

SVAR

## Løsningsforslag til eksamensoppgaver i KJB492 Bioinformatikk, høsten 2002

### Oppgave 1

Programmene FASTA og BLAST er eksempler på bruk av såkalte heuristiske metoder.

a) Hva vil det si at en metode er heuristisk? Hvorfor bruker man heuristiske metoder?

*En heuristisk metode er en tilnæringsmetode som ikke nødvendigvis finner den optimale løsningen på et problem, men som finner en løsning som regel er god nok i praksis. Man bruker heuristiske metoder fordi det er umulig eller vanskelig å bruke en metode som finner den optimale løsningen, ofte fordi problemet er så vanskelig å løse at det ville ha tatt alt for lang tid. F.eks. bruker man sjelden en optimal lokal sammenligningsmetode (Smith-Waterman) når man søker i store databaser, men heller en heuristisk metode slik som BLAST.*

I det første trinnet i søkeprosessen velger FASTA og BLAST ut små grupper av symboler basert på søkesekvensen som de leter etter i databasesekvensene.

b) Forklar hvordan FASTA og BLAST velger ut disse symbolgruppene og påpek forskjellene mellom disse to programmene. Forklar betydningen av parametrene/opsjonene hos FASTA og BLAST som styrer utvalget av symbolgruppene. Forklar også hvordan hastigheten og sensitiviteten til programmene påvirkes når disse parametrene/opsjonene endres.

*FASTA sammenligner grupper av 1-2 påfølgende aminosyrer eller 1-6 påfølgende nukleotider. Gruppen av nukleotider fra søkesekvensen og gruppen av nukleotider fra databasesekvensen må være helt identisk for at det regnes som et treff. Det er parameteren  $k$  (også kalt  $ktup$ ) som bestemmer hvor mange symboler som inngår i gruppene. Dersom  $k$  er liten (f.eks. 1) vil FASTA være mest sensitiv, men også bruke lengst tid. Dersom  $k$  er stor vil FASTA være mindre sensitiv, men til gjengjeld gå raskere.*

*Når BLAST brukes til å sammenligne aminosyresekvenser vil programmet bruke grupper (ord, words) som består av normalt 3 påfølgende aminosyrer. Gruppen av aminosyrer fra søkesekvensen og gruppen av aminosyrer fra databasesekvensen må ikke nødvendigvis være helt identiske for at det skal regnes som et treff, men de må ha en viss likhet. Gruppene sammenlignes og det beregnes en score som må være større eller lik en viss grenseverdi (f.eks. 11). Scoren beregnes ved å summere verdier som hentes fra en substitusjonsmatrise som indekseres med aminosyrene i gruppene som sammenlignes. Parameteren  $W$  styrer lengden av ordene, mens parameteren  $T$  styrer grenseverdien. Når  $T$  økes mens  $W$  holdes konstant vil BLAST vanligvis bli mindre sensitiv, men raskere. Omvendt dersom  $T$  gjøres mindre. Når  $W$  økes mens  $T$  holdes konstant vil BLAST vanligvis bli mer sensitiv, men tregere. Omvendt dersom  $W$  gjøres mindre.*

*Når BLAST brukes til å sammenligne nukleotidsekvenser vil programmet bruke grupper (ord, words) som består av normalt 11 påfølgende nukleotider. Gruppen av nukleotider fra søkesekvensen og gruppen av nukleotider fra databasesekvensen må være helt identiske for at det regnes som et treff. Parameteren  $W$  styrer størrelsen på gruppen (lengden av ordene). Når  $W$  økes vil BLAST vanligvis bli mindre sensitiv, men raskere. Omvendt dersom  $W$  gjøres mindre.*

*Hovedforskjellen mellom FASTA og BLAST på dette punktet finner man i måten man velger ut grupper av aminosyrer. FASTA krever altså at aminosyrene i søkesekvensen er helt identiske med dem i databasesekvensen, mens BLAST bare krever at det er en viss likhet mellom gruppene totalt sett. For nukleotidsekvenser er hovedforskjellen at størrelsen på symbolgruppene vanligvis er større hos BLAST enn hos FASTA.*

c) Søkeprogrammene beregner som regel en såkalt expect-verdi (E-verdi) for sekvensene i databasen som ligner på søkesekvensen. Hva uttrykker denne verdien?

