Bioinformatica II

LM Biologia Evoluzionistica, Università di Padova

Docente: Dr. Stefania Bortoluzzi

Esercitazione 4

Padova, 7 dicembre 2017

GUIDA

Uso della shell Unix

Obiettivo dell'esercitazione

L'obiettivo dell'esercitazione è di familiarizzare con alcuni comandi della *shell* Unix e di usarli per analizzare file di dati bioinformatici in due dei formati più comuni (FASTA, vcf). I file per l'esercitazione possono essere scaricati dal seguente indirizzo web:

http://compgen.bio.unipd.it/downloads/esercitazione4.tgz

Spostarsi nella cartella dove sono stati decompressi i file dell'esercitazione

Usando il comando **cd** (*change directory*) spostarsi nella propria *home directory*:

```
$ cd
```

Per spostarsi automaticamente nella propria *home directory* usare il comando **cd** senza argomenti.

Decomprimere l'archivio contenente i file per l'esercitazione usando il comando tar:

```
$ tar xzvf ~/Scaricati/esercitazione4.tgz
```

Spostarsi nella cartella esercitazione4 con il comando **cd** e il *path* relativo verso la cartella esercitazione4:

```
$ cd esercitazione4
```

Il comando **pwd** (*print working directory*) mostra la *directory* in cui ci troviamo. I comandi che lanciamo agiscono sulla directory in cui ci troviamo:

```
$ pwd
```

```
/home/ale/esercitazione4
```

Il comando **pwd** restituisce il *path* o percorso dove ci troviamo. Il simbolo / (*slash*) è il separatore dei nomi delle cartelle.

Spostarsi nella propria *home directory* utilizzando il comando **cd** e in seguito spostarsi nella *directory* contenente i *file* per l'esercitazione utilizzando il *path* assoluto. Attenzione, nel vostro computer il *path* assoluto potrebbe essere diverso:

```
$ cd /home
```

```
$ cd /home/ale/esercitazione4
```

Creare un unico file FASTA a partire da 10 sequenze singole

Listare il contenuto della directory usando l'opzione l sul comando ls:

```
$ ls -l
```

Verranno visualizzati i nomi dei *file* presenti nella *directory*, tra cui 10 sequenze in formato FASTA di trascritti riconducibili a geni coinvolti nel cancro al seno. Ogni sequenza è in un *file* separato.

Visualizzare il contenuto di uno dei *file* con il comando **cat**:

```
$ cat seq01.fasta
```

Il comando **cat** "scorre" l'intero *file*. Per poterlo visualizzare una riga per volta usare il comando **less**:

```
$ less seq01.fasta
```

Usare le frecce verso l'alto e verso il basso per scorrere il contenuto del *file*. Per uscire dal comando **less** usare il tasto **q**. Un *file* fasta contiene un'intestazione che inizia con il carattere **>** e poi la sequenza vera e propria.

Concateniamo tutte le sequenze FASTA in un unico *file*. NB: ogni sequenza ha una propria intestazione, quindi concatenandole in un unico *file* saranno comunque riconoscibili:

```
$ cat seq*.fasta > all.fasta
```

***** è un carattere speciale (*wild card*) che corrisponde ad uno o più caratteri. Il simbolo **>** *redireziona* l'output di un comando (**cat** in questo caso) in un *file*.

Ispezionare il *file* appena creato con:

```
$ less all.fasta
```

Verificare che il numero di sequenze nel *file* creato sia corretto. Per fare questo usare il comando **grep** che permette di filtrare le righe di un *file* che hanno un *match* con un dato *pattern*.

```
$ grep ">" all.fasta
```

contare quante righe vengono date in *output*, oppure per farlo in maniera automatica usare l'opzione **-c** del comando **grep**:

```
$ grep -c ">" all.fasta
```

Se vogliamo ottenere informazioni sul comando grep o su qualsiasi altro comando, possiamo usare il comando man:

```
$ man grep
```

Oppure usiamo l'opzione -h del comando grep:

```
$ grep -h
```

Contare il numero totale di righe presenti nel *file* all.fasta utilizzando il comando wc

```
$ wc all.fasta

516 536 30936 all.fasta

$ wc -l all.fasta

516 all.fasta
```

Le pipe

Spesso può essere utile inviare *l'output* di un comando non a video ma ad un altro comando prima di essere visualizzato. Usando l'operatore pipe '**|**' comunichiamo alla shell di collegare *l'output* del primo comando all'*input* del secondo comando.

