CAPÍTULO 1

Alineamiento de ADN

Introducción

Aunque los datos moleculares pueden proporcionar grandes cantidades de caracteres, presentan retos en la asignación de homologías al tratarse exactamente de las mismas bases en todas las secuencias; además en muchas ocasiones las secuencias tienen un tamaño desigual. Para resolver estos problemas se han ideado los métodos de alineamiento, que buscan recuperar las posibles homologías dentro de diferentes secuencias, al utilizar una matriz de transformaciones, entre las cuatro bases y los gaps o InDels (inserción/eliminación); así, colocando gaps las dos secuencias son equivalentes en longitud y posiciones para las bases; por ejemplo, compare la secuencia 1x con cualquiera de los alineamientos 2. [1]

```
1x: a c t T C C g A A T T T g g c t a c T T C C g A A t T T g G c
2x: a c t C G A T T G C C T A C T C G A T T G C C
2a: a c t - - C g A - - T T g c c t a c T - C - g A - t T - g C c
2b: a c t - C - g A - T - T g c c t a c - T - C g A - t - T g C c
2c: a c t - C - g - A T T - g c c t a c - T - C g - A t - T g C c
```

Ya que se necesitan más de tres secuencias para un análisis filogenético, es necesario el alineamiento simultáneo de múltiples secuencias. El primer método usado es el alineamiento múltiple que usa un árbol **guía** con las secuencias en los terminales; a partir de este árbol se hace el primer alineamiento para el par más cercano[2], luego optimiza los gaps, posteriormente se alinea con la siguiente terminal que sea la más cercana y el proceso se repite hasta llegar a la base; la idea es parecida a la optimización en un análisis filogenético.
Una nueva idea, usada principalmente por cladistas, es la optimización directa (Wheeler, 1996), donde el objetivo no es construir el alineamiento, sino directamente el cladograma; en este caso, se hacen los alineamientos locales (como en el alineamiento múltiple) y los gaps son considerados transformaciones, es decir se colocan temporalmente en los nodos. Wheeler (1996) argumenta que implementa una idea de homología dinámica acorde con el análisis filogenético. En la optimización de estados fijos (Wheeler, 1999) se usa la secuencia completa, y la matriz de transformaciones sólo sirve para asignar costos, pero no estados en el nodo.

[1] Ejemplo modificado de Siddall, http://research.amnh.org/ siddall/methods.

[2] En realidad es el par más similar, ya que se usa una técnica de distancia para obtener el árbol guía.

Aunque es intuitivamente muy interesante, no se ha usado en la práctica. Cualquiera que sea la idea para realizar los análisis moleculares, los resultados dependen de la matriz de transformación inicial; Wheeler (1995) desarrolló una metodología para comparar los parámetros de las transformaciones, conocida como "análisis de sensibilidad". Bajo esta idea, se hacen diferentes análisis con diferentes matrices de transformación, y se selecciona aquella que maximice algún criterio previamente seleccionado (ya sean las topologías, como el costo de las transformaciones); hasta ahora, esta idea solo se ha utilizado en el contexto de análisis con múltiples genes o entre genes y morfología simultáneamente.

1.1. Técnicas

Algunos programas (como CLUSTAL) usan un solo árbol guía derivado del cálculo de la distancia entre las secuencias; como todos los métodos basados en distancias, es dependiente del orden de entrada de los datos; Muscle aunque parte de un árbol guía de distancia, revisa el resultado del alineamiento, priemro a nivel goblal y luego a nivel local; otros programas (como MALIGN o POY) construyen distintos árboles y prefieren el alineamiento que tenga el menor costo. Aunque sean más lentos, POY y MALIGN producen mejores resultados (en cuanto a la calidad del alineamiento con miras a evaluar la filogenia) que CLUSTAL. Giribet et al. (2002) recomiendan que si se usa CLUSTAL, se prueben diferentes secuencias de entrada de los datos. Es importante recalcar que el árbol usado por CLUSTAL y por MALIGN es sólo un árbol para optimizar las secuencias, no un árbol filogenético como tal; mientras que el árbol generado por POY es un árbol de alineamiento y a la vez es un árbol filogenético. Dado que el objetivo de POY es el análisis simultáneo, a lo largo de este libro se usará POY como otro programa más de análisis. Aun así, POY puede producir los alineamientos implícitos de cada árbol (así se lo utilizará en este capítulo), es posible usar este resultado como entrada para programas de búsquedas que usen secuencias alineadas. Giribet et al. (2002) enfatizan que de usar los alineamientos, los costos usados para el alineamiento deben ser los mismos usados para el análisis filogenético. Al asignar los costos, hay que tener en cuenta la desigualdad triangular. Es decir, que el costo de una transformación nunca puede ser mayor al de una transformación equivalente, pero que tome otros estados. Un ejemplo clarifica esto: si las transiciones tienen un costo de 3 y las transversiones de 1, la transformación A→ G, tendría un costo de 3, pero si se hace de la forma A→ T→ G, el costo sería de 2 (dos transversiones). Esto haría que a los nodos se les pudiesen asignar estados no observados en los terminales.
Para facilitar tanto la velocidad de alineamiento como las asignaciones de homología, al usar POY se pueden (y preferencialmente se deben) dividir las secuencias analizadas en pequeñas secuencias o marcos; definidos mediante iniciadores universales, estructura secundaria o motivos conservados. Cuando las secuencias son muy desiguales en estos segmentos, muchas veces se prefiere eliminarlas a analizarlas (o se hace un análisis exploratorio que incluya esas secuencias).

