Universidad de Zaragoza

Bioinformática

Memoria de la práctica 3

Alineamientos con secuencias de SARS-COV-2

Autor:

Adrián Martín Marcos 756524

Zaragoza, España

Curso 2020 – 2021

Introducción

En esta tercera práctica se han realizado multialineamientos de secuencias de SARS-COV-2 proporcionadas por la profesora de la asignatura. Para ello se han usado diferentes programas, entre los que destacan MAFFT y una implementación propia del método de la estrella (se aportan los fuentes junto a este documento). Además, se ha estudiado previamente la herramienta MAFFT y las diferentes opciones de configuración de las que dispone.

Después, se ha realizado una comparativa de los resultados obtenidos con cada alternativa, y se han tratado de identificar sobre las secuencias alineadas las mutaciones correspondientes a la variante británica del coronavirus.

Índice

1. Pruebas con MAFFT

En este primer apartado se ha ejecutado la herramienta MAFFT con algunas de sus opciones principales, con el objetivo de comparar sus resultados y su tiempo de ejecución.

Las pruebas han sido realizadas con solo 25 de las secuencias de SARS-COV-2 de Aragón dadas como material para la práctica, ya que se buscaba realizar intentos rápidos con los que poder comparar las diferentes configuraciones que ofrece la herramienta.

Cabe destacar que, dado que MAFFT no muestra el tiempo de ejecución, todas las ejecuciones son prefijadas con el comando *time*.

1.1. Opción recomendada en la presentación

En primer lugar, se ha ejecutado MAFFT con el comando recomendado por los compañeros en la presentación:

```
time mafft --auto \
           --addfragments Aragon1_gisaid_hcov-19_2021_03_02.fasta \
           RefSeqWuhan.fasta > alineamiento25-mafft-auto.fasta
```

El tiempo de ejecución ha sido de solo 1'3 segundos, y el valor de SP del resultado de 29448'13. Es un muy buen resultado, y el tiempo de ejecución con el que se ha obtenido es extraordinariamente bajo.

1.2. Métodos progresivos

En esta segunda prueba, se ha probado la implementación de métodos progresivos de MAFFT, usando la opción FFT-NS-2 para alinear las secuencias. El comando para ello es el que se muestra a continuación:

```
time mafft --retree 2 \
           --addfragments Aragon1_gisaid_hcov-19_2021_03_02.fasta \
           RefSeqWuhan.fasta > alineamiento25-mafft-ns2.fasta
```

Tras superar la media hora de ejecución, y dado que se trata del método más rápido, se detuvo el programa con una señal.

1.3. Refinamiento iterativo

El objetivo de esta tercera prueba era usar las opciones de refinamiento progresivo de MAFFT para alinear las secuencias. Siguiendo los consejos de los compañeros que realizaron la presentación de la práctica, la parametrización concreta que se iba a probar era de dos ciclos, llevada a cabo con el siguiente comando:

```
time mafft --maxiterate 2 \
           --addfragments Aragon1_gisaid_hcov-19_2021_03_02.fasta \
           RefSeqWuhan.fasta > alineamiento25-mafft-NS-i.fasta
```

No obstante, dado que el tiempo de ejecución de la opción que se suponía más rápida ha sido tan elevado, no se ha llegado a probar esta configuración. Tampoco se han considerado opciones que a priori se pensaban probar pero que se sabe que son incluso más lentas que esta.

1.4. Conclusión

Se ha decidido usar la opción recomendada dado el tiempo de ejecución requerido por las diferentes alternativas con las que se ha experimentado.

2. Scripts implementados

En este apartado se describen brevemente los scripts implementados, así como la finalidad de los mismos y su forma de uso.

Cabe destacar que ha habido varios problemas derivados del uso de BioJulia para el desarrollo de esta práctica. Entre ellos, los más relevantes han sido la redefinición de símbolos entre las bibliotecas BioSequences y FASTX, el cambio del API entre la documentación de Bio.jl (que es la principal [6]) y la que incorpora cada módulo por separado (debido a una refactorización del proyecto [2]) y el acceso a la información almacenada en los tipos de datos que venían definidos (no era modificable y muchas veces estaba codificada en binario, por lo que ha sido necesario crear tipos de datos que permitiesen su manipulación).

