Schematic algorithmic pipeline:

Raw data

Sample.fastq files

Sam and Bam files

Unzip and concatenate files for each sample

MapFastq.py

CreateHitFile.py

Sample\_Hits.txt files

DrawORFNei.py

DrawChromosomeMap.py

ListORFsInSamples.py

Summary file

Chromosome map figure

Zoom figure

File formats

## Column description for all \_Hit.txt files (output of CreateHitFile.py):

**Chromosome:** chromosome name to which the read was map

**Up feature type:** nearest feature located up stream to the hit position, nan if the position is up stream to any given feature.

* All features are taken from chromosomal feature file downloaded from CGD.
* Feature are of types ORF, long\_terminal\_repeat, centromere etc.

**Up feature name:** feature name as given in the file (nan if there hit is upstream to any feature)

**Up gene name:** gene name if exists, nan if feature is not a gene

**Up feature dist**: the distance (number of nucleotides) between the hit position and the feature position

**Down feature type, Down feature name, Down gene name, Down feature dist:** same as the above for feature downstream of the hit.

IntergenicType: of type UpType-DownType where upType and DownType represents the strand of the nearest feature (is the nearest feature is either on the W/C strand) and also indicate if the hit is within an feature or only near to a feature, that is, UpType and DownType can be one of the following:

* "-" : no feature found up/down stream the hit
* "c": nearest feature found on 'c' strand
* "w": nearest feature found on 'w' strand
* "ORF(W)" : the hit is within a scope of an existing feature found on 'w' strand (in that case both Up feature dist and Down feature dist will be 0)
* "ORF(C)" : the hit is within a scope of an existing feature found on 'c' strand (in that case both Up feature dist and Down feature dist will be 0)

The intergenicType can be a combination of any of the following types with a dash separating the nearst upstream and downstream features.

**Hit position:** The position of the hit on the chromosome. If the hit was a merge of several neighbor reads, the position is of the most upstream read.

**Hit count:** number of reads found in this hit

## Column description for ORF summary files: (output of ListORFsInSamples.py)

**Feature name:** A list of all ORFs (only) found in the feature file from CGD

**Gene name:** Gene name if exists, nan otherwise

**Start coord:**  start coordinate of the feature in the chromosome

**Stop coord**: stop coordinate of the feature in the chromosome

For the following columns, for each sample exists a column starting with XXX (e.g. "Num Of Hits") and then the name of the hit file from which the data was extracted (e.g. PlatesAll\_C\_Hits)

* **Num Of Hits PlatesAll\_C\_Hits:** number of hits found in this ORF
* **Total reads PlatesAll\_C\_Hits:** sum of all reads of all hits found in this ORF
* **Max reads per hit PlatesAll\_C\_Hits**: the highest hit in this ORF
* **HitsPerORFPer100bpPlatesAll\_C\_Hits:** number of hits found in this orf, normalized to the ORF length.

For example for ORF A of length 200 with the 3 following hits in sample A: hit1 with 14 reads, heat2 with 100 reads and hit 3 with 1 read the Columns will be as following. Num Of Hits = 3; Total reads = 115; Max reads per hit = 100; HitsPerORFPer100bp = 1.5

**Unnamed: 7:**  empty field, please ignore

**SampleWithHits:** total number of samples with at least one hit with any number of read found in this ORF

**AvgHitsPerORFAllSamples:** the average of hits in ORF found across all samples files

**GeneLen:** gene length (=start – stop position)

**AvgHitsPerORFPer100bpAllSamples:** the average number of hits per 100bp in the ORF across all samples

Pipeline instructions:

