PEC1. Análisis bioinformático con el terminal

Magí Bas

2023-03-31

```
knitr::opts_chunk$set(engine = "bash", eval = FALSE)
```

Ejercicio 1 – Manipulación de datos Next Generation Sequencing (NGS) con Unix

1. **Descarga el fichero de datos desde el aula de la asignatura. Crea un directorio de trabajo en tu directorio "home" y**

copia en él, el archivo testdata.tar (1 punto)

2. **Descomprime el archivo (1 punto)**

```
$ tar -xvf testdata_1.tar

$ gzip -d testdata_1.fastq.gz
```

3. **Examina el contenido utilizando comandos bash con el fichero comprimido y sin descomprimir, sin editarlo (mínimo 4 opciones) (1 punto)**

```
# Examinar permisos, tamaño del archivo y fecha de modificación
$ ls -lh testdata_1.tar

# Examinar paraules, caracters y línies
$ wc testdata_1.tar

# Mostrar los nombres de los fcheros del archivo comprimido
$ tar -tf testdata_1.tar

# Mostrar la data de modificación, nº de líneas y permisos
$ tar tvf testdata_1.tar
```

4. **Examina el contenido del fichero utilizando solo comandos bash y contesta a las siguientes preguntas: (1 punto)**

a. ¿Qué codificación de calidad utilizan estos archivos: which offset?

```
$ file testdata_1.fastq
testdata_1.fastq: ASCII text
```

El offset que utilizan estos archivos es el ASCII text, American Standard Code for Information Interchange. Es un archivo de caracteres alfanumericos y signos de puntuación a los cuales asigna valores numericos de 7 bits de longitud.

b. ¿Cuál es la cabecera del tercer read? ¿Es paired-end (pe, forward) o mate-pair (mp, reverse)? Muestra únicamente la línea de la tercera cabecera

```
$ sed -n '12p' testdata_1.fastq
@@@DDDB>F+=D<+<AFFFIEFFIFDHFFE9GGCFC>?9?<)?@D*9;@;7BAECA8CDCE).=CEAEE?)771
```

Hay que tener en cuenta que cada 4 líneas empieza un nuevo read con "@"

c. ¿Cuál es la cabecera del antepenúltimo read? ¿Es paired-end (pe, forward) o mate-pair (mp, reverse)? Muestra únicamente la línea de la antepenúltima cabecera

```
$ tail -n 12|head -n 1 testdata_1.fastq
@DHKW5DQ1:324:C2G0EACXX:4:2312:6849:85099 1:N:0:TGAACTGG
```

```
paired-end (ep): 1
```

```
mate-pair (mp): 2
```

5. **Cuántas líneas contiene el fichero? (1 punto)**

```
$ wc testdata_1.fastq
  3352520   4190650 219167958 testdata_1.fastq
```

3352520 líneas

6. **Utilizando sólo comandos bash, ¿cuántos reads contiene el fichero? (1 punto)**

```
$ awk 'END {print NR/4}' testdata_1.fastq
838130
```

7. **¿Cuántos reads tipo paired-end (pe) y mate-pair (mp) contiene el fichero? (1 punto)**

```
$ grep -c "1:N" testdata_1.fastq
419065
$ grep -c "2:N" testdata_1.fastq
419065
```

8. **Extraer los pe/mp reads contenidos en el fichero de datos y escribirlos de manera separada en los ficheros testdata_1.fastq y testdata_2.fastq. Muestra el resultado (1 punto)**

```
#Utilizamos "1:N" i "2:N" para identificar los reads
$ grep -A 3 ' 1:N:' testdata_1.fastq > testdata_pe.fastq && grep -A 3 ' 2:
```

9. **a. Cuenta el número de reads que contienen la secuencia TGCACTAC en testdata_1.fastq**

```
$ grep -c "TGCACTAC" testdata_1.fastq
5924
```

b. **Cuenta el número de reads que comienzan con la secuencia TGCACTAC en testdata_1.fastq. Se denomina in-line barcode a los 8 primeros nucleótidos de cada secuencia. (1 punto)**

```
$ grep ^TGCACTAC testdata_1.fastq |wc -l
4443
```

10. **Modificar todas las cabeceras de cada read presentes en testdata_1.fastq. Reemplazar la parte de**

la cabecera que identifica el read como pe y los diferentes campos descriptivos por "/1" (formato Illumina) y escribe el resultado en testdata_1_nueva_cabecera.fastq (1 punto)

```
#Substituimos "1:N:0:TGAACTGG" por "/1
$ sed '/^@/ s/ 1:N:0:TGAACTGG/\/1/' testdata_pe.fastq > testdata_1_nueva_c
```