Adicionalmente, se quiere hacer hincapié en la corrección del código entregado, en base al uso de tipos en los datos utilizados (pese a no ser neccesario por el tipado dinámico) y a la formulación de aserciones para asegurar que se cumplían algunos invariantes en ciertas secciones del código.

2.1. Biblioteca para lectura y escritura de ficheros

En primer lugar se ha implementado una biblioteca que abstrae el tratamiento de ficheros *FASTA* y *CSV*. Para ello, provee de funciones de lectura y escritura de estructuras de datos matriciales.

Adicionalmente, implementa el tipo de dato *Secuencia*, que no es más que una redefinición del tipo de dato *Record* de *BioSequences* para poder manipular las secuencias de DNA.

2.2. Multialineamiento con el método de la estrella

Este script implementa el multialineamiento de las secuencias de SARS-COV-2 dadas mediante el método de la estrella. Para ello, toma como centro de la estrella una secuencia que se le debe pasar explícitamente como parámetro, con el flag *–ref-seq*.

La implementación del método de la estrella llevada a cabo cuenta con dos fases diferenciadas: obtener los alineamientos por pares de cada secuencia con la de referencia y aglutinar todos los alineamientos en uno solo con la técnica *Once a gap - always a gap*.

La primera de ellas se lleva a cabo aplicando el algoritmo Needleman-Wunsch [1], que está implementado en la función *pairalign* de la biblioteca *BioAlignments* [3]. El sistema de puntuación que utiliza es el que se propone en el enunciado de la práctica, pero es parametrizable usando los flags *–match*, *–mismatch*, *–gap_ open* y *–gap_ extend*. Además, dado que la obtención de cada alineamiento por pares es independiente del resto, se ha paralelizado la ejecución del bucle en el que se implementa, de tal forma que se ha conseguido bajar el tiempo de ejecución para veinticinco secuencias a la mitad (véase el subapartado de parametrización).

La segunda implementa la técnica *Once a gap - always a gap* de manera secuencial (se podría paralelizar, pero hacerlo no es tan directo y además el tiempo de ejecución de esta fase no es tan elevado en comparación con la anterior). Para ello se sirve del tipo de dato *Alineamiento*, que fue diseñado para facilitar la aplicación de este paso con operaciones como la inserción de gaps sobre todo un conjunto de secuencias ya alineadas.

En la siguiente sección se profundiza en las parametrizaciones con las que se ha experimentado.

2.2.1. Parametrización

En primer lugar, se debe mencionar la existencia del flag *–help*, que hace las veces de manual del script, indicando las opciones existentes y su forma de uso.

Como se ha mencionado antes, hay instrucciones iterativas en el código que están preparadas para llevar a cabo una ejecución concurrente de sus iteraciones. Para que eso ocurra, se debe declarar antes la variable de entorno *JULIA_NUM_THREADS*, y darle como valor el número de hilos que queremos que use Julia en su ejecución. Se puede saber si el cambio se ha hecho efectivo viendo la salida estándar del script (antes de empezar el alineamiento, muestra el número de hilos que va a utilizar).

En la máquina en la que lo he ejecutado dispongo de cuatro núcleos físicos con dos hilos lógicos cada uno. Para determinar cuántos hilos debía configurar, he realizado pruebas alineando 25 secuencias de SARS-COV-2 con la de referencia. Sin paralelismo el tiempo de ejecución estaba en 83 segundos. Cuando he usado ocho hilos, el tiempo de ejecución se ha disparado por encima de 100 segundos. Por ello, he pasado a considerar el número de cores físicos en la parametrización, y he visto que con dos hilos se bajaba a 44 segundos, con tres a 38 segundos y con cuatro se volvía a subir a 53 segundos. Por lo tanto se han usado tres hilos en las pruebas posteriores con este script. El por qué de este valor se cree que se puede deber a que el scheduler de biblioteca de Julia ocupa el cuarto núcleo físico, degradando las prestaciones cuando se consideran cuatro o más hilos debido a cambios de contexto.

2.3. Cálculo de la matriz de puntuación de un alineamiento

Este script calcula la matriz de puntuación de un alineamiento dado.

Para hacerlo, recorre cada par de secuencias del alineamiento evaluando su parecido en base a un sistema de puntuación, y con el resultado obtenido rellena el elemento correspondiente en el triángulo inferior de la matriz (y su simétrico en el superior).

