Exercise: Analyzing SAM alignment files with Linux command-line utilities.

The official SAM format specification document is here.

A SAM file begins with 'header' lines that begin with @, and provide information about the reference sequences used by the alignment program, which alignment program was used, and (optionally) additional information about the source of the sequence data. After the header lines, the main part of the SAM file is tab-delimited columns of results of trying to align each sequence read in the input dataset to some region of one or more reference sequences. Some 'fields', or columns, of the SAM file contain text data, and some contain numerical data, including "bit-wise flags" that are pretty obscure. Fields 1 – 11 are fixed, but optional fields can occur after column 11 so line length can be variable, and the same fields are not always in the same columns.

The first 11 columns in a SAM file are defined as the following:

Col	Field	Description
1	QNAME	Query (pair) NAME – the sequence read ID
2	FLAG	bitwise FLAG
3	RNAME	Reference sequence NAME
4	POS	1-based leftmost POSition/coordinate of clipped sequence
5	MAPQ	MAPping Quality (Phred-scaled)
6	CIGAR	Compact Idiosyncratic Gapped Alignment Report (see below)
7	MRNM	Mate Reference sequence NaMe ('=' if same as RNAME)
8	MPOS	1-based Mate POSition
9	ISIZE	Inferred insert SIZE
10	SEQ	query SEQuence on the same strand as the reference
11	QUAL	query QUALity (ASCII-33 gives the Phred base quality)

These values can be retrieved or searched in a SAM format file using the `awk` variables $1 through $11, or by using the utility programs `cut`, `sort`, `uniq`, `wc`, and `grep`. Note – names of programs and utilities are given in `Courier New` font to make the spacing more obvious, because spacing is important in command lines. Words in *`italic Courier New`* font represent arguments to the commands; these should be replaced by something more meaningful before executing the command.

CIGAR format: A CIGAR string is comprised of a series of operation lengths followed by letters identifying the operations. The conventional CIGAR format allows for three types of operations: M for match or mismatch, I for insertion and D for deletion. The extended CIGAR format allows four more operations, as is shown in the following table, to describe clipping, padding and splicing. The CIGAR string reports the number of bases, followed by the type of operation. See http://davetang.org/wiki/tiki-index.php?page=SAM for example alignments and CIGAR strings describing them.

operation	Description
M	Alignment match (can be a sequence match or mismatch)
I	Insertion to the reference
D	Deletion from the reference

N	Skipped region from the reference
S	Soft clip on the read (clipped sequence present in <seq>)
H	Hard clip on the read (clipped sequence NOT present in <seq>)
P	Padding (silent deletion from the padded reference sequence)

Bitwise Flag Encoding: From a SeqAnswers forum posting by Simon Andrews regarding the "flag" field of the SAM file (available at http://seqanswers.com/forums/showthread.php?t=2301):
The SAM flag field, although it appears as a single number, actually contains several pieces of information which have been combined together. It is a bitwise field, which means that it makes use of the way that computers represent numbers to store several small values stored in one large value.
If you think of a standard integer as being composed of 11 bits (0 or 1) then it would look like:
00000000000
However SAM uses this single number as a series of boolean (true false) flags where each position in the array of bits represents a different sequence attribute.
Bit 0 = This query (read) was part of a pair during sequencing
Bit 1 = Both this query (read) and the mate of the pair are mapped
Bit 2 = The query sequence is unmapped
Bit 3 = The mate is unmapped
Bit 4 = Strand of query (0=forward 1=reverse)
Bit 5 = Strand of mate (0=forward 1=reverse)
Bit 6 = The query is the first sequence of a pair
Bit 7 = The query is the second sequence of a pair
Bit 8 = The alignment reported is not the best alignment of this sequence
Bit 9 = The sequence or alignment does not pass quality controls
Bit 10 = The sequence is considered to be a PCR or optical duplicate
Constructing the value from the individual flags is fairly easy. If the flag is false don't add anything to the total. If it's true, then add 2 raised to the power of the bit position.
For example:
Bit 0 - true - add 2**0 = 1 [note that this means all flag values for paired-end reads will be odd numbers]
Bit 1 - true - add 2**1 = 2
Bit 2 - false - add nothing [A flag value of 4 means the read is unmapped]
Bit 3 - false - add nothing
Bit 4 - true - add 2**4 = 16
Bit 5 - true - add 2**5 = 32
Bit 6 - true - add 2**6 = 64
Bit 7 - false - add nothing
Bit 8 - false - add nothing
Bit 9 - false - add nothing
Bit 10 - false - add nothing
Bit pattern = 00001110011 = 64+32+16+2+1 = 115. Note that the lowest-numbered-bit is at the right end of the string of bits, and the highest-numbered-bit is at the left end of the string.
So the flag value would be 115.
To convert from the numerical value back to the flag values, use the bc tool in the Linux shell. The command `echo "obase=2; 115" | bc` sends the base-10 value 115 to the bc program with instructions to convert it to base 2; bc returns the answer 1110011 (the leading zeros are not displayed unless another formatting command is given). The quotes are required in Linux, but not in Gnu On Windows.