1.2. Materiales

Secuencias para:
MALIGN (datos.malign.dat).
FASTA (datos.fas.dat, datos2.fas.dat).
MUSCLE (datos.ou.dat).

Parámetros para MALIGN (param.txt).
Matriz morfológica (datos.ss.dat).

1.3. Métodos

En CLUSTAL/MUSCLE:

1. Abra en un editor los archivos datos.ou.dat
2. ejecute CLUSTAL con los parámetros predefinidos cual es la relación transición/tranversión definida?
3. Modifique los parámetros de tal forma que obtenga una relación de ts/tv de (a) 1/2 (b) 1/5.
4. ejecute nuevamente CLUSTAL con estas modificaciones en los parámetros.
5. Cambie los parámetros dando a los gaps un costo del doble del costo de las transversiones.
6. Ejecute MUSCLE con los parametros predefinidos. ¿cuál es la relación transición/tranversión definida?
7. Repita el paso 5 y ejecute nuevamente MUSCLE con los nuevos parámetros.
8. Guarde el alineamiento en un archivo de texto en formato FASTA, PHYLIP o NEXUS.

En MALIGN:

1. Abra en un editor los archivos datos.malign.dat y param.txt.
2. Ejecute MALIGN con los parámetros predefinidos. ¿Cuál es la relación transición/ transversión definida?
3. Modifique el archivo de parámetros de tal forma que obtenga una relación de ts/tv de (a) 1/2 (b) 1/5..
4. Ejecute nuevamente MALIGN con estas modificaciones en los parámetros.
5. Cambie los parámetros dando a los gaps un costo del doble del costo de las transversiones.
6. Modifique el archivo de parámetros y esta vez produzca la salida para buscar el árbol en PAUP.
7. Revise la salida de cada una de las modificaciones y evalúe el tamaño de la matriz resultante en cada caso.

En POY:

1. Abra los datos en POY usando `read(''datos.fas.dat'')`, coloque todos los costos iguales con `transform(tcm(1,1))` y realice la búsqueda `build(100) select(unique) swap() select()`, cada instrucción o grupo de ellas van seguidas de `enter`.

2. Pase sus secuencias de homología dinámica a estática, use `transform(static_aprox)`.

3. Guarde el alineamiento en un archivo de texto usando `report(''alin.txt'',phastwinclad)`.

4. Revise el alineamiento con Winclada.

5. Repita los pasos anteriores incluyendo la matriz de datos morfológicos.

6. Repita los pasos anteriores con un alineamiento tipo FSO use `transform(fixed_states)`

7. Pese la matriz morfológica 2 veces el valor de los datos moleculares usando `transform((static,weight:2))`.

8. Cambie los parámetros de tal manera que los gaps valgan el doble que las sustituciones.

9. Elabore una matriz de costos para obtener relaciones de ts/tv de 1/2 o 1/5, léala usando `transform((all, tcm:(''costos.txt''))`, y genere el alineamiento.

10. Haga un análisis de sensibilidad usando las matrices `datos.fas.dat` y `datos2.fas.dat` usando los mismos costos para los dos conjuntos de datos y con costos diferenciales para cada tipo de datos, revise el apéndice B, página 10.

11. Haga un análisis tradicional (build,select,swap,select) y comparelo con un análisis sin seleccionar los árboles para swap (build,swap,select), ¿qué efecto tiene el select intermedio?

12. La secuencia análizada partala en al menos tres marcos y repita el proceso, evalue los tiempos usados en ambos procesos.

13. ¿Puede hacer el análsis del punto anterior con algún comando de POY? busquelo y una vez lo haga repita el proceso pero fraccionandio la secuencia usando en 1 y luego las secuencias usadas en dos.

14. Repita los tres pason anterioes usando ML.

15. ¿Cómo hace el proceso con ML? revise Pedraza et al 2013.

16. ¿Las topolgías obtenidas vía parsimonia son iguales (o no) a las generadas por ML?