Ad esempio per ottenere il numero approssimativo di nucleotidi totali presenti nelle nostre 10 sequenze:

```
$ grep -v ">" all.fasta | wc -c
```

L'opzione -v indica a grep di invertire il *match*, cioè di non mostrarci le righe che hanno il *match*; è importante mettere le virgolette intorno al maggiore dato che è un carattere che ha un significato preciso nella shell (redirezionamento dell'*output* di un comando).
Da notare che il numero ottenuto di caratteri che otteniamo non è preciso dato che vengono conteggiati anche i *new line*, cioè un carattere "invisibile" che indica al computer di andare a capo

Analisi di un file VCF (Variant Call Format)

Per una descrizione dettagliata del formato vcf consultare il seguente documento in formato pdf:
http://samtools.github.io/hts-specs/VCFv4.2.pdf

```
##fileformat=VCFv4.2
##fileDate=20090805
##source=myImputationProgramV3.1
##reference=file:///seq/references/1000GenomesPilot-NCBI36.fasta
##contig=<ID=20,length=62435964,assembly=B36,md5=f126cdf8a6e0c7f379d618ff66beb2da,species="Homo sapiens",taxonomy=x>
##phasing=partial
##INFO=<ID=NS,Number=1,Type=Integer,Description="Number of Samples With Data">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Total Depth">
##INFO=<ID=AF,Number=A,Type=Float,Description="Allele Frequency">
##INFO=<ID=AA,Number=1,Type=String,Description="Ancestral Allele">
##INFO=<ID=DB,Number=0,Type=Flag,Description="dbSNP membership, build 129">
##INFO=<ID=H2,Number=0,Type=Flag,Description="HapMap2 membership">
##FILTER=<ID=q10,Description="Quality below 10">
##FILTER=<ID=s50,Description="Less than 50% of samples have data">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype Quality">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Read Depth">
##FORMAT=<ID=HQ,Number=2,Type=Integer,Description="Haplotype Quality">
#CHROM POS     ID        REF    ALT     QUAL FILTER INFO                              FORMAT      NA00001        NA00002        NA00003
20     14370   rs6054257 G      A       29   PASS   NS=3;DP=14;AF=0.5;DB;H2           GT:GQ:DP:HQ 0|0:48:1:51,51 1|0:48:8:51,51 1/1:43:5:.,.
20     17330   .         T      A       3    q10    NS=3;DP=11;AF=0.017               GT:GQ:DP:HQ 0|0:49:3:58,50 0|1:3:5:65,3   0/0:41:3
20     1110696 rs6040355 A      G,T     67   PASS   NS=2;DP=10;AF=0.333,0.667;AA=T;DB GT:GQ:DP:HQ 1|2:21:6:23,27 2|1:2:0:18,2   2/2:35:4
20     1230237 .         T      .       47   PASS   NS=3;DP=13;AA=T                   GT:GQ:DP:HQ 0|0:54:7:56,60 0|0:48:4:51,51 0/0:61:2
20     1234567 microsat1 GTC    G,GTCT  50   PASS   NS=3;DP=9;AA=G                    GT:GQ:DP    0/1:35:4       0/2:17:2       1/1:40:3
```

Noi ci concentreremo sulle "*data lines*" e non sulle righe dell'intestazione (*header file*). Dare un'occhiata al file con il comando less:

```
$ less sample.vcf
```

Contiamo il numero di righe nel file:

```
$ wc -l sample.vcf
```

Questo comando ci mostra il numero totale di righe nel file compresa l'intestazione. A noi interessa sapere quante varianti sono presenti nel *file*. Quindi procederemo con il creare un *file* vcf senza l'intestazione:

```
$ grep -v "^#" sample.vcf > sample_no_header.vcf
```

Il primo argomento di grep in questo caso è una *regular expression*, ossia un modo per indicare famiglie di stringe (parole). In questo caso la nostra *regular expression* specifica tutte le parole che sono all'inizio della riga (^) e che iniziano con #. Questa è una caratteristica unica delle righe presenti nell'intestazione del *file* vcf. Ricordatevi che l'opzione -v indica a **grep** di invertire il *match*, quindi di non mostrare le righe che "matchano" la *regular expression*.

Contiamo ora quante varianti ci sono all'interno del nostro *file*:

```
$ wc -l sample_no_header.vcf
```

Il comando cut

Vogliamo ora vedere quante varianti sono state identificate nel cromosoma 1. Ci serve il comando **cut** che come citato nella sua *man page*: "*cut out selected portions of each line of a file*". Quindi ci permette di selezionare soltanto alcune porzioni da ogni riga del *file*. In particolare le "porzioni" devono essere delimitate da un qualche carattere detto "*delimiter*" che nel nostro caso è il TAB.
Facciamo in modo di stampare il primo campo (il numero del cromosoma):

```
$ cut -f 1 sample_no_header.vcf
```

Ora selezioniamo solo le righe che contengono il cromosoma 1:

```
$ cut -f 1 sample_no_header.vcf | grep 1
```

Contiamo quante sono:

```
$ cut -f 1 sample_no_header.vcf | grep 1 | wc -l
```

Per finire proviamo a fare la stessa cosa partendo dal *file* vcf contenente anche l'intestazione, senza creare un file intermedio.

```
$ grep -v "#" sample.vcf | cut -f 1 | grep 1 | wc -l
```