1. **Scripts and files can be found on longleaf or git**

* /proj/cdjones\_lab/Meisha/msh6\_project/
* https://github.com/mam288/JonesLab.git

1. **Check to see if the following are installed on your system and install any that are not:**

* Python 3
  + pandas
  + time
  + datetime
  + argparse
  + pyfasta
  + os, sys
  + itertools
* samtools
* bwa
* freebayes

1. **Index reference genome fasta file**

* **Commands:**
  + bwa index [reference genome fasta file]
  + samtools faidx [reference genome fasta file]

1. **Create vcf and coverage files for parent fly lines:**

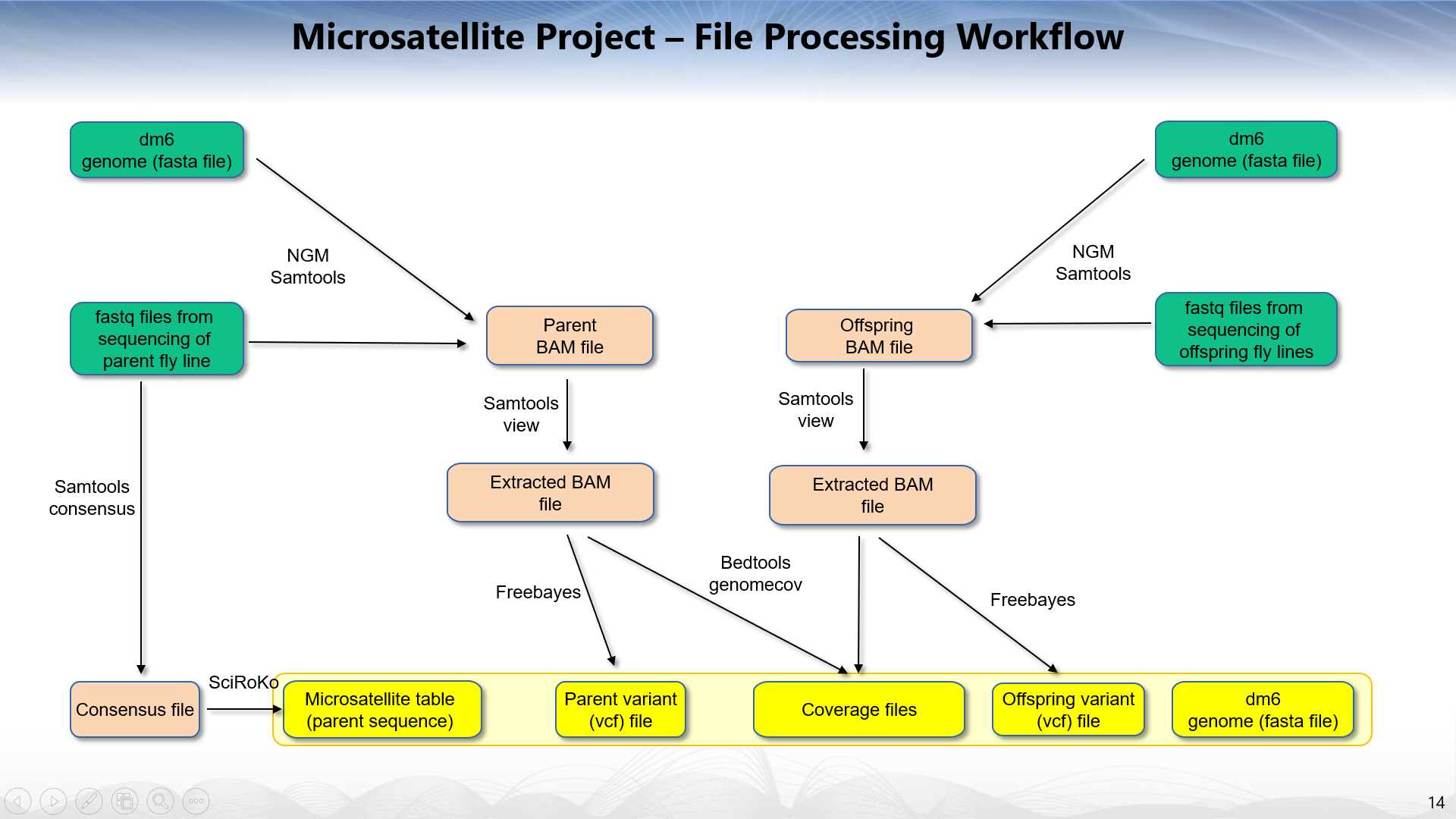
* **Command:**
  + sbatch create\_parent\_files.sh
* **Input Parameters:**
* **steps**
  + This indicates which steps in the processing pipeline should be performed.
  + Processing steps
    1. Align – align the fastq files using bwa
       - Required files: Paired read fastq files (2 per parent)
       - Creates a bam file with the name of the parent fly line saved as the read group
    2. Sort – sort the bam files created in previous step
       - Required files: bam files
       - Creates sorted bam files
    3. Extract– extract the microsatellite regions from the bam file, sort and index the resulting bam file
       - Required files: sorted bam file
       - Creates bam file with microsatellite regions extracted
    4. Sort and Index– sort and index the bam files created in previous step
       - Required files: bam files
       - Creates sorted and indexed bam files
    5. Create coverage files – create coverage files for the parent sequencing data
       - Required files: sorted, indexed, and extracted bam files
       - Creates a coverage file for each parent
  + Input string is a 5-character string with a y or – for each step indicating whether the step is to be performed (y) or not (-)
    - -yyyy would indicate that the fastq files have already been aligned but that the rest of the file processing steps should be performed
* **Set variables in create\_parent\_files.sh:**
  + ct – coverage threshold
    - Minimum amount of coverage required in a microsatellite region to use the information in that microsatellite
  + ad – allele depth threshold
    - Minimum allele depth (number of allele’s supporting that variant) required for a variant to use that variant’s information
  + mq – mapping quality threshold
    - Minimum mapping quality required for a variant to use that variant’s information
  + par1\_name – parent 1 name
    - name of the first parent fly line
  + par2\_name – parent 2 name
    - name of the second parent fly line
  + ref – reference name
    - name of the reference genome
* **Set file paths in create\_parent\_files.sh:**
  + If you are not using the sample data provided, set paths to the parent 1 and parent 2 fastq files and the reference genome fasta file
* **Naming fastq files:**
* Parent fastq files need to be named [parent name]\_R1.fastq and [parent name]\_R2.fastq