17. ¿Los alineamíentos implícitos son equivalentes? use la longitud de los alineamientos y caracteres informatívos evaluados en winclada.

18. Repita todo el proceso usando FSO (¿És posible hacer FSO usando ML?)

1.3.1. Programas

Si desea hacer un alineamiento múltiple rápido use preferencialmente MUSCLE sobre CLUSTAL, pero si el objetivo es la filogenia sugerida por las secuencias es mejor usar MALIGN o POY, ya que estos tienen en cuenta el árbol final. Puede hacer una exploración en CLUSTAL y posteriormente llevar sus marcos como archivos separados para ser analizados con POY.

1.3.2. Comandos

MALIGN utiliza un archivo de parámetros para configurar el alineamiento; algunos de los parámetros más importantes son `changecost` para asignar el costo de una transformación y `gap` para asignar el costo de adicionar un gap. `matrix` asigna una matriz de costos del alineamiento. Las búsquedas pueden ser `quick`, una búsqueda rápida, o `build`, junto con `aspr` o junto con `atbr`, que permutan ramas, mientras que `keepalignment` indica el número de alineamientos igualmente óptimos que se van a retener. Cuanto más grande sea este valor más lento será su análisis. Con `hennig86`, o `nexus`, da el formato de salida para NONA o para PAUP*. POY cuenta con una interface de usuario que es relativamente rápida de dominar. Para los diferentes tipos de costos, el comando más importante es `transform()`. Por ejemplo, con `transform(tcm(1,2))` se da peso de 1 a las sustituciones y de 2 a los gaps o a los datos morfológicos (estáticos). Matrices de transformación más complejas pueden elaborarse y luego leerse con ese mismo comando. Una de las grandes ventajas de POY es que puede ser usado en un *cluster* de computadoras.

1.4. Preguntas

1.4.1. Práctica

Compare los álineamientos de MUSCLE y CLUSTAL, al modificar los distintos parámetros, ¿Son similares los resultados?
Compare los árboles de MALIGN, POY y los obtenidos en un análisis cladístico con el programa de su preferencia. ¿Son similares los resultados?
Compare sus resultados con los de sus compañeros. ¿Cómo son las longitudes de los árboles (las reportadas por POY) y las topologías?
Dados los diferentes costos usados en el análisis simultáneo de morfología y datos moleculares, ¿cuál cree usted que es el resultado óptimo y por qué?

1.4.2. Generales

Existe una disputa sobre una relación entre los juegos de costos y el soporte. Dados sus conocimientos, escriba un pequeño ensayo donde indique sus ideas, su posición y sus argumentos en esta discusión.

1.5. Literatura recomendada

Edgar, 2004.[Compara los alineamientos deivados de MUSCLE y CLUSTAL].
Frost et al., 2001 [Un artículo empírico sobre el análisis de sensibilidad].
Giribert, 2003 [Revisa la exploración de los resultados del análisis de sensibilidad vs. soporte de clados].
Giribet & Wheeler, 1999 [Uno de los pocos artículos que discute explícitamente el tema de los gaps].
Wheeler, 1995 [La propuesta del análisis de sensibilidad para la asignación de parámetros en los alineamientos].
Wheeler et al., 2006 [Una guía completa para POY].

APÉNDICE A

Formatos de archivos

A.1. Formatos de matrices

A.1.1. NONA

Es válido para NONA, TNT, Hennig86 (formato de morfología de POY) y WinClada; comienza con `xread`, luego el número de caracteres y el número de taxa, los polimórficos entre paréntesis angulares y los desconocidos con `-` o `?`. Al final, un punto y coma y si se desea la aditividad de los caracteres (comenzando desde 0). Termina con `p/` o `p-`, que los programas interpretan como fin del archivo.

```
xread 'Matriz ejemplo'5 4
out 00000
alpha 10-20
beta 1102[01]
gamma 1?111
lamda 11111
;
cc -0.2 +3 -4;
p/;
```

Para ADN (en NONA) se usa `dread`, con la clausula `gap` seguida de `?` si se quieren asumir los *gaps* como desconocidos, o con `;` si quiere que sean un quinto estado. Se usa codificación tipo IUPAC.

```
dread gap ; match . 'DNA'5 4
out ACGTC
alpha AT-CG
beta RTAAC
gamma CGAY-
lamda TCNCC
;
cc -.;
p/;
```

