Pipeline for de novo calling of RNA editing from RNA-sequencing data

This pipeline is based on **Jacusa2**, a JAVA framework for identification of RNA editing sites [manual here](#) [github here](#)

Dependencies

- snakemake
- R/4.0.2
- jacusa2
- annovar

Input files

- config.yaml
  - metadata : tsv file with the following columns
    - sample: sample ids
    - bam_path: full path to a folder where all the BAMs for this experiment are located
    - library: strand specification for RNA-seq library
      - options: UNSTRANDED, RF-FIRSTSTRAND, FR-SECONDSTRAND
  - projectDir: full path to the working directory where you the intermediate files, logs, and final outputs of the pipeline to go
  - refDir: full path to a folder with the fasta format reference genome
  - editingRefDir: full path to a folder with any files you want to provide jacusa to filter editing calls based on (i.e. blacklisted genome bed file)
  - humandbDir: full path to folder housing annovar libraries

Our Pipeline Dissected

Jacusa 2

Some things to note about the Jacusa output format

Column 5 is the test-statistic of the likelihood ratio test for comparing two conditions I and II. Base call count vectors D-I and D-II are modelled with the Dirichlet-Multinomial distribution see manual 5.1.3.

Higher values of the test-statistic indicate a higher divergence of base call vectors between conditions.

When only one condition is provided, the command call-1 creates an *in silico* condition by using available reference information to replace non-reference base calls, creating synthetic base call vectors and applying the likelihood ratio test defined above.

Column 7 is the bases, where read counts for each base is provided in the following vector format: A, C, G, T.

Jacusa2 flags used in our pipeline explained

- -m flag filters positions with MAPQ < MIN-MAPQ for all conditions (default: 20)
- -P flag is for library type (first strand, second strand, or unstranded)
- -a flag is the artifact/feature filter
    - D: filters potential false positive variants adjacent to indels, adjacent to read start/end (6bp), adjacent to splice sites (6bp)
    - M: max allowed alleles per site (default 2)
    - Y: filter wrong variant calls within homopolymers (dafault length 7)
    - E: exclude sites contained in file
        - currently we supply the ENCODE blacklist as a bed file here, but potentially in the future this could be where a vcf could be provided to integrate WGS directly within the de novo caller for removing SNP calls
- -p flag specifies threads
- -s flag stores feature-filtered results in another file (= RESULT-FILE.filtered if no argument) or (= FILTERED-FILE)
- -F 1024 removes PCR duplicates

sample-level filtering

This step relies on the script *firstfiltering.R* to perform reformatting of the Jacusa2 output and only maintain editing sties that pass the following requirements:

- minimum read coverage at that site (default = 10)
- minimum alternate allele coverage (default = 2)
- ref allele is different than alt and multiallelic variants are dropped

sample merging and cohort-level filtering

This step relies on the script *secondfiltering.R* to aggregate editing sites across samples and generate two matrices: one with editing ratios and another with read coverage, whereby columns represent unique samples and rows represent unqiue editing sites. These matrices are then filtered, requiring:

- editing sites must validate across a percentage of samples (default = 0.5)
- minimum mean editing ratio for each site (default = 0.1)

Finally, the sites that pass these requirements are formatted into an input compatible for annovar.

annovar annotation

Using the *table_annovar.pl* script in annovar, editing sites are annotated by gene, gene region, repeat element, and common SNPs.

Editing sites are dropped based on the column specifying overlap with common variants. This filtered list of sites is then used to filter the editing ratio and coverage matrices created above.

Output Files

- **editing ratio matrix**
- **coverage matrix**
- **editing site annotations**

Downstream Analyses

Preparing known sites from REDIportal

This is an example of the code that I used to prepare a bed file of REDIportal editing sites to classify our called sites as known or novel.

```
##shell
cut -f1-11 TABLE1_hg38.txt > TABLE1_hg38_cut.txt

##R
hg38redi <- read.delim("/Users/winstoncuddleston/Downloads/TABLE1_hg38_cu
t.txt", sep = "\t", header = T)
library(dplyr)
REDI <- hg38redi %>% dplyr::filter(!grepl('_', hg38redi$Region))
write.table(REDI, file = "/Users/winstoncuddleston/Downloads/TABLE1_hg38.
txt", quote = F, sep = "\t", row.names = F, col.names = T)
REDIbed <- REDI[,c(1:5)]
REDIbed$Start <- REDIbed$Position-1
REDIbed <- REDIbed[,c(1,6,2,3,4,5)]
write.table(REDIbed, file = "/Users/winstoncuddleston/Downloads/hg38_REDI
.bed", quote = F, sep = "\t", row.names = F, col.names = F)
```