SAM file post-processing

Nearly all mappers will output a very plain, un-sorted, un-compressed, un-processed SAM file. And most of the various down stream analysis you may want to do: mutation calling, peak calling, gene expression require a more processed output. The requirements are different for the different types of analysis. For mutation calling there is a rather complex serious of steps that need to be done to prepare the mapping files for the mutation calling algoritms. For gene expression counting some programs (like `htseq-count`) require nothing although it is still a good idea to compress the plain text SAM files to BAM's, which are compressed.

For post processing SAM files there are two tools widely in use:

- PICARD (http://broadinstitute.github.io/picard/)
- SAMTOOLS (http://www.htslib.org)
  Note the older version of samtools still ranks first in google (http://samtools.sourceforge.net). For the course we have loaded the newer version and I suggest sticking with that one. It has a lot on newer features.

  General tip; you should balance sticking with a fixed copy of software versus continually updated to the latest. It is not an easy balance to find. And at some point in research you need to fix/freeze the pipeline in the same way there are data freezes.

These two packages have a lot of overlap. One big difference though is that PICARD does not have any way to stream files while SAMTOOLS does. I am a huge (HUGE) fan of streaming (pipeing) and it really bothers me that PICARD will not let me use that feature of UNIX. However, I **STRONGLY** recommend using PICARD whenever there is a choice. You will be much happier although you will have lots of intermediate files.

The one exception; indexing. If you need a quick way to index `samtools index` is the way to go.

Simple PICARD post processing

As I said the exact post processing steps you need to do will depend on the down stream analysis being done; however there is a core set of steps that nearly everything needs:

- Sorting
- Compressing (SAM->BAM)
- Indexing
- Marking Duplicates

The first 3 are almost universially needed so most mapping pipelines will simple just do them. MarkDups is not always needed but you can do it in a reversable way (ie by marking rather than removing duplicates) so you can safely us it also.

PICARD actually has a module for each of these steps; however since most of the time you are manipulating a BAM file you also want to sort and index it you can do all three of the first steps with the sort command.

First find the PICARD JAR file. There should be a copy here:

```
/usr/local/picard/build/libs/picard.jar
```

You should add this to your `config.sh` script along with the paths to other jar's we will be using.

```
PICARD=/usr/local/picard/build/libs/picard.jar
GATK=/usr/local/GATK/GenomeAnalysisTK.jar
MUTECT=/usr/local/mutect/mutect-1.1.7.jar
SNPEFF=/usr/local/snpEff/snpEff.jar
```

To test if that worked type:

```
java -jar $PICARD
```

You should see a nice colorful (although that depends on your terminal) help screen. You might want to send the output through more (or less) but if you are getting the colors that will be messed up. And it goes to stderr not stdout. But try the following:

```
java -jar $PICARD 2>&1 | less -R
```

to keep the colors and get paging to work.

What we want to do is sort some of the SAM files we generated in the previous exercise. The command to do that is SortSam. PICARD has super nice and useful help screens. Type

```
java -jar $PICARD SortSam
```

and it will tell you how to run the SortSam command. PICARD does not use the unix convention for command options `-x XYZ`, instead it uses a different one:

```
java -jar $PIACARDJAR SortSam I=Input.sam O=Output.bam SO=coordinate
```

It also decides what to do based on the extension of the output file. `.bam` generates BAM's, `.sam` generates SAM's. `SO=coordinate` means sort the reads by there position along the genome. Alternative you can sort them by the name of the read (`queryname`) which is useful if you need to process paired end reads together.

One last option to add. You can not see it because it is a generic option use the `-H` option (PICARD does use unix style options for some things) to see it.

```
java -jar $PIACARDJAR SortSam -H
```

The options is to create an index: `CREATE_INDEX=true`

So get the SAM files we made in the previous exercise ready for use in IGV with the following command.

```
java -jar $PICARD SortSam \
    CREATE_INDEX=true \
    SO=coordinate \
    I=inputFile.sam \
    O=outputFile.bam
```

where you need to pick inputFile and outputFile names accordingly. Use this oppurtunity to give your files meaningful names.