Exercise: use the smallRNA-seq.sam example SAM file in the bit815 folder, and the command-line utilities `cut`, `sort`, `uniq`, `wc` and `grep`, to answer these questions

From http://rous.mit.edu/index.php/Manipulate_alignment_files_using_UNIX_commands

1. What is in the file?
2. How many alignments of reads to reference sequences are in the file?
3. How many different reads are in the file?
4. How many distinct sequences are in the file?
5. How many reads are uniquely mapped?
6. How many reads are multi-hits?
7. How many alignments are reported for each read?
8. How many different reference sequences are represented in the file?
9. List the reference sequences according to the read distribution
10. Which are the most and least represented chromosomes?
11. What mapping flags are in the file?
12. What reads and alignments are mapped on the reverse strand?
13. What is the relationship between mapping qualities and number of multiple hits in the file?
14. How many reported alignments are gapless?
15. How many reported alignments are perfect matches?

Solutions:

1. What is in the file? Use `head` to look at the first 10 lines, `tail` to look at the last 10, or `more` to display the file to the screen, one screen at a time. Review the SAM format specification document (http://samtools.sourceforge.net/SAM1.pdf) to see what the different parts of the SAM file represent.

2. How many alignments of reads to reference sequences are in the file? Each alignment is one line of the SAM file, but not all lines are successful alignments – the header lines (those starting with @) contain information about the reference sequences and alignment program, and some reads do not align to a reference sequence but are still reported in the alignment section of the file. Unmapped reads have a flag containing a value of 4 in field 2, but that can be hidden by the presence of other flag values. Unmapped reads also have an asterisk (*) in field 3, the reference ID field, so counting the number of asterisks in field 3 or the number of 4 values in field 2 will let us evaluate the number of unmapped reads. The `cut` function extracts one or more columns from a tab-delimited file, and `grep -c \*` counts the lines of output that contain an asterisk. Note that an asterisk is a meta-character for `grep` that means 'match any number of any character except newline', so it must be "escaped" using the backslash to be taken as a literal search term. The whole command, for field 3, is `cut -f3 smallRNA-seq.sam | grep -c \*` – either quotes or \ are necessary to have the * interpreted as a literal character rather than a special character. This yields the result that 846 lines contain non-aligned reads. The alignment lines all begin with a read ID (and in this file, all read IDs begin with the characters "1:") can be counted using `grep -c "^1:"` – this yields a count of 3889. The difference between all alignment lines, and alignment lines for unmapped reads, yields the number of alignments. This can be calculated using the `bc` command-line calculator: `echo 3889-846 | bc` yields the answer 3043. Note that the last 3889 lines of the file are alignment data – any commands that should only be applied to the

alignment data can use the command `tail -3889 smallRNA-seq.sam` to extract only the alignments from the file for further analysis.

3. How many different read IDs are in the file? The query (read) ID is in field 1. Some reads may have multiple alignments, so the number of lines is not necessarily the number of reads. Use `tail -3889 smallRNA-seq.sam | cut -f1` to capture the first field of the alignment lines, then `sort` to get like entries on adjacent lines, and `uniq` to recover the list of unique read ids. This list can be piped to the `wc -l` command to count the lines: `tail -3889 smallRNA-seq.sam | cut -f1 | sort | uniq | wc -l` Answer: 2194

4. How many different read sequences are in the file? Read ids represent different molecules in the flowcell lane, but some of those molecules may be identical due to PCR artifacts or bias in library construction. The query (read) sequence is in field 10, so a similar pipe of `cut`, `sort`, `uniq`, and `wc` can be used to determine the number of unique sequences. In this case it is not necessary to eliminate the header lines, because those have only three fields: `cut -f10 smallRNA-seq.sam | sort | uniq | wc -l` Answer: 3155. How can there be more unique sequences than unique read IDs? This SAM file is from an alignment of paired-end reads, and the FASTQ headers do not contain information about which read of the pair is being mapped, so the two reads of a pair have the same read ID in field 1.

5. How many reads are uniquely mapped? This can be done using the `-u` option to `uniq`, which returns only those items that are unique in the list of data supplied to the function, or using the optional flag XT:A:U produced by the BWA aligner, which specifies that the read was uniquely mapped (see http://bio-bwa.sourceforge.net/bwa.shtml; the description of optional fields in the SAM output is toward the bottom of the page). This file is not well-suited to counting read IDs that appear only once, because the paired-end reads from the same cluster have the same ID, so those seem to be non-uniquely mapped even though they may well be uniquely mapped. The command `grep -c XT:A:U smallRNA-seq.sam` returns a value of 1909.

6. How many reads are multi-hits? The `-d` option to the `uniq` command specifies that the returned list should include only items detected more than once, so one option would be to use `cut -f1 smallRNA-seq.sam | sort | uniq -d | wc -l` to retrieve the read IDs, sort them, recover the list of those that appear multiple times, and count the number of items in the list. As previously noted, however, this file contains repeated read IDs even for those reads that map uniquely, so that approach would not work well. Fortunately, the BWA alignment program that produced this SAM file includes an optional flag that denotes whether a read mapped uniquely or repeatedly, as described in section 5. The command `grep -c XT:A:R smallRNA-seq.sam` returns a value of 337.