1. **Create microsatellite files for parent fly lines:**

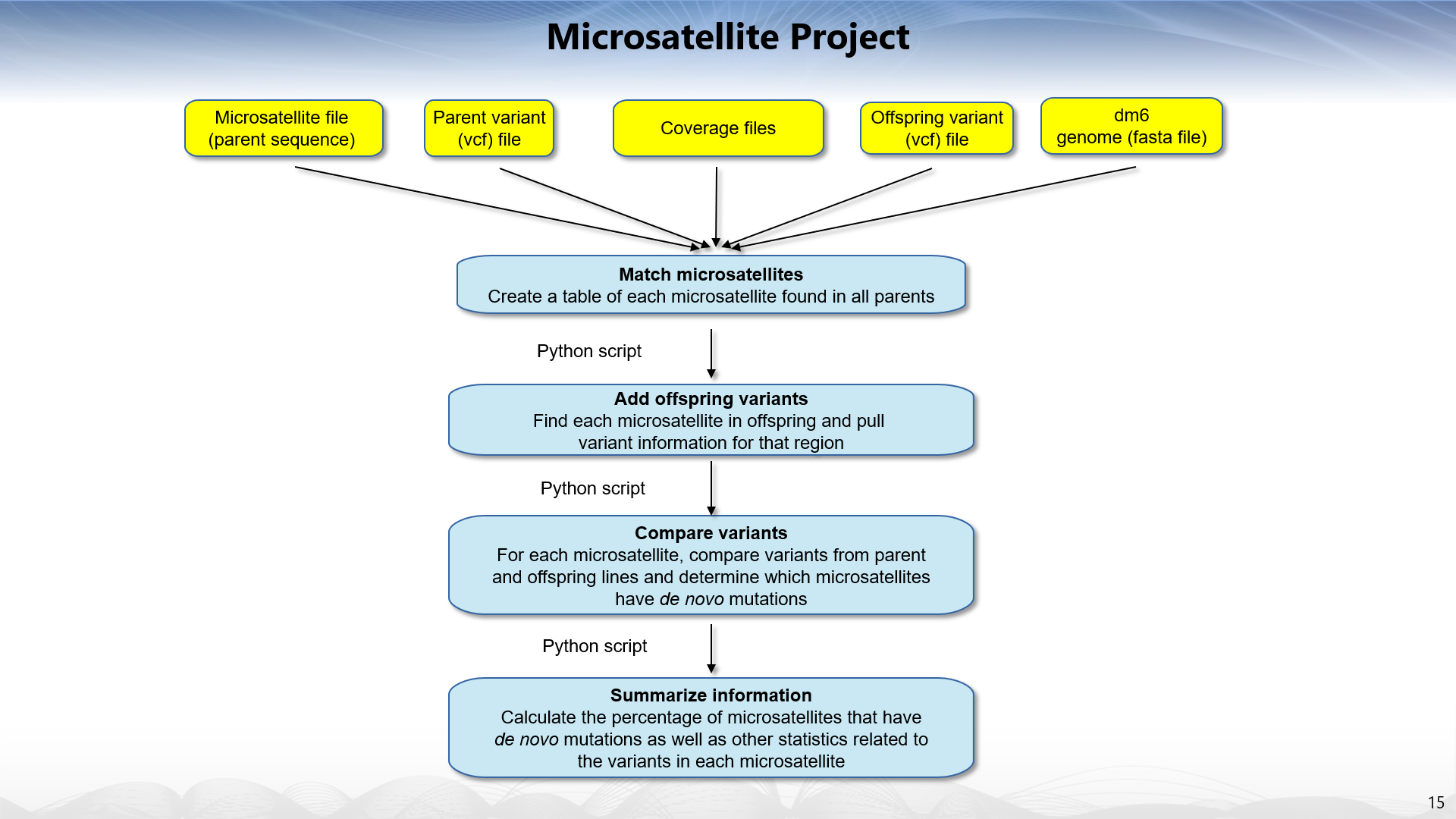
* **Download and install SciRoKo**
  + <https://kofler.or.at/bioinformatics/SciRoKo/>
* **Load the reference genome fasta file into SciRoKo and run search**
  + Use default parameters and save results as a tab delimited (td) file
  + Run: python make\_microsatellite\_regions\_file.py -td [reference genome td file]
* **Create consensus sequence files for each parent using mpileup**
  + Use the unextracted, sorted bam files created in step 4
  + Run the command: samtools mpileup -uf [reference genome fasta filename] [bam filename] > [output filename]
* **Load the consensus fasta files for each parent into SciRoKo and run search**
  + Use default parameters and save results as a tab delimited (td) file
  + Resulting file should be saved as [parent line name]\_microsats.td