*E-verdien uttrykker det forventede antall treff i databasen med en sekvenssammenligning som har en score som er lik eller høyere (bedre) enn det aktuelle treffet.*

d) Dersom man får et treff med  $E=100$ , er det vanligvis et statistisk signifikant treff? Hva med  $E=0.00001$ ?

*Nei, et treff med  $E=100$  er vanligvis ikke statistisk signifikant. Et treff med  $E=0.00001$  er vanligvis statistisk signifikant. (Fra et biologisk synspunkt kan det imidlertid være anderledes hva som er signifikant.)*

e) Man gjør et søk med BLAST og får et treff som har  $E=0.01$ . Ett år senere gjør man et nytt søk med samme søkesekvens og samme program og får treff på samme sekvens som forrige gang, men denne gangen har E-verdien økt til 0.02. Er databasen blitt større eller mindre? Ca hvor mye større eller mindre?

*Databasen er nok blitt omtrent dobbelt så stor.*

f) Forklar i grove trekk hvordan en iterativ søkemetode (slik som PSI-BLAST) fungerer og hvordan den skiller seg fra en ordinær søkemetode (for eksempel BLAST). Hva er fordelene med å bruke en iterativ søkemetode?

*En iterativ søkemetode foretar flere omganger med søk. Den starter vanligvis med en ordinær søkesekvens og gjør et søk med denne. Resultatene prosesseres og danner grunnlaget for en ny søkeomgang. For hver runde forfines inndata litt basert på resultatene fra forrige runde. (F.eks. lager PSI-BLAST en sekvensprofil basert på en flersekvensoppstilling av treffene med en E-verdi under en viss grense. Denne sekvensprofilen brukes som inndata til neste runde.) Prosessen stoppes etter et visst*

*antall omganger, eller når resultatene forandres lite eller ingenting mellom to omganger.*

*Fordelen med en iterativ metode er at den ofte er mer sensitiv enn en ordinær søkemetode.*

## **Oppgave 2**

Vi har et nytt protein som vi kun kjenner aminosyresekvensen til. Vi vil bruke varianter av BLAST til å finne ut mer om proteinet. Vi ønsker å finne ut mer om følgende egenskaper ved proteinet:

- a) funksjonen til proteinet
- b) tredimensjonal molekylstruktur for proteinet
- c) genstrukturen (plassering av introner og eksoner) til genet som koder for proteinet
- d) mRNA-sekvensen som koder for proteinet

Hvilke databaser og hvilke varianter av BLAST bør vi bruke i hvert tilfelle?

*a) Vi bør bruke blastp til å søke i en proteindatabase, f.eks. SWISSPROT eller nr (non-redundant protein database). Da er det mulig at vi finner sekvenslikhet med et protein med kjent funksjon, og derved kan anta at vårt protein kan en lignede funksjon.*

*b) Vi bør bruke blastp til å søke i PDB-databasen som inneholder proteiner med kjent tredimensjonal struktur. Hvis vi finner sekvenslikhet med et protein i denne databasen er det mulig at den tredimensjonale strukturen for deler av eller hele proteinet vårt ligner på tilsvarende deler av proteinet i PDB.*

*c) Vi bør bruke tblastn til å søke i en sekvensdatabase med genomiske sekvenser (f.eks. deler av GenBank eller EMBL). Programmet tblastn vil translaterer databasesekvensene i alle 6 mulige leserammer og deretter sammenligne med vårt protein. Dersom vi finner flere treff med lange strekk med fullstendig sekvenslikhet vil vi kunne finne hvor de proteinkodende delene av eksonene er plassert.*

*d) Her bør vi bruke tblastn til å søke i en sekvensdatabase med mRNA-sekvenser (f.eks. deler av GenBank eller EMBL). EST-databaser kan også være aktuelle, men vil ofte ikke gi hele mRNA-sekvensen.*

**Forslag til momenter som bør inngå i besvarelse av oppgave gitt til KJB492 eksamen H2002:**

“Tenkt deg at du har sekvensert et gen (som er omlag 2000 baser langt) fra 100 arter og er interessert i å beregne fylogenen til disse sekvensene. Hvilke fylogenetiske metoder ville du ha brukt for å beregne den mest optimale fylogenen? Beskriv fordeler og ulemper med de ulike metodene du mener er egnet å bruke på dine data.”

