Linux Command Line

Rachel Wood

2023-03-22

For this portfolio I will use the example of bioinformatics to showcase the utility of the Linux command line. The data we will be using is the genome sequence from the SARS-CoV-2 virus, which can be found at https://www.ncbi.nlm.nih.gov/genome/?term=SARS-CoV-2.

Working with Files and File Systems

We start in a folder containing the file downloaded from the link above:

```
ls
GCF_009858895.2_ASM985889v3_genomic.fna.gz
```

We first create a folder to work within using the `mkdir` command and use the `cd` command to move into this new file. Finally we can use the `ls` command to view the contents of this folder:

```
mkdir bioinf_ex
```

As expected, the directory is empty, but we can move the genome and protein files to our new folders. We use the `cd ..` to go one 'step back' in the directory and return to our original directory. We use the `ls` command again and then the `mv` command to move our two files into the new folder we have created:

```
mv GCF_009858895.2_ASM985889v3_genomic.fna.gz -t bioinf_ex/
cd bioinf_ex/
```

We now check these files have been successfully moved:

```
ls
GCF_009858895.2_ASM985889v3_genomic.fna.gz
```

We notice the `.gz` file extension, meaning we need to unzip the files. We can do this with the `gzip` command:

```
gzip -d GCF_009858895.2_ASM985889v3_genomic.fna.gz
```

```
ls
GCF_009858895.2_ASM985889v3_genomic.fna
```

We finally rename the files to something more convenient with the `mv` command:

```
mv GCF_009858895.2_ASM985889v3_genomic.fna genomic.fna
```

Evaluating Genomic Data

This section focuses on the genomic.fna file. We first use the `head` command to view the first few lines of the file:

```
head genomic.fna
NC_045512.2 Severe acute respiratory syndrome coronavirus 2 isolate Wuhan-Hu-1, complete
↪  genome
ATTAAAGGTTTATACCTTCCCAGGTAACAAACCAACCAACTTTCGATCTCTTGTAGATCTGTTCTCTAAACGAACTTTAA
AATCTGTGTGGCTGTCACTCGGCTGCATGCTTAGTGCACTCACGCAGTATAATTAATAACTAATTACTGTCGTTGACAGG
ACACGAGTAACTCGTCTATCTTCTGCAGGCTGCTTACGGTTTCGTCCGTGTTGCAGCCGATCATCAGCACATCTAGGTTT
CGTCCGGGTGTGACCGAAAGGTAAGATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGTCCAACTCAGTTTGC
CTGTTTTACAGGTTCGCGACGTGCTCGTACGTGGCTTTGGAGACTCCGTGGAGGAGGTCTTATCAGAGGCACGTCAACAT
CTTAAAGATGGCACTTGTGGCTTAGTAGAAGTTGAAAAAGGCGTTTTGCCTCAACTTGAACAGCCCTATGTGTTCATCAA
ACGTTCGGATGCTCGAACTGCACCTCATGGTCATGTTATGGTTGAGCTGGTAGCAGAACTCGAAGGCATTCAGTACGGTC
GTAGTGGTGAGACACTTGGTGTCCTTGTCCCTCATGTGGGCGAAATACCAGTGGCTTACCGCAAGGTTCTTCTTCGTAAG
AACGGTAATAAAGGAGCTGGTGGCCATAGTTACGGCGCCGATCTAAAGTCATTTGACTTAGGCGACGAGCTTGGCACTGA
```

We might want to know how long a genome is, this is often measured by the number of base pairs, here each pair is represented by a character - a base pair is either `A-T` or `G-C`. We use the `grep` command using the `-o` flag instead combined with the `wc` command to get the count of `A`, `C`, `G` or `T` characters in the file (excluding the header line):

```
tail -n+2 genomic.fna | grep -o [ATCG] | wc -l
29903
```

The square brackets indicate that we want want the count of any of these letters instead of the pattern `ATGC`

We can use similar commands to obtain the individual counts of the letters:

```
tail -n+2 genomic.fna | grep -o A | wc -l
8954
tail -n+2 genomic.fna | grep -o T | wc -l
9594
tail -n+2 genomic.fna | grep -o C | wc -l
5492
tail -n+2 genomic.fna | grep -o G | wc -l
5863
```

From this we can see the genome is `A-T` enriched.

