(Fasta) File Manipulation

Max Carter-Brown
Anglia Ruskin University, Wellcome Sanger Institute
max.carter-brown@aru.ac.uk

Contents

1. Outcomes

By the end of this document, you should be able to:

1. Do some basic text processing in the terminal
2. Know what the fasta file format is
3. Understand the basics of how to use `wc`, `sort`, `uniq`, and `grep`

2. Introduction

Manipulating files is the cornerstone of practical bioinformatics. We will often use other peoples software to convert input files into output files through analysis. At other times we will want to write our own software to do a specific, yet automated task. At the end of our analysis, we will often want to convert files to images to present figures, or into tables to put in to papers.

In this brief tutorial, we will use some simple commands on made-up fasta files. However, these commands are used by bioinformaticians frequently as they are so useful.

3. What is a fasta file?

A fasta file is a simple text file that is used to store biological sequences. Each sequence is stored as a header line, followed by a sequence. The header line starts with a `>` character, followed by the name of the sequence. The sequence is then stored on the next line(s). Usually sequences are stored on multiple lines, with a maximum of 80 characters per line.

Shell 1: An example fasta file

```
# here is an example of a fasta file
>example_sequence
GTACGGTACGTTATACGTACGTTG
```

4. Making a file to manipulate

To start with, we need a simple text file to manipulate. We will concentrate on fasta files, exploiting their structure to demonstrate some useful commands.

Shell 2: Creating a fasta file

```
# make a directory where we will do all the work
mkdir fasta_tutorial
cd fasta_tutorial
# make a file called example.fasta
echo ">seq1" > example.fasta
echo "GTACGGTACGTTATACGTACGTTG" >> example.fasta
# add another sequence
echo ">seq2" >> example.fasta
echo "TACGGTACGTTATACGTACGTTGG" >> example.fasta
```

Question time

1. Describe what the `>>` does in the `echo` command?
   What would happen if you replaced it with a single `>`?

5. Counting the number of sequences in a fasta file

Counting sequences might sound like a trivial task, but it can be useful to know how many sequences are in a file before you start processing it. As with all of the commands in this tutorial, there are many ways to achieve the same result. A fast and simple way to count the number of sequences in a fasta file is to count the number of lines that start with a `>` character.

Shell 3: Counting the number of sequences

```
grep -c ">" example.fasta
```

`grep` takes two arguments. The first argument is the pattern to search for, in this case the `>` character. This argument takes the form of a 'regular expression'. Regular expressions are an extremely useful way of finding patterns in text. `grep` will only print, or in this case, count lines that contain the pattern. The `-c` flag tells `grep` to count the number of lines that match the pattern.

Question time

1. What happens when you remove the `-c` flag?
   Can you find a way to remove the header lines from the output?
   Hint: check out the inverse flag.

6. Extracting sequences from a fasta file

For downstream analysis, we may want to extract a single sequence from a fasta file. If we want to extract 'seq1':

Shell 4: Extracting a single sequence

```
# extract a single sequence
grep -A 1 "seq1" example.fasta
```

You can see we have replaced the ">" from before with "seq1", so that the pattern we are searching for is the sequence name. The `-A 1` flag tells `grep` to print the line that matches the pattern, as well as the next line.

Question time

1. Can you find a way to extract the second sequence from the file?
   How could you extract both sequences (describe two ways if you can)?

7. Counting the number of characters in a fasta file

Counting the number of characters in a fasta file can be useful for a number of reasons. For example, you may want to know the total number of bases in a genome, or the length of a gene. To count the number of characters in a file, you can use the `wc` command.

Shell 5: Counting the number of characters

```
# filter out headers
grep -v ">" example.fasta | wc -m
```

Question time

1. Describe what the `|` does, and why is it useful?
   What happens when you remove the `-m` flag in `wc`?
   Given your knowledge of filtering sequences, how could you count the number of characters in the second sequence?

8. Finding and quantifying repeated sequences

First, let's add a few more sequences to `example.fasta` so we have some meaningful data to play with.

Shell 6: Adding a few more sequences

```
# add a few more sequences
echo ">seq3" >> example.fasta
echo "GTACGGTACGTTATACGTACGTTG" >> example.fasta
echo ">seq4" >> example.fasta
echo "GTACGGTACGTTATACGTACGTTG" >> example.fasta
# and some duplicate headers
echo ">seq4" >> example.fasta
echo "GTACGGTACGTTATACGTACGTTG" >> example.fasta
echo ">seq4" >> example.fasta
echo "GTACGGTACGTTATACGTACGTTG" >> example.fasta
```

You can see that we added extra 'q4' sequences with the same sequence.

Now we are going to find and quantify repeated sequences. This is a useful task if you are trying to clean up a file before analysis. To do this, we will use the `sort` and `uniq` commands.

Shell 7: Finding and quantifying repeated sequences

```
# extract headers first
grep ">" example.fasta > headers.txt
# now sort and count the number of unique headers
sort headers.txt | uniq -c
```

And we will finish off with a few questions.

Question time

1. What happens when you remove the `-c` flag in `uniq`?
2. Can you find a way to sort the headers in reverse order?
3. How could you find and quantify repeated sequences in the sequences themselves?

9. Counting the number of bases in a fasta file

This seemingly simple task is actually a bit more complex than it first appears. This is a minor warning that the following explanation might be a bit more complex than the other commands we have used.

Finding the total sequence length in a fasta file is a more complex problem. This is because a fasta file can contain sequences that are split over multiple lines. If the fasta file is a single line ***and a single record***, we can use:

Shell 8: The simplest case

```
# remove headers          and count characters
grep -v ">" example.fasta | wc -m
```

Otherwise things get a bit more complicated. This is because `wc` will count the number of characters in the file, but this will include the new line characters (yes, new lines themselves are characters, defined by `\n`). So we need to remove these before counting the characters. We can do this with `tr`, which is a kind of text replacer.

Shell 9: Account for newlines in most cases

```
# remove headers          and newlines  and count characters
grep -v ">" example.fasta | tr -d '\n' | wc -m
```

Note that this will count the bases in the entire file, if you want specific records, the problem unfortunately gets even harder.

Question time

1. Can you explain why then problem of counting bases per record is non-trivial?
2. Can you write a non-code explanation of how you might approach the problem?