A.1.2. PAUP*

Se inicia con la clausula `#nexus`, y luego con el bloque `data`. se puede definir si los caractertes son morfológicos, ADN o proteínas. Los polimórficos se colocan entre paréntesis redondos. ADN en formato IUPAC.

```
#nexus
begin data;
dimensions ntax=4 nchar=5;
format missing=? gap=- symbols="0 1 2";
matrix
out 00000
alpha 10-20
beta 1102(01)
gamma 1?111
lamda 11111
;
end;
begin assumptions;
typeset tipoUno=unord:1-3 5, ord:4;
end;
begin paup;
[Aqui puede colocar instrucciones específicas de paup, por ejemplo búsquedas]
hsearch add=random;
end;
```

```
#nexus
begin data;
dimensions ntax=4 nchar=5;
format missing=? gap=- datatype=dna;
matrix
out ACGTC
alpha AT-CG
beta RTAAC
gamma CGAY-
lamda TCNCC
;
end;
```

A.1.3. MALIGN

El esquema de `MALIGN` es similar al de GenBank.

```
SequenceA
1 CAGCAGCACG CAAATTACCC ACTCCCGGCA CGGGAGGGTA GTGACGAAAA ATAACAATAC
61 CCGTC

SequenceB
1 CAGGCACGCA AATTACCCAC TCCCGGCAGA GGTAGTGACA AAAAATAACG ATACGGGACT
61 CCGTCAC

SequenceC
1 GGCACGGAGG TAGTGACGAA AAATAACGAT ACGGGACTCA TCCGAGGCCC CGTAATCGGA

SequenceD
```

```
1 AAATTACCCA CTCCCGGCAC GGAGGTAGTG ACGAAAAATA ACGATACGGG ACTCA

SequenceE
1 GAGGTAGTGA CGAAAAATAA CAATACAGGA CTCATATCCG AGGCCCTGTA ATT
```

(Para proteinas, el asterisco "*"forza la interpretación como proteina)

```
SequenceF
1 ILAVEELVI SLIVES

SequenceG*
1 AAYVTTTCC KKYK
```

A.1.4. POY

POY y Clustal, utilizan el formato de FASTA (la primera línea tiene un > seguido por el nombre y comentarios de la secuencia; la siguiente línea comienza la secuencia como tal).

```
>taxonA Comentarios
aaacgt
aac
>taxonB
aaacgt
```

A.2. Formatos de árboles

A.2.1. NONA

NONA, Hennig86, WinClada y TNT usan este formato; con * indican que hay mas árboles y con ; que es el último árbol. Nótese el espacio para separar los terminales. El primer taxon es 0. Al igual que en las matrices, p- o p/ indican el final de lectura del archivo. Winclada puede incluir al incio la lista de nombres, pero no es compatible con otros programas.

```
tread 'tres arboles'
(0 (1 (2 (3 4 ))))*
(0 (1 (2 3 4 )))*
(0 ((1 2 )(3 4 )));
p/;

tread 'solo un arbol'
(0 (1 (2 (3 4 ))));
p/;
```

A.2.2. PAUP*

En PAUP* el árbol está embebido en el archivo de la matriz o en un formato aparte. Los grupos son separados por comas y el primer taxon es 1.

```
#nexus

begin trees;
translate
1 lamda,
2 alpha,
```

```
3 beta,
4 gamma,
5 out
;
tree *primero=(5,(1,(2,(3,4))));
tree politomico=(5,(1,(2,3,4)));
tree tercero=(5,((1,2),(3,4)));
end;
```

El orden no altera los árboles en los programas. Así:
`(0,(1,(2,(3,4))))`
es igual a
`((1,((3,4),2)),0)`

APÉNDICE B

Algunos comandos básicos para POY

B.1. Análisis de sensibilidad con costos diferenciales

```
(* tomado de http://groups.google.com/group/poy4/ *)
(* dados dos conjuntos de datos en dos archivos *)
(* 1.fas y 3.fas *)

read(''1.fas'', ''3.fas'')
transform((names:(''1.fas''), tcm:''412.txt''),(names:(''3.fas''), tcm:''121.txt''))

store(''misdatos'')

build(5, trees:2)

select(best:1)

transform((all, static_approx))

report(''todo13.mtr'', phastwinclad)

(* Resultados en el archivo Sensitivity_results.txt *)

echo (''412+121 Resultados'', output:''Sensitivity_results.txt'')

report (''Sensitivity_results.txt'', treestats)

(* Deseche los árboles *)

select(best:0)

use(''misdatos'')

select(characters,names:(''1.fas''))

build(5)

select()
```

```
echo(''1.fas Resultados'', output:''Sensitivity_results.txt'')
report(''Sensitivity_results.txt'', treestats)
(* Deseche los árboles *)
transform((all, static_approx))
report(''matriz1.mtr'', phastwinclad)
use(''misdatos'')
select(best:0)
select(characters,names:(''3.fas''))
build(5)
select(best:1)
echo (''3.fas Resultados'', output:''Sensitivity_results.txt'')
report (''Sensitivity_results.txt'', treestats)
transform((all, static_approx))
report (''matriz3.mtr'', phastwinclad)
quit()
```