The start codon Methionine is represented by `ATG`, we can see how many times this appears.

For this we first need to remove the header line and all new lines characters (denoted `\n`) and save the changes in a new file:

```
tail -n+2 genomic.fna > genomic_new.fna
echo -n $(tr -d "\n" < genomic_new.fna) > genomic_new.fna
```

Note the use of `echo`, this creates a subshell, which allows us to read from and write to the same file.

We can now search for the Methionine codon, using the `grep` command again,

```
grep -o ATG genomic_new.fna | wc -l
725
```

The covid nucleocapsid phosphoprotein starts with the motif ATGTCTGATAAT and ends with AACTCAGGCCTA. We want to isolate this gene, so find the character position of the character before the start of the patterns using `grep -b` (the `-b` indicates we want the byte off-set)

```
grep -aob ATGTCTGATAAT genomic_new.fna
28273:ATGTCTGATAAT

grep -ob AACTCAGGCCTA genomic_new.fna
29520:AACTCAGGCCTA
```

We can now write the gene to a new file using the `cut` command, so we want the 28274th character to the 29520th character:

```
cut -c 28274-29532 genomic_new.fna > phosphoprotein.txt
```

We can check the new `phosphoprotein.txt` looks how we expect it to:

```
head phosphoprotein.txt
ATGTCTGATAATGGACCCCAAAATCAGCGAAATGCACCCCGCATTACGTTTGGTGGACCCTCAGATTCAACTGGCAGTAACCAGAA
TGGAGAACGCAGTGGGGCGCGATCAAAACAACGTCGGCCCCAAGGTTTACCCAATAATACTGCGTCTTGGTTCACCGCTCTCACTC
AACATGGCAAGGAAGACCTTAAATTCCCTCGAGGACAAGGCGTTCCAATTAACACCAATAGCAGTCCAGATGACCAAATTGGCTAC
TACCGAAGAGCTACCAGACGAATTCGTGGTGGTGACGGTAAAATGAAAGATCTCAGTCCAAGATGGTATTTCTACTACCTAGGAAC
TGGGCCAGAAGCTGGACTTCCCTATGGTGCTAACAAAGACGGCATCATATGGGTTGCAACTGAGGGAGCCTTGAATACACCAAAAG
ATCACATTGGCACCCGCAATCCTGCTAACAATGCTGCAATCGTGCTACAACTTCCTCAAGGAACAACATTGCCAAAAGGCTTCTAC
GCAGAAGGGAGCAGAGGCGGCAGTCAAGCCTCTTCTCGTTCCTCATCACGTAGTCGCAACAGTTCAAGAAATTCAACTCCAGGCAG
CAGTAGGGGAACTTCTCCTGCTAGAATGGCTGGCAATGGCGGTGATGCTGCTCTTGCTTTGCTGCTGCTTGACAGATTGAACCAGC
TTGAGAGCAAAATGTCTGGTAAAGGCCAACAACAACAAGGCCAAACTGTCACTAAGAAATCTGCTGCTGAGGCTTCTAAGAAGCCT
CGGCAAAAACGTACTGCCACTAAAGCATACAATGTAACACAAGCTTTCGGCAGACGTGGTCCAGAACAAACCCAAGGAAATTTTGG
GGACCAGGAACTAATCAGACAAGGAACTGATTACAAACATTGGCCGCAAATTGCACAATTTGCCCCCAGCGCTTCAGCGTTCTTCG
GAATGTCGCGCATTGGCATGGAAGTCACACCTTCGGGAACGTGGTTGACCTACACAGGTGCCATCAAATTGGATGACAAAGATCCA
AATTTCAAAGATCAAGTCATTTTGCTGAATAAGCATATTGACGCATACAAAACATTCCCACCAACAGAGCCTAAAAAGGACAAAAA
GAAGAAGGCTGATGAAACTCAAGCCTTACCGCAGAGACAGAAGAAACAGCAAACTGTGACTCTTCTTCCTGCTGCAGATTTGGATG
ATTTCTCCAAACAATTGCAACAATCCATGAGCAGTGCTGACTCAACTCAGGCCTA
```

We can see here, the gene has been successfully isolated.