Ryan Meng
10/18/17
BI 624

PCR Duplicate Problem Definition and Pseudo Code

Define the problem:

A PCR duplicate refers to any two reads that came from the same molecule of DNA and are a result of amplification during the library amplification process leading up to sequencing. PCR duplicates in theory should be located on the same strand and chromosome in addition to having the same alignment position and unique molecular identifier (UMI). The presence of PCR duplicates can greatly hinder RNA-seq experiments because they could give a false sense of abundance of a particular read that is not due to expression. Soft clipping, which is alignment adjustments at the 5' and 3' ends, makes PCR duplicate identification not as simple as looking for the above characteristics.

Strategies to address the problem:

There are two ways in which one can identify PCR duplicates. The first is to look for matching sequences (based on identical strand, chromosome, and position). The other is to label your sequences with UMIs cleverly so that you will be able to identify the PCR duplicates easier. In terms of developing code to identify PCR duplicates, it will be important to have the known list of UMIs available in addition to sorting the reads first by chromosome and position. In order to better understand the task at hand, I will generate two example SAM files, one that will serve as the input file with PCR duplicates and one that will serve as the output file without PCR duplicates. I will then develop a pseudo code structure, including important functions that will be featured in my script.

Pseudocode:

- Note: I will not be taking into account paired end data using this pseudocode.
- The very first step I would take in addressing this problem from a coding standpoint would be to sort the files by chromosome and position using SAMTools Sort (Convert existing SAM file to BAM file, sort using SAMTools, and then convert back to SAM)
- At this point I would have a SAM file that is sorted by both chromosome and position.
- My next step would involve creating a list of all of the known UMIs, so that when looping through each read, I could check to see if the UMI is one of the 96 known in the list. If

the UMI does not match the list of known UMIs, the read is discarded, if it does, the program continues.

- I would then loop through each read setting the UMI and starting position as key value pairs in a dictionary. Saving that read to a new output file.

- To determine the true starting position, I would look at the cigar string and determine if soft clipping has occurred. If it has, the starting position will have to be adjusted, if not the starting position will remain the same.

- I would then move onto the next read and check to see if it's UMI and starting position matched the previous read's UMI and starting position, again accounting for soft clipping.

- If the next read did match (starting position and UMI), that read would be discard (not written to the new file) and the loop would continue until hitting a unique read.

- Once it hit a unique read that was not a PCR duplicate, that read would then reset the dictionary becoming the new, and only, key value pair, again being written to the output file.

High Level Functions:
UMI_Check (Checks to see if UMI is in a known list of UMIs)
UMIist = [List of Known Umis]
If UMI inUMIllist::
    Continue
Else:
    Discard read

Example of a known UMI (from the list of 96): AACGCCAT
Example of a UMI that is discarded (not from the list of 96): TTTAGCTA

Position_adjustment(starting position, CIGAR string):
(Checks to see if the position for the alignment needs to be adjusted based on the CIGAR string)
Takes in the starting position based on the information from the same file
If soft clipping is present:
    Starting position = Starting position - soft clipping value
If soft clipping is not present:
    Starting position stays the same
Return true starting position (either adjusted or not)

Example of a read that needs to be adjusted: CIGAR String = 2S12M
Example of a read that does not need to be adjusted: CIGAR String= 14M

Dictionary(starting position, UMI)
Add read to dictionary (UMI and Starting Position)
If next read matches both UMI and starting position, after starting position is adjusted:
    Discard read, move onto next read
Else:
    Reset the dictionary with new key value pair containing starting position and UMI of current read
Return the read and output it to a new output file that will contain no PCR duplicates.

Reads in example SAM files are an example of which reads would be retained and which reads would be thrown out.

Additional Notes from Lecture(10/16/17)

Strand
Chromosome
Position
UMI matches

Example SAM file: Use UMIs, UMIs will not be in line

Headerlinemoreinfo:AAAAAAAA then rest of the sam file format, chromosome cigar string etc.

UMIs in the header

Colon UMI in the example SAM file

You need to check the headers, extract the UMI from the header

If UMI does not match known UMIs, toss it

For now write it for single end

Sort with SAMTools first, sort by chromosome and position