**Hovedpunkter som studenten bør ha inkludert i besvarelsen er:**

- Modeltest for å finne den mest optimale evolusjonsmodellen (Program som kan brukes: Modelltest). Dette gjøres ved en såkalt hierarkisk test av alle vanlige (og mye brukte) evolusjonsmodeller, hvor likelihood verdien til hvert enkel

modell blir beregnet. Deretter gjøres *likelihood ratio test* som forteller hvilken modell som inneholder færrest parametre men som samtidig er signifikant bedre enn andre mindre parameterrike modeller. I og med at dette er en maximum likelihood metode, vil likelihooden fortelle hvor sannsynlig det er for å produsere sekvensalignementet gitt modellen, og er altså motsatt av andre fylogeni-metoder (\*). I denne modellen inngår to komponenter som må være tilstede for å kunne gjøre denne type likelihood beregninger: et *tree* som viser fylogeni til sekvensene og selve *evolusjonsmodellen* (inkl. Substitusjonsrater, basefrekvenser og rate distribusjonen).

- Den mest optimale modellen inneholder en del elementer, hvor de viktigste er *basefrekvenser*, *substitusjonsmatrise* og *rate distribusjon* mellom sites (gamma korreksjon, hvorav alpha ( $\alpha$ ) bestemmer gamma-fordelingskurven.) Av alle parametrene er gamma korreksjon den som gir høyest forskjell i likelihood score.
- Det er mange former for substitusjonmodeller, men de viktigste som har vært gjennomgått er Jukes & Cantor, Felsenstein 81, HKY og GTR (“General time reversible” og er en 6 parameter modell). Felles for all er at de antar en *Markov Chain*, dvs at alle substitusjonene er uavhengig av tidligere substitusjoner i samme site og uavhengig av substitusjoner i andre sites. De er også “*time reversible*” dvs at substitusjonen  $A > C = C > A$  etc. Hoved-forskjellen mellom modellene er at de antar at alle basefrekvensene og substitusjonratene er like eller de kan anta at de er ulike (\*).
- Den mest optimale maximum likelihood modellen som foreslås fra en modelltest kan implementeres i både maximum likelihood og distanse metoder.
- *Maximum likelihood (ML)* er den mest robuste metoden som brukes idag, men pga. det store alignmentet i denne oppgaven vil det i praksis være umulig å gjøre en slik analyse (selv om man har tilgjengelig all tungregnekapasiteten på UiO....).
- Derfor er en *distanse analyse* det mest fornuftige (dvs. ML-distanse).
- Andre distanse-metoder som ikke inngår i modelltester er *LogDet* som er lite påvirket av ulik nukleotide (eller AA) komposisjon mellom sekvensene.
- I tillegg kan *parsimony*-metoder brukes.
- Det er relativt viktig at alle metoder gir *kongruente svar*. Om dette ikke er tilfelle, kan det skyldes svakheter i enkelte av metodene, eller at dataene man bruker ikke inneholder nok *fylogenetisk informasjon*.
- Det finnes i hovedsak to måter å søke etter fylogenetisk trær: Enten å bruke en *optimalitetskriterie* (ML, Minimum evolution eller Parsimony) eller *cluster-algoritmer* (feks. Neighbor joining).
- Fordelene ved å bruke optimalitetskriterie er at man kan rangere trær etter hvor gode de er (dvs. hvor optimale de er ifølge optimalitetskriteriet). Dessuten vil det være mulig å gjøre “*branch-swapping*” (dvs flytting av grener) etter at et første-tre er laget. En slik topologi-endring gjør det mulig å optimalisere tre-topologien ytterligere. Fordelen med cluster algoritmer er at de kan gjøre utregninger veldig mye fortere enn de opt. baserte metodene.
- Den ultimate metoden å bruke i de fleste sammenhenger er ML. Grunnen er at den tar hensyn til ulikheter i *grenlengder*, dvs at metoden tar hensyn til at substitusjoner er mer sannsynlige i noen grener enn i andre og dermed kan kompensere for dette. Parsimony vil har ingen slike korreksjoner innebygd, og kan da (på grunn av *homoplasier*) foretrekke feil tre, hvor lange greiner plasseres sammen (long branch attraction). Distanse metoder har den ulempen at

de omgjør alle karakterene i dataene om til *distansematrise*. I denne transformasjonen mistes viktig fylogenetisk informasjon. Imidlertid er distanse metoder raskere og kan implementere substisjonsmodeller (inkl. gamma korreksjon) som inneholder korreksjoner for ulike rate mellom sites – som er en viktig forbedring i forhold til parsimony. Ulempen med ML er at den er veldig regnekrevende -som gjør det vanskelig å bruke den i mange tilfeller.