1. Download all fastq file, and unzip.
2. For each sample merge all fastq file into one fastq file if necessary
   1. e.g. using cat command
   2. example for files :
      1. Tn-1\_S13\_L001\_R1\_001.fastq.gz
      2. …
      3. Tn-1\_S13\_L001\_R1\_004.fastq.gz
   3. Needs to run unzip Tn-1\_S13\_L001\_R1\_001.fastq.gz .. Tn-1\_S13\_L001\_R1\_004.fastq.gz and then cat Tn-1\_S13\_L001\_R1\_001.fastq Tn-1\_S13\_L001\_R1\_002.fastq.gz to obtain file Tn-1\_All.fastq
3. For each united sample file run script name MapFastq.py files
   1. This script takes three parameters:
      1. Input file (-i): this is the raw fastq files we just generated from the zip files we downloaded (fName henceforth)
      2. Output dir (-o): the output directory to which it saves all output files, uses script dir if not specified
   2. The program creates the following files:
      1. fName.Clean.fq: this file contains all reads which contained the Tn adaptor + tail sequence, after removing Tn sequence (created by cutadapt)
      2. fName.sam: human readable file with all reads and their mapped location on the genome (with respect to the index file given as input) created by bowtie.
      3. fName.bam: more compact binary file of the sam file
      4. fName.sorted.bam : sorted binary file, can be viewed using IGV tool
      5. fName.sorted.bam.bai
   3. dependencies:
      1. CInd: a name of index file created. Currently I use build 22 of 5314 strain for chromosom A (we compare only to chromosome A as the haploid is mostly chromosome A). The file can be found on the NAS under 'shared\yaelsilb\Code\dependencies'. Can create new files using bowtie index command (if wish to compare to a newer version). The file should be specified with only its prefix (e.g. 5314\_A22\_HapA )
4. Run CreateHitFile.py script
   1. This script creates the hit file from the sam file while:
      1. Traverse all .sam file in a directory
      2. Uniting close hits (±2bp, distance is given as a parameter, see below)
      3. Find for each merged hit its closest ORF in each strand
      4. Output all hits into fName\_Hits.txt file
   2. Notice, if hit file already exists for sam file the script does not re-create it, as it is very time consuming. For re-creating all files, either specify another output directory, or delete all previous hit files.
   3. Run example:
      1. python CreateHitFile.py
         1. python CreateHitFile.py –m 0 –i /home/SamDir –o /home/HitDir –q 5
   4. parameters (can be viewed using ‘python CreateHitFile.py –h’ command)
      1. –i (--InDir) the directory with all .sam file to parse (default is the script directory)
      2. –o (--OutDir): the directory to output all hit files
      3. –q (--MapQ): map quality of the reads to take into consideration, default is 10, that is all reads with MapQ< 10 are omitted due to ambiguous mapping.
      4. –m (--MergeDist): the maximal distance between two hits to merge (-m 0 won’t merge hits at all), default is 2
5. Run DrawChromosomeMap.py script
   1. This file creates the chromosome map for each hit file (per sample). The chromosome map is a map of all hits in each chromosome. If chromosome file already exists for a hit file, the script does not re-create the file. For map re-creation either specify a new directory as output directory or delete all previous files.
   2. Run example:
      1. python DrawChromosomeMap.py
         1. python DrawChromosomeMap.py –y 500 –i /home/HitDir –o /home/MapDir –q 5
   3. parameters (can be printed using ‘python CreateHitFile.py –h’ command
      1. –i (--InDir) the directory with all hit files to parse (default is the script directory)
      2. –o (--OutDir): the directory to output all chromosome map .png figures
      3. –y (--yLim): Maximum hits per position to display in figure (truncate y axis height – all hits will be of this maximum length), default is 5,000, must be bigger than 0. Can search the hit file to see maximal / average hit count
   4. Dependencies file (currently hard coded, should be moved to configuration file):
      1. ChrfName: fasta file containing of haploid strain to obtain chromosome length.
      2. CentromersfName: a file containing centromeres location as obtained from Judith (should be read from ChromosomeFeature file which also contains this information, probably more up to date)
6. For getting a zoom in on a specific ORF and its neighborhood run DrawORFNei.py
   1. This script gets a ORF name and a radios to zoom and create a figure of hits for a given sample
   2. Parameters:
      1. -i (--InFileName): Input hit file name, that is the \_Hit file previously created for a sample of choice.
      2. -o (--OutFileName): Output figure file name, the .png file to which the map will be saved
      3. -y (--yLim): Maximum hits per position to display in figure (truncate the hit height on y axis). Default is 5,000
      4. -z (--xZoom): # of nucleotides to display from each side of the ORF starting location. Default is 5,000
      5. -g (--GeneName): ORF Name as given in feature/Hit file.
   3. Dependencies file (currently hard coded, should be moved to configuration file):
      1. ChrFeaturefName: feature file name with all features and ORFs and their location on the genome.
7. For getting a summary of ORFs and their reads and hits across samples run ListORFsInSamples.py
   1. Run example: python ListORFsInSamples.py -i /home/dataanalysis/Seq2/GeneList/ -o ORFSumm.txt
      1. If –i is specified it is important to notice that the directory name ends with a “/” (in Linux) or “\” (in windows)
   2. Parameters:
      1. -i (--InDir): Input directory containing relevant hit files, should end with “\” or “/”, (default current directory). The script search for all \*Hits.txt in a directory and unite them into a summary file. If we want to create a summary file for a subset of the samples should copy the relevant samples hit file to a folder and run on that folder.
      2. -o (--OutFileName): Output file name, mandatory
   3. Dependencies file (currently hard coded, should be moved to configuration file):
      1. ChrFeaturefName: feature file name with all features and ORFs and their location on the genome.

Remarks:

* The raw data for Seq4 is saved on the NAS by Ella in the folder:
  + Ella\ CGEBMS89\_CGEBMP79A\_Ella-29554537
* The raw data for Seq2 is saved on the NAS server under:
  + Shared\yaelsilb\DataAnalysis\Seq2\Samples
* The raw data of Seq1 (though less informative) is saved on the NAS under
  + /Shared/yaelsilb/13 samples fastq
* The raw data of Seq3 (though less informative) is saved on the NAS under:
  + Shared\yaelsilb\DataAnalysis\Seq3\Samples
* All results file are given in the DataAnalysis folder under the folder GeneList for Hit files and ChrMap for the chromosome hit map figures. Though the file naming are different in the Seq 2.