1. **Process fastq files and analyze microsatellites:**

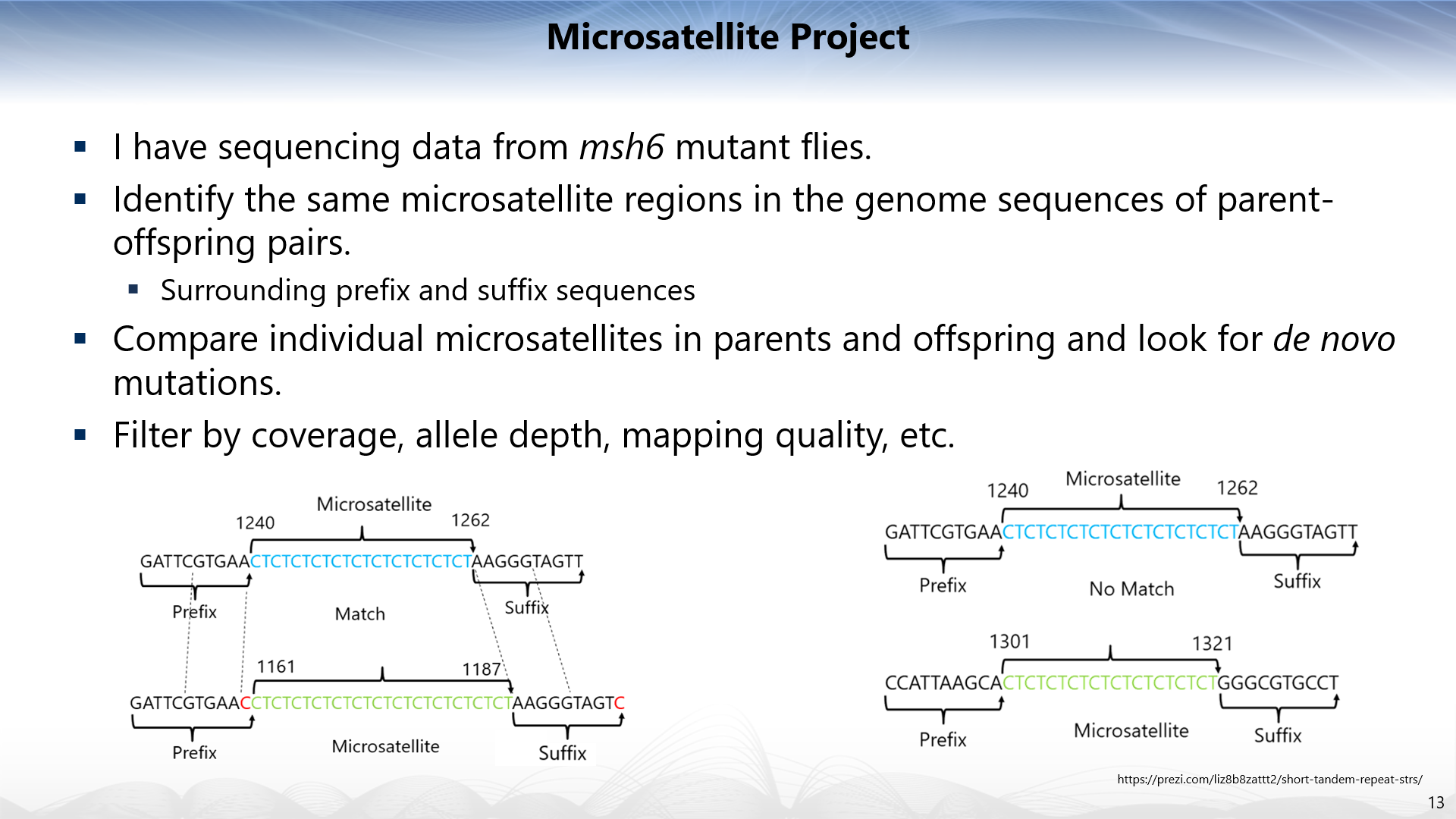
**Diagram of workflow pipeline**



**Diagram of microsatellite comparison pipeline**



* **Command:**
  + sbatch file\_processing\_pipeline\_microsatellite\_instability\_detection.sh [steps] [trunc] [save]
    - example: sbatch file\_processing\_pipeline\_microsatellite\_instability\_detection.sh yyyyyy 1000 True
* **Input Parameters:**
* **steps**
  + This indicates which steps in the processing pipeline should be performed.
  + Processing steps
    1. Align – align the fastq files using bwa
       - Required files: Paired read fastq files (2)
       - Creates a bam file with the name of the offspring fly line saved as the read group
    2. Sort – sort the bam file created in previous step
       - Required files: bam file
       - Creates a sorted bam file
    3. Extract and Sort – extract the microsatellite regions from the bam file, sort and index the resulting bam file
       - Required files: sorted bam file
       - Creates a sorted and indexed bam file consisting of only the reads that fully overlap identified microsatellite regions
    4. Create coverage files – create coverage files for the offspring sequencing data
       - Required files: sorted, indexed, and extracted bam file
       - Creates a coverage file from the offspring sequencing data
    5. Call Variants – Call variants on parent bam files and offspring bam file together using freebayes
       - Required files: sorted, indexed, and extracted bam files for each parent and offspring (each having an identifying read group)
       - Creates a vcf file with alignment information from all 3 bam files
    6. Split Variant Files – Splits vcf file from previous step into 3 separate vcf files based on read groups
       - Required files: vcf file with alignment information from all 3 bam files
       - Creates 3 separate vcf files each with alignment information from a specific read group
    7. Process Microsatellite Files – use vcf files and coverage files to pull variant information from microsatellite regions and compare parent and offspring variants to identify novel mutations
       - Required files: vcf files, coverage files, td files for microsatellites identified in parents, fasta file for reference genome (dm6 used in sample data)