- Alle ML-, distanse- og parsimonymetoder forutsetter at alle sekvenser i alignmentet inneholder samme basefrekvenser. Dette imidlertid ikke tilfelle i mange situasjoner, og kan føre til at fylogien blir inkonsistent. En måte å løse dette på er å gjøre en LogDet analyse. Problemet med LogDet er at den ikke har innebygd gamma korreksjon – hvis det er store forskjeller i substisusjon mellom sites, vil LogDet også bli inkonsistent.
- Alle modellene i forrige punkt antar at alle sites evoluerer uavhengig av hverandre. Det er heller ikke tilfelle i alle situasjoner (antagelig aldri!). Det fører til at evolusjonshastigheten kan bli forskjellig i de ulike sekvensene. For å ta hensyn til slik evolusjon, må man bruke en *covariotide* (\*\*) metode som tillater at evolusjonsraten varierer mellom sekvensene.

Dette er bare hovedmomentene som kan inngå i en slik besvarelse, men det finnes en mengde andre saker som kan tas opp i en slik diskusjon.

\* Jeg antar at dette er kjent stoff og utbroderer ikke dette nærmere her.

\*\* Covariotide anser jeg for å være et “D moment” som er ikke er gjennomgått nøye i forelesninger.

Håper dette er en hjelp til å rette eksamensoppgavene,

Vennlig hilsen Kamran

## Oppgave 6

- (a) Struktur er bedre konservert enn sekvens. To proteiner kan ha lignende struktur mens man ikke kan oppdage sekvenslikhet. Dette betyr at struktursammenligninger kan synliggjøre likheter mellom proteiner som man ikke klarer å oppdage ved hjelp av sekvenssammenligninger.
- (b) Sekvens alignment er av avgjørende betydning. Ved en likhet så lav som 35 % vil det å lage en alignment være vanskelig, særlig hvis man har mange delesjoner og insersjoner (de som nevner dette siste bør belønnes). Man vil stadig være i tvil om man jobber med en ”riktig” alignment. Man kan forbedre sitt alignment ved å bruke en multiple sekvens alignment, istedenfor en alignment av bare sitt protein og templatproteinet (Dette siste punktet bør telle rundt 20 %)
- (c) Ramachandran: plot phi og psi kombinasjoner for hvert aminosyre i et protein. Man kan så se om man har ”outliers” som ikke er tillatt (Nb. glysin kan ligge overalt og vil derfor dukke opp i ”outlier-positions”; Asn har også utvidede muligheter i forhold til de andre aminosyrene). Generelt sier dette lite om modellbyggingsprosedyren fordi prosedyren vanligvis ikke innebærer endringer av hovedkjeden; så de feil man har er feil som fantes i templatstrukturen.

Her kan man gi bonus til de som sier at Ramachandran plott kan avsløre feil rundt modellerte delesjoner/insersjoner. Ramachandran plott kan også avsløre feil på ikke-konserverte posisjoner: f eks: modellen kan ha proliner på posisjoner hvor proliner ikke kan forekomme (feil phi-vinkel).

- (d) Det finnes flere databaser som inneholder tredimensjonale modeller for alle proteiner som det går an å bygge modell av.
- (e) PHD bruker et multiple sekvens alignment som basis og det betyr at PHD, i motsetning til CF, tar hensyn til global informasjon om struktur. Dvs at PhD tar hensyn til konserverte mønstre, for eksempel hydrofob-polar-hydrofob-polar-..... for beta-strand og lignende mønstre (polar på i, i+3, i+4, i+7.....) for heliks. I tillegg bruker PHD neural networks (men det er ikke dette jeg er ute etter; det handler om multiple alignments)
- (f) "Threading" eller "fold recognition". Her tar man sekvensen og ser hvor kompatibel den er med hver eksisterende proteinstruktur. Jeg ville være fornøyd med dette korte svaret. Man kan tilføre noe som: Til dette omdanner man alle 3D strukturer til sekvensprofiler, og så aligner man sin sekvens med alle disse profiler.