dudup part 1

October 21, 2018

1 Deduper part 1: planning

1.0.1 Purpose

Filter putative PCR duplicates in a SAM file, resulting in a new file with only one read per unique library fragment. We determine a fragment's identity from: chromosome + UMI + position + strand.

1.0.2 Main challenges

- Soft-clipping obscures "true" start position
  - TACTIC: use CIGAR string to correct for soft-clipping
- Start position is the left-most aligned nt, regardless of strand
  - TACTIC: use bitwise flag to determine strand for soft-clip correction
- samtools sort doesn't take soft-clipping into account, duplicates may not be adjacent in input file, file too big to store all entries in RAM
  - TACTIC: use a dict of UMIs that each points to a set of hashes to store essential information about prior PCR duplicates
- PCR dupe "hash" set may be too big to keep all in memory
  - TACTIC: reset/clear "hash" sets whenever we get to a new chromosome/group/whatever

1.0.3 Stretch goals (in order of priority)

- Handle paired-end reads
  - TACTIC: process read 1 and read 2 as before, only call PCR dupe if *both* are dupes of the same prior molecule. This basically just means incorporating both read 1 and read 2 data into the hash.
- Keep highest-quality read from each set up dupes
  - TACTIC: Instead of a hash *set*, use a hash *dict* with key = hash, val = (quality, read) pair. This is kind of lame, since we would then have to iterate over our sets to write reads at the end.

- UMI error correction
  - If UMI has n or fewer switches from nearest good UMI, correct to good UMI.

1.0.4 Algorithm overview

The main function will iterate over the lines in the SAM file. It will write all header lines to the output file. For each read, it will parse out the pertinent parts and determine strand (or unmapped). This will be used in conjunction with either the beginning or the end of the CIGAR string to "correct" the start position. The strand and corrected start position will be

```
In [25]: """ keeping track of necessary imports """
         from subprocess import call

         def getArgs():
             """ use argparse to return arguments from invocation:
                 -f, --file: (str) required arg, absolute file path
                 -p, --paired: (action='store_true') optional arg, designates file is paired e
                 -u, --umi: (str) optional arg, designates file containing the list of UMIs (u
                 -h, --help: optional arg, prints a USEFUL help message (see argparse docs)
                 """
             pass

         def sortSAM(unsorted_file_name):
             """ takes a SAM entry, calls samtools w/ params, returns the name of a sorted SAM
             """ TESTING: use unit test SAM file """
             # use 'call' to run samsort on unsorted_file_name w/ appropriate params
             #return(sorted_file_name)
             pass

         def buildDict(umi_list):
             """ take a list of UMIs, return a dict of (UMI: empty set) pairs """
             """ TESTING: with UMI file """
             #return(umi_dict)
             pass

         def getStrand(flag):
             """ take a SAM bitwise flag, and return int for validity/strand (0 = not mapped,
             """ TESTING: 165 -> 0, 20 -> 0, 769 -> 1, 784 -> 2"""
             if 4 & flag > 0: # checks for 'unmapped'
                 return(0)
             if 16 & flag == 0: # checks for 'reverse strand'
                 return(1)
             return(2)

         def unSoftClip(start_pos, cigar, strand):
             """ use strand and CIGAR strand to correct for soft clip and return "true" start
             """ TESTING: strand1+start100+10M -> 100, strand1+start100+10S90M -> 90, strand1+
             """          strand2+start100+100M -> 200, strand2+start100+10S90M -> 191, strand
```

```
    # subtranct non-maps at beginning of CIGAR string from start pos
    # if strand == 2: # correct for left-based strand 2 start posititon
        # add sum of CIGAR string to start pos
    #return(corr_pos)
    pass

def makeHash(strand,start):
    """ return unique float hash from strand, start position """
    #return(start+strand/10)
    pass

def parseRead(raw_read):
    """ parses , returns chromosome, strand, corrected start pos, UMI """
    """ TESTING: use reads from unit test input file """
    global sorted_file # why keep passing this around?
    # read line, split string into list
    # call getStrand on bitwise flag, get strand
    # call unSoftClip on start pos, get corrected start pos
    #return(chrom,strand,start_pos,umi)
    pass

def main():
    """ DO IT ALL """
    # call getArgs to get invocation info
    # call sortSAM to sort SAM file
    # read UMI file, get list of UMIs
    # call buildDict to build dict of (UMI : hash_set) pairs
    # open sorted read file
    # create + open results file
    # set curr_chr to 'START'
    # iterate over lines in file
        # if header, write to file
        # else parseRead
            # if not mapped, next line
            # else
                # if new chromosome, record new one and blank hash sets
                # if in UMI set, next line
                # else add to UMI set, write to file
    # close files
    pass

if __name__ == "__main__":
    main()
```

1.0.5 Testing

Unit test files in repo, individual function test files in function docstrings.