extinderea (în stânga, în dreapta, alternativ în ambele direcții) cât timp scorul potrivirii este mai mare decât un prag T>0 (pentru calculul scorului se utilizează matricea Blosum 62). Pragul T poate fi controlat de către utilizator.
- dintre toate extinderile cu scor mai mare decât T se afișează cele cu scorul maxim.

Obs. Extinderile nu conțin gap-uri

Varianta 2. Să se implementeze algoritmul star-center pentru aliniere multiplă de secvențe ADN (algoritmul este prezentat în cursul 8).