*AnCovMulti* Readme

*AnCovMulti* v0.1 written by *David D. Pollock* from Dec 11 2020 to Dec 28, 2020.

Based on seqmerAC [derived from *seqmer* (March 22, 2019)] and globals packages (begun Sept 16, 2018; last version Dec 7 2020), and derived from previous *AnCov* versions (begun Nov 11, 2020).

## Summary

*AnCovMulti* reads in multiple sequence read files, applies stringent filters to amplicon reads, and outputs regions that do not map to a reference sequence. It is aimed to build libraries of variants using kmers as a fast mapping tool, and to allow evaluation of the context in which variants are found. At this time, it has been set up and tested on fastq format sequences from Illumina, using 14mers. It is written in the language *Go* [or Golang] and takes about 6 seconds to run on a 2019 Mac Pro (3.5 GHz 8 core Intel Xeon W) for the test case of 5 Illumina sequence read files with filters already constructed.

## Future

The near-term aim is to automate the variant analysis, which will output a list of variants of interest along with their position in the reference, what they changed from and into in nucleotides and amino acids (if in protein, also listed). Further, the filenames will be translated into human readable identifiers, such as day of sampling and source, and the frequencies will be provided across the files. There will also be a similar file with more details, such as left and right primer source, R1 and R2 for Illumina files, the mean count of the variant, or min and max, or best estimate, the number of reads of that read type, and the number of reference reads of that read type. Finally, we will address the Minion reads and figure out reasonable parameter settings for them.

## Input and Output File Structure

*AnCovMulti* imports a number of standard packages as well as the custom *globals* and *seqmerAC* packages, which should be included in the release.

The globals package uses *control*, *mode*, and *factory* files to introduce environmental parameters, and these files must be included in the top-level folder. Parameters settings are in the form “*name = value*”, and if a required parameter is not in the mode or control file the program will barf. The program will also barf if the value cannot be converted to the expected type, but other than that there is no parameter checking. Parameters are set to their last set value, and lines are not read past a hashtag.

* The *control* file is the last parameter file read, and the parameter choices here will override any previous choices. Ideally it will be used to control most parameter changes among runs.
* The *mode* file is meant to set the type of analysis being done, and the structure of the files. A lot of parameters are set in it. They are organized with headers.
* The *factory* file is meant to set what the parameters, their type, and any bounds should be. It is meant to not be messed with and is not really implemented here.

Using the naming conventions in the supplied mode file,

* sequences are in *fastq* format with *.fq* ending, and are found in the *sequences* folder.
* Reference files are in the references folder and include:
  + the WuhanHu1 sequence in fasta format;
  + left\_primers.txt and right\_primers.txt (tab-delimited start and stop in WuhanHu1 reference, no header);
  + the WuhanWu1\_kpos.xls kmer, count, positions (tab-delimited, with header);
  + SequenceDayTranslation.txt in tab-delimited format;
  + filterseqs\_2020-10-14\_CO.xls list of filter names on different lines, not used but required;
  + and kcounts14\_WuhanHu1\_14Oct2020.xls which is not used and is the same as the kpos file without the positions.
* Query sequence filter files, when produced, are placed into the qfilter folder. They are later read from the same folder.
* All other output files are placed in the ACmulti\_output folder, including
  + Kpos.xls, the positions of the kmers in the reference sequence
    - This should be the same as the kpos file in the refefences folder
  + Primers\_out.xls, first position, end position, direction (left or right) in tab-delimited format with headers.
  + qnotk contains qnotk files for the filtered query sequences, meaning they are kmers found in the query sequences but not in the reference sequence
    - labeled left or right with klen and seqfile name, .xls at end, in tab-delimited format with headers.
    - Kmer sequence, count in filtered query sequences, first kmer location in WuhanHu1 ref, read count (from pcounts file), calculated positions in Wuhan reference sequence based on primer location
  + kandq contains kandq files for the filtered query sequences, meaning they are kmers found in both the query sequences and the reference sequence
    - labeled left or right with klen and seqfile name, .xls at end, in tab-delimited format with headers.
    - Kmer sequence, count in filtered query sequences, first kmer location in WuhanHu1 ref, read count (from pcounts file), calculated positions in Wuhan reference sequence based on primer location
    - Only printed if *bigoutput* parameter is true
  + Kcounts contains kcount files for the filtered query sequences, labeled left or right with klen and seqfile name, .xls at end, in tab-delimited format with headers.
    - Kmer sequence, count in filtered query sequences, first kmer location in WuhanHu1 ref, read count (from pcounts file), calculated positions in Wuhan reference sequence based on primer location
    - Contains all primers found in either qnotk or kandq
    - Only printed if *bigoutput* parameter is true
  + Pcounts contain primer location and count found in filtered query sequence file
    - This count is used to identify read counts

The provided folder structure contains the output expected with the sequence, references, and control file parameter settings. (bigoutput is false; qnotkmode is false but was run previously to produce the qfilters)

## Other Key Parameters

* Key variables **likely to change** in the control file
  + qminprint = 1
    - don’t print qmers less than this to kcount file for the query and related kmer sets
  + limitseqs = 50
    - just limit the number of seqs to look at for testing if you have a really big set of sequence files
  + runorient = right
    - either right or left, to control the orientation of analysis
  + qnotkmode= t
    - control run so don't repeat creation of qfilters
    - t, T, true, True, TRUE accepted
  + bigoutput= t
    - don't output big kcount and kandq files if we don't need them
* Key variables **not likely to change**
  + klen = 14
    - klen needs to match the reference inputs or a mess
  + left = left
  + right = right
    - left and right need to match the reference input or it will be a mess
  + printNs = false
    - do you really ever want to print kmers with Ns I them? Probably not.
  + right = right
    - left and right need to match the reference input or it will be a mess
* **Query filter boundary** parameters
  + minseq = 247
    - Illumina length minimum for 250 bp runs
  + minmatch = 50
    - the number of kmer matches in a read (matchcount) cannot be less than this
  + maxmisses = 42
    - the number of misses is possible - matchcount + 1
  + the firstmatch and lastmatch are the order of reference kmers found in the query sequence
  + first and second are whichever of these are first and second in order in the Wuhan reference sequence
  + Wudiffmax = 254
    - the difference between the second and first + klen cannot be more than this
  + maxfirsthit = 2
    - the read position of the Wuhan first kmer cannot be greater than this
  + minlasthit = 219
    - the read position of the Wuhan first kmer cannot be greater than this
  + posdiff is the lastposition - sequence length + klen -1
  + minposdiff = -16
    - posdiff can't be less than this
  + if left, the first kmer match must be a primer start
  + if right, the second kmer match + klen must be a primer end boundary
* The remaining variables are less important and all variables are explained in the mode file

## Code

Three folders of code are provided, *AnCovMulti*, *seqmerAC*, and *globals*, which contain, respectively, the files *AnCovMulti*.*go*, *seqmerAC.go*, and *globals.go*. The *AnCovMulti* folder also contains the binary *AnCovMulti* file, compiled on *macOS* *version 10.15.5*, *go1.14.4* downloaded June 1 2020. Information about the Go programming language[[1]](#footnote-1) can be found at <http://golang.org/>.

1. The Go Authors. The Go Programming Language Specification. <https://golang.org/ref/spec>, November 2016 [↑](#footnote-ref-1)