Cabe destacar que el sistema de puntuación utilizado es, por defecto, el propuesto en el guión de la práctica, pero se puede configurar con los flags *–match*, *–mismatch*, *–gap_open* y *–gap_extend*.

Otra consideración adicional es que los valores 'N' en las secuencias se evalúan como errores de secuenciación, y se interpretan como çualquier base", por lo que siempre dan como resultado un match, salvo en el caso de que la base con la que se compara sea un gap.

2.4. Suma entre pares dada una matriz de puntuación

Este script es una modificación del desarrollado en la práctica anterior para evaluar los alineamientos de Clustal Omega; calcula la puntuación de un alineamiento dado a partir de su matriz de puntuación. Por la fórmula que se aplica, el resultado obtenido es el *sum of pairs* del alineamiento (*SP*), es decir, la normalización de los elementos de uno de los triángulos simétricos de la matriz sin considerar la diagonal.

EL formato del fichero de entrada debe ser equivalente al que genera el script *obtenerMatrizPuntuacion.jl*.

2.5. Búsqueda de mutaciones de la cepa británica del SARS-COV-2

Este script implementa la búsqueda de las mutaciones correspondientes a la variante británica en un alineamiento de secuencias de SARS-COV-2.

Para ello, primero mapea en un diccionario las posiciones de la secuencia de referencia sin gaps sobre la secuencia de referencia del alineamiento (que sí tiene gaps y por tanto no coinciden), para poder buscar así por las posiciones que se quiere sin tener que considerar de su correspondencia.

Después, en función de los flags de la invocación (*–ORF1ab*, *–Spike*, *–ORF8*, *–N* o *–all*), se busca en uno o varios de los genes del SARS-COV-2 las mutaciones correspondientes a la cepa británica para cada secuencia distinta a la de referencia.

En el apartado cuatro se detalla el uso de este script para la detección de posibles casos de cepa británica.

3. Comparación de los resultados

En este apartado se compara el tiempo de ejecución y la calidad de los resultados obtenidos con los distintos programas con los que se han alineado las 512 secuencias de SARS-COV-2 de todo Aragón, aportadas como material para esta práctica.

3.1. Multialineamiento con MAFFT

Se ha usado el comando elegido tras la comparación realizada en el primer apartado. Es el siguiente:

```
time mafft --auto \
           --addfragments Aragon1_gisaid_hcov-19_2021_03_02.fasta \
           RefSeqWuhan.fasta > alineamientoTodoAragon-mafft-auto.fasta
```

El tiempo de ejecución fue increíblemente bajo, y el SP del resultado muy alto 1.

3.2. Multialineamiento con método de la estrella

Se pasa ahora a ejecutar ese mismo alineamiento con el script implementado. La ejecución del mismo se ha llevado a cabo como se muestra a continuación:

```
export JULIA_NUM_THREADS=3 # en mi máquina de 4 cores
./estrellaMSA.jl secuencias/Todo_Aragon_gisaid_hcov-19_2021_03_03_12.fasta \
                 -r secuencias/RefSeqWuhan.fasta \
                 -o alineamientoTodoAragon-estrella.fasta
```

El tiempo de ejecución ha sido mucho más elevado, pero bastante menos del que se podría esperar. Con respecto al resultado, es sorprendentemente bueno, y casi idéntico al de MAFFT pese a la simplicidad del algoritmo implementado 1.

3.3. Multialineamiento con Clustal Omega

Por último, se ha intentado realizar el mismo alineamiento con Clustal Omega. También a él se le ha dado como base para construir el multialineamiento la secuencia de referencia. El comando utilizado es el siguiente:

```
clustalo --profile1 RefSeqWuhan.fasta \
         -i Todo_Aragon_gisaid_hcov-19_2021_03_03_12.fasta \
         -o alineamientoTodoAragon-clustal.fasta -v
```

Tras tres horas de ejecución, se detuvo Clustal intencionadamente, ya que seguía en la fase de k-tuplas con un progreso del 180 %.