7. How many alignments are reported for each read? The `-c` option to the `uniq` command specifies that the returned list include a count of the number of times each list element was detected. This list can be sorted in numerical order using options to the sort command, and redirected to a text file using the > operator: `tail -3889 smallRNA-seq.sam | cut -f1 | sort | uniq -c | sort -nr > sortedreadcount.txt`
Look at the resulting file – you see that many reads are mapped two times, but many are also mapped once. How would you count the number of reads mapped twice versus once? Note that the default delimiter for the `cut` command is a tab, but the output of `uniq -c` is a space-separated list, so `cut-f1` will not return the column of count values. Instead, use the so `cut -c1-8` option to return characters 1 to 8 from each line of output from the `uniq -c | sort -nr`

pipeline , then repeat the `sort | uniq -c` steps to count the number of occurrences of each of the count values. The entire command sequence then becomes: `tail -3889 smallRNA-seq.sam | cut -f1 |sort | uniq -c | sort –nr | cut –c1-8 | sort | uniq –c` – 499 read IDs occur once, and 1695 read IDs occur twice.

8. How many different reference sequences are represented in the file? The name of the reference sequence is field 3, so `tail -3889 smallRNA-seq.sam | cut –f3 | sort | uniq | wc -l` will extract the list of names, sort it, and return the unique values to the `wc -l` function, which counts 849 different reference sequences. These are contigs in a transcriptome assembly – if we were analyzing RNA-seq reads mapped to a reference genome, we might expect to see only a few chromosomes, say 5 for Arabidopsis, 10 for maize, or 22 for humans.

9. List the reference sequences according to the distribution of reads mapped to each reference. This is simply an extension of the result from question 9 – rather than simply counting the unique reference names with `wc –l`, we can use the `uniq -c | sort -nr` combination to count the number of occurrences of each reference sequence name, sort the names in descending numerical order, and then use the `head` command to see just the first 10 lines of output:
`tail -3889 smallRNA-seq.sam | cut –f3 | sort | uniq –c | sort –nr | head`

10. Which are the most and least represented chromosomes? This is more relevant to the myfile.sam example file provided on the rous.mit.edu webpage than to the smallRNA-seq.sam example file, because the myfile.sam example shows alignments of reads to the human genome, so reference sequence names are chromosomes. The command used for question 9, if applied to the myfile.sam example file, produces a list of chromosomes with the number of reads mapped to each. Using `head -1` or `tail -1` as the last command in the pipeline displays only the first element or the last element in the list, respectively. For the RNA-seq data mapped to transcriptome contigs in the smallRNA-seq.sam example file, the number of reads mapped to a contig is proportional both to the size of the contig (ie the length in base-pairs) and to the abundance of the corresponding transcript. Where can we find information on the size of the reference contigs?

11. What mapping flags are in the file? Mapping flags are in field 2, so extract field 2, sort, find unique elements, count, and report sorted counts (if desired) to answer this question.
`tail -3889 smallRNA-seq.sam | cut –f2 | sort | uniq –c | sort -nr`

12. What reads and alignments are mapped on the reverse strand? This is easy with the myfile.sam example file from rous.mit.edu, because that file has flag values of either 0 or 16, and a flag value of 16 means the read mapped on the reverse strand of the reference sequence. The smallRNA-seq.sam sample file is much more difficult – no flags have the value of 16, because all reads were sequenced as pairs, and many of the reads have other bit values contributing to the final flag value. Answering this question for the smallRNA-seq.sam file would require parsing the bitwise values out of the flags, which is an advanced topic.

13. What is the relationship between mapping qualities and number of multiple hits in the file? The reads mapped to multiple reference sequences are denoted with XT:A:R, as noted in section 6. The following command extracts lines containing that string, then captures the mapping quality score from field 5 and produces an ordered list of the most common mapping qualities: `grep XT:A:R smallRNA-seq.sam | cut –f5 | sort | uniq –c | sort –nr | head` – substitute XT:A:U to produce a list of mapping scores for uniquely-mapped reads.

```
XT:A:R      XT:A:U
 203 0      744 60
  47 36     297 37
  41 29     226 29
  13 25     195 23
  11 15     145 15
  10 23      73 25
   7 22      63 17
   3 14      29 0
   1 9       26 20
   1 12      22 46
```

14. How many reported alignments are gapless? The reads in the smallRNA-seq.sam file are 40 nt long, so an ungapped alignment will have a CIGAR string in column 6 of "40M". The count of ungapped alignments can be obtained from `cut –f6 smallRNA-seq.sam | grep –c 40M` – there are 2340 ungapped alignments.

15. How many reported alignments are perfect matches? The optional field XM produced by the BWA aligner reports the number of mismatches in the alignment – the output in that field for a perfect match alignment is XM:i:0. Because this field is flanked by other optional fields in the SAM file, it is not always in the same column, so we cannot use `cut` to retrieve the correct field. Instead, we can use `grep -c XM:i:0 smallRNA-seq.sam` – there are 157 alignments that are perfect matches.