       - Creates an excel file with a table containing variant and matching information for each microsatellite as well as summary data for the whole fly line
  + Input string is a 7-character string with a y or – for each step indicating whether the step is to be performed (y) or not (-)
    - -yyyyyy would indicate that the fastq files have already been aligned but that the rest of the file processing steps should be performed
* **trunc** 
  + To facilitate troubleshooting the microsatellite table can be truncated to a user-designated length before processing with the microsatellite\_compare\_parent\_offspring.py script
  + The input is an integer indicating how many rows of the microsatellite table to process
  + Set trunc to 0 if you do not want to truncate the tables
* **save** 
  + Boolean variable indicating whether to save the results to an excel file (True) or not (False)
* **Diagram of prefix-suffix comparison to identify matching microsatellites (same microsatellite in different fly lines)**



* + **Set variables in file\_processing\_pipeline\_microsatellite\_instability\_detection.sh:**
  + See variables set in step 4
  + Offspring\_name – name of offspring fly line
  + b - length of segment used for comparison of prefixes and suffixes during matching of microsatellite rows
    - b=5 means that the prefix/suffix sequence for one microsatellite is examined for the presence of a 5bp segment of the prefix/suffix of another microsatellite to determine if they are the same microsatellite
  + m - factor to multiply the buffer by when getting the prefix and suffix
    - m=3 means a prefix and suffix of 3x the buffer size is pulled for each microsatellite. With a b of 5bp each microsatellite will have prefix and suffix sequences of 15bp.
  + sd - how many bp apart the start distances can be to match the rows
  + covtype - type of coverage file - change to d if calling bedtools genome\_cov with -d option instead of -bg
  + **File Paths:**
  + If you are not using the sample data provided, set paths to the python script, offspring fastq files, reference genome fasta file, and microsatellite (.td) files for parent 1 and parent 2
* **fastq files:**
  + fastq files need to be named [offspring\_name]\_R1.fastq and [offspring\_name]\_R2.fastq