11. **Extraer los primeros 1000 reads de testdata_1_nueva_cabecera.fastq y salva el fichero como testdata_1_sub1000.fastq. Posteriormente comprime el fichero en formato gz. (1 punto)**

```
#Cada read son 4 lineas
$ head -n 4000 testdata_1_nueva_cabecera.fastq > testdata_1_sub1000.fastq
$ gzip testdata_1_sub1000.fastq
```

12. **Repite el ejercicio 10 & 11 intercambiando "/1" por**

"/2" en una única línea de comandos utilizando pipe "|" desde el fichero testdata_2.fastq hasta el fichero comprimido testdata_2_sub1000.fastq.gz (1 punto)

```
$ sed -i 's/2:N:0:TGAACTGG/\/2/g' testdata_mp.fastq && head -n 4000 testda
```

13. **Detalla todos los reads etiquetados con el inline barcode TGCACTAC en el fichero testdata_1_sub1000.fastq.gz y guárdalo en el fichero sample_TGCACTAC_sub.1.fastq . Muestra el resultado. (1 punto)**

```
#Descomprimimos
$ gzip -d testdata_1_sub1000.fastq.gz

#Buscamos la cantidad de barcodes
$ grep -c "TGCACTAC" testdata_1_sub1000.fastq
11
```

```
#Exportamos los reads a otro documento y lo abrimos con cat
$ grep -B 1 -A 1 "TGCACTAC" testdata_1_sub1000.fastq > sample_TGCACTAC.fas
$ cat sample_TGCACTAC.fastq

@DHKW5DQ1:324:C2G0EACXX:4:2312:9717:85068/1
ACGTCTACTGCACTACACCTTATCAGCATTAACCAGCATCAATGTATTTACTTACATCCACAGTGAATATATTG
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:10017:85129/1
TGCACTACACACACACACACACACACACACACACACACACAAACATGACACACATGCATGCTCAAACACACACAGT
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:10228:85233/1
TGCACTACTGCTCTGCTACTAAGAACAAGGTGTTTTCTATGCATTAGGGTTCAATTTAGTACATCTGCAATAAA
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:10404:85197/1
TGCACTACTGCAGGACACTGGCACTCGAGGCCTGGAGCTGCCCCTGAGCTAAGATCCCACTCTGTGGGCAATTT
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:12524:85224/1
ACACGACATGCACTACAGCTTAGTTTAGGGGGAAGTTTATGTGTATTTTCCATAAGGTTATGATGATGCTGACA
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:14565:85186/1
GTCAGTGTTGCAGGGCAAACACCTTAAAGCATGACACTTATCTGTGATGAAGTCCAGAGGCAAACCAAGAGGAA
+
```

```
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:2488:85365/1
TGCACTACTGCAGGCTGATCATTTAACCTCAAGCTGCTGCCTTCATAGTAATTAATTTTCATTTGAAACGCAGA
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:6985:85424/1
TGCACTACTGCAGGTTTCAGATGTATCCTTCATCCAACACAGCTGATTTAATTACCTCCTCAACATGTCTTGAA
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:13676:85286/1
TGCACTACTGCAGGTCAGCAATTTGTCTCTCATCATAGATGCCTGGTATAATATCCAGAAAGCCCTGTTTTAGT
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:18689:85419/1
TGCACTACTGCAGCCACAGAAATTCTTTTGTCCATGAAACCATAAAGCTGCACTTTCTTTTTGCCTTATAGTCT
+
--
@DHKW5DQ1:324:C2G0EACXX:4:2312:1712:85792/1
TGCACTACTGCATCGACTCTGCAGCTTCACAGGCTGCCACTTGTTCCTGACACGTTTTTGCTACATTGTTTTGC
+
```

14. ¿Cuáles son los 12 in-line barcodes (secuencias de longitud 8) más comunes en el fichero testdata_1.fastq y cuán frecuente aparecen? (2 puntos)

```
#Buscamos las secuencias de 8 bases de longitud al inicio de cada rea
$ grep -oE '^[ACGT]{8}' testdata_1.fastq | sort | uniq -c | sort -nr

  23071 TGTGACTG
  21327 GTACATCA
  19298 TGTGCAGT
  18703 CACACAGT
  18435 TGCATCGT
  15571 GTCAGTGT
  15465 GTCATGTG
  13799 ACGTCTAC
  13372 ACGTGCTG
  13093 CATGCGAC
  10335 GTACTCGT
  10006 GTGTACTG
```