3.4. Conclusiones

Se comparan a continuación en una tabla los resultados obtenidos con cada uno de los programas. Nótese que el valor SP se ha obtenido de la forma siguiente:

```
./obtenerMatrizPuntuacion.jl alineamientoTodoAragon-*.fasta \
                             -o matrizTodoAragon-*.csv
./calcularSP.jl matrizTodoAragon-*.csv
```

La tabla comparativa se muestra a continuación:

	Tiempo ejecución	**Sum of Pairs (SP)**
Clustal Omega	$>$ 3 horas	undefined
Método de estrella	25 min 45 s	29430.55
MAFFT	22'4 s	29431.83

Tabla 1: Tabla comparativa de los programas de alineamiento probados

En base a estos resultados, queda claro que MAFFT es la mejor opción dada la relación tiempo-calidad que se ha evaluado en las pruebas realizadas.

4. Detección de mutaciones de la cepa británica

En este último apartado se va a comentar brevemente cómo se han analizado los alineamientos obtenidos para detectar posibles casos de la cepa británica.

Todo el análisis llevado a cabo se ha hecho a partir de la salida del script *detectarB117.jl* que se ha desarrollado, y éste se ha aplicado sobre el alineamiento de 25 secuencias de Aragón obtenido con el método de la estrella. El comando ejecutado es el siguiente:

```
./detectarB117.jl alineamiento25-estrella.fasta -a > mutaciones25-estrella.txt
```

4.1. Detección con el script entregado

La salida del comando anterior, entregada en el fichero *mutaciones25-estrella.txt*, muestra los resultados de la búsqueda de todas las mutaciones que caracterizan la cepa británica del SARS-COV-2 sobre las 25 secuencias distintas a la de referencia que hay en el alineamiento. Para cada una de ellas, se indica si ha sido encontrada o no la mutación que se buscaba.

En base a los resultados, parece bastante probable que los casos detectados como 1007859, 1059975, 1059978 y 1059979 sean de la cepa británica.

4.2. Visualización con MSAviewer

Por último, se muestra la visualización de la mutación A23063T en el gen Spike con el programa MSAviewer, junto con esa misma información vista como salida del el programa implementado.

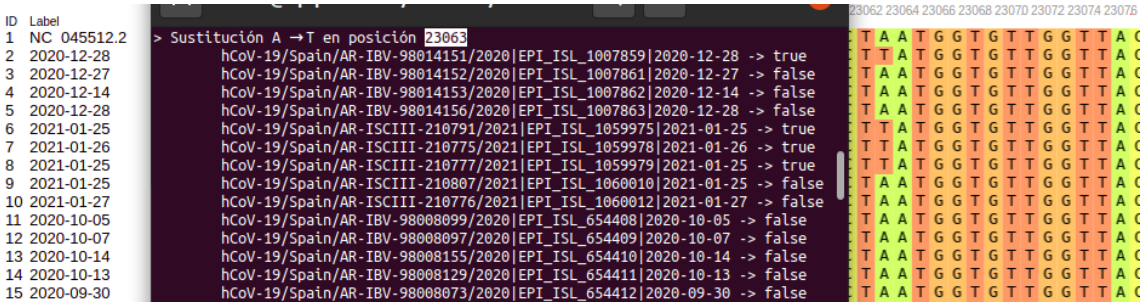

Figura 1: Captura en MSAviewer de mutación A23063T en el gen Spike, junto con la salida del script detectarB117.jl

Referencias

[1] *Algoritmo de Needleman-Wunsch.* URL: `https://es.wikipedia.org/wiki/Algoritmo_Needleman-Wunsch` (visitado 17-03-2021).

[2] *Aviso de descontinuación de Bio.jl y equivalencias con las nuevas librerías.* URL: `https://biojulia.net/post/biojl/` (visitado 17-03-2021).

[3] *Documentación de la función pairalign, que implementa el algoritmo de Needleman-Wunsch como parte de la librería BioAlignments.jl.* URL: `https://biojulia.net/BioAlignments.jl/stable/pairalign/` (visitado 17-03-2021).

[4] *Documentación de los algoritmos implementados en MAFFT.* URL: `https://mafft.cbrc.jp/alignment/software/algorithms/algorithms.html` (visitado 17-03-2021).

[5] *Imagen de cabecera de las páginas.* URL: `https://www.freepng.es/png-0m3sw0/` (visitado 16-02-2021).

[6] *Página de documentación principal de Bio.jl (actualmente depreciada).* URL: `https://biojulia.net/Bio.jl/stable/` (visitado 17-03-2021).