1. **Output:**

* **Microsatellite table columns:**
  + **Length** – length of the microsatellite in base pairs in the reference genome
  + **Motif** - repeat motif for the microsatellite
  + **Motif\_Standardised** – standardized motif for the microsatellite
  + **SSR\_End** – start coordinate of microsatellite in reference genome
  + **SSR\_Start** – stop coordinate of microsatellite in reference genome
  + **Seq\_Name** – name of the chromosome where the microsatellite is located
  + **Sequence** – sequence of the microsatellite
  + **dm6\_prefix** – prefix sequence for the microsatellite in the reference genome
  + **dm6\_seq** – reference genome sequence for the microsatellite
  + **dm6\_suffix** – suffix sequence for the microsatellite in the reference genome
  + **par1\_AD/par2\_AD/offspring\_AD** – allele depth for each variant found in the parents and offspring
  + **par1\_MQM/par2\_MQM/offspring\_MQM** – mapping quality for each variant found in the parents and offspring
  + **par1\_coverage\_avg/ par2\_coverage\_avg/offspring\_coverage\_avg** – average coverage over microsatellite region for parents and offspring
  + **par1\_genotype /par2\_genotype/offspring\_genotype** – genotype for each variant found in the parents and offspring
  + **par1\_length/par2\_length/offspring\_**length – length of microsatellite in parents and offspring for each microsatellite
  + **par1\_seq/par2\_seq/offspring\_seq** – parent and offspring sequences for each microsatellite
  + **par1\_start/par2\_start/offspring\_start –** start coordinate for microsatellite in parent and offspring sequences
  + **par1\_stop/par2\_stop/offspring\_stop** – stop coordinate for microsatellite in parent and offspring sequences
  + **par1\_variants/par2\_variants/offspring\_variants** – list of variants found in parents and offspring for each microsatellites
  + **par\_seq\_list** – list of the microsatellite sequences for the parent lines
  + **par\_var\_list** – list of variants found in the parent lines
  + **offspring\_match -**
    - True: each variant in the offspring is found in at least one parent
    - False: one or more variants in the offspring does not match either parent
    - I/C: incomplete coverage – cannot determine match status as coverage is below set threshold
    - I/AD: incomplete AD – cannot determine match status as AD is below set threshold
    - I/MQ: incomplete MQ – cannot determine match status as MQ is below set threshold
  + **offspring\_bp\_change** - list of the change in length from the reference sequence for each variant
  + **offspring\_TYPE** - list of the variant type for each variant
  + **offspring\_seq\_list** - list of the microsatellite sequences for the offspring line
  + **offspring\_seq\_par\_match** – list of match status for each variant (does the variant match one of the parents)
  + **offspring\_bp\_change\_mult\_motif** – list of the remainder left when the change in length is divided by the motif length. If the remainder is 0 the length changed by multiple of the motif length
* **Summary table columns**
* **Line** – the name of the fly line being analyzed
* **All Microsats** – total number of microsatellites identified across parents and offspring
* **Matched Microsats/ % Matched Microsats** – number and percentage of microsatellites that match between the parents and offspring
* **Not-matched Microsats/ % Not-matched Microsats** – number and percentage of microsatellites that do not match between the parents and offspring (novel mutations)
* **Not-matched indels/snps/complex/ %Not-matched indels/snps/complex** – number and percentage of variants in not-matched microsatellites that are insertions or deletions, snps, or complex
* **% Total Motif-multiple** – percentage of variants in match and not-matched microsatellites that change the microsatellite length by a multiple of the motif length
* **% Not-matched Motif-multiple** - percentage of variants in not-matched microsatellites that change the microsatellite length by a multiple of the motif length
* **Total indels/snps/complex/ %Total indels/snps/complex -** number of variants in matched and not-matched microsatellites that are insertions or deletions
* **Total variants** - total number of variants identified
* **avg\_cov –** average coverage over the microsatellite region
* **genotype\_non\_matched –** genotypes of the not-matched variants [homozygous reference, heterozygous, homozygous alternate]
* **genotype\_all** - genotypes of the match and not-matched variants [homozygous reference, heterozygous, homozygous alternate]
* **total\_microsatelltes** - total number of microsatellites identified across parents and offspring