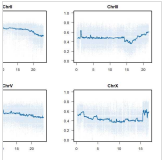


Jun 10, 2021

Pristionchus Whole Genome Sequence Analysis

erick.rios. [1], Ray L Hong[2]

[1]California State University Northridge; [2]California State University, Northridge

erick.rios. : erick.rios.492@my.csun.edu;

Ray L Hong: ray.hong@csun.edu

This protocol is published without a DOI.

NoviceCodingErick

Ray Hong
California State University, Northridge

ABSTRACT

With whole genome sequencing becoming more affordable and accessible every year, more diverse research groups will be capable of overcoming barriers to entry to utilize sequencing as a powerful tool for their research. The combination of mutagenesis experiments and sequencing of genomes of model organisms provide researchers a means to study mutations responsible for observed developmental phenotypes or behaviors. Sequencing of genomes tends to produce exorbitantly large data files that require specialized tools such as Burrows-Wheeler Aligner (BWA), Samtools, and Bcftools for processing. As the amount of sequencing in labs increases, the demand for bioinformatics skill also increases. Fortunately, existing computational workflows can minimize the amount of computer-knowledge required by individuals to handle whole genome reads. Nevertheless, these workflows require individuals to manually input each command from start to finish. Depending on the size of the files or the commands used, the time required to complete individual steps can take from a few seconds to several hours; with the entire workflow potentially requiring 8 or more hours on an average laptop computer. In this protocol, we provide a method to automate a workflow designed for mapping and finding candidate suppressor mutations in the nematode *Pristionchus pacificus*.

This protocol is a automated workflow adapted from "A simplified workflow for the analysis of whole-genome sequencing data from mutant lines with an application to the nematode *Pristionchus pacificus*" by Christian Rödelsperger. https://doi.org/10.1101/2020.11.12.379388

PROTOCOL CITATION

erick.rios. , Ray L Hong 2021. Pristionchus Whole Genome Sequence Analysis. **protocols.io** https://protocols.io/view/pristionchus-whole-genome-sequence-analysis-bu83nzyn

KEYWORDS

Whole Genome Sequence analysis, mutations, samtools, BWA, Automation, Bash, Script

IMAGE ATTRIBUTION

A snapshot of a mapping region output

CREATED

May 23, 2021

LAST MODIFIED

Jun 10, 2021

PROTOCOL INTEGER ID

50171

GUIDELINES

Make sure you have sufficient memory space (>30 Gb) and processing power.

MATERIALS TEXT

Uses BWA, samtools, and bcftools which are tools or packages that can be readily downloaded and installed. These tools are "linux/Mac