If you run this you will likely get the following error.

```
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
```

java has be default a very small amount of memory allocated and even the tiny SAM files we are looking at need more. The virtual machines have 8Gb of RAM. If you are using your own machine check out much it has. 2Gb should be enough for the small files we are using. If you are an advanced UNIX person you might want to try searching for google for the answer before looking below

To tell JAVA to use 2Gb you do:

```
java -Xmx2g -jar $PICARD SortSam \
    CREATE_INDEX=true \
    SO=coordinate \
    I=inputFile.sam \
    O=outputFile.bam
```

If you took the Bash Scripting module you probably already have a script ot wrap picard. If not consider wrapping this now.

Picard for stats

Another extremely use part of the PICARD tool kits are statistics. They have large number of statistics modules or what they call metrics you can compute on BAM files. A partial listing (taken from the help screen):

CollectAlignmentSummaryMetrics
Produces a file containing summary alignment metrics from a SAM or BAM.

CollectInsertSizeMetrics
Writes insert size distribution metrics for a SAM or BAM file

CollectMultipleMetrics
A "meta-metrics" calculating program that produces multiple metrics for the provided SAM/BAM

CollectRnaSeqMetrics
Produces RNA alignment metrics for a SAM or BAM file

I suggest at least running the `CollectAlignmentSummaryMetrics` on the sorted BAM files from the previous section and perhaps the `CollectRnaSeqMetrics` on the RNA mapped BAMs.

Samtools view

As I said you should try to use PICARD when possible for manipulating SAM/BAM files and also for stats. But there is one samtools command that is incredible useful. It is `samtools view`. It can be use to convert SAMs to BAMs and visa verse. But its perhaps most useful features are

- listing (`cat`-ing) BAM files. If you do:

```
samtools view bamfile.bam
```

it will uncompress and list the contents of the BAM file to the terminal. This can be usefull in an enourmous number of ways and there are a number of options. Do

```
samtools view
```

by itself to see them. One of things very useful features of samtools view is it can extract reads from a sepecific range from an index BAM.

```
samtools view bamfile.bam chr22:12345000-12346000
```

will extract the reads that overlap that 1,000bp region. Play with it on the BAM files you have created.

- indexing large FASTA files and extracting subsequences

You can use `samtools faidx` to both index and then once index extract subsequences from a large FASTA file. This second use case is not well known but can be enormously useful.

For example if you have already run

```
samtools faidx genome.fa
```

you will know this is done if there is a file called `genome.fa.fai`, then

```
$ samtools faidx $GENOME_FASTA 7:123456-123557
>7:123456-123557
```

```
TGGGCCTCATCCATACACTTCCCAGCAACCGTGCATCTCGGCTGGCAGCCTTCACTCCCG
GAGGGGTGGGTACCGGGGTGGAGGGACGGCATAGGGTGATCC
```

FASTQ \(\to\) ProcessedBAM wrapper script

Write a wrapper script that creats finished ready to be analyzed BAMs from FASTQ files.

It should take as input:

- Path to GENOME_INDEX
- 2 FASTQ files for paired end mapping
  - If you have time make your script work in both PE and SE modes
- Name of output BAM file.

So something like:

```
$ mapFastq2BAM $GENOME_BWA \
    $DATA/samp_R1_.fastq.gz $DATA/samp_R1_.fastq.gz \
    samp.bam
```

and the output bam should be:

- Sorted
- be marked with Read Groups:
  - set SM, LB, PU to samp
- Have duplicates marked.

Do this for STAR also if you have time.

BEDTOOLS

I did not have time to cover bedtools but you absolutely should familiarize yourself with it. It can process BAM's, BED's, GFF, and VCF files. It basically is a genome region processor where the regions can be specific in one of those formats. It does what it calls *Genome arithmetic* including:

- intersect == Find overlapping intervals in various ways.
- window == Find overlapping intervals within a window around an interval.
- closest == Find the closest, potentially non-overlapping interval.
- coverage == Compute the coverage over defined intervals.
- map == Apply a function to a column for each overlapping interval.
- genomecov == Compute the coverage over an entire genome.

- merge == Combine overlapping/nearby intervals into a single interval.
- cluster == Cluster (but don't merge) overlapping/nearby intervals.

and tons of other stuff. If it is installed just type `bedtools` to see a full list of features and for sure checkout the man page at: (http://bedtools.readthedocs.org/en/latest/)

One could do a 4 hour course on just BEDTOOLS; however the `R` GenomeRange packages is also similarly featured so if you are more comfortable with `R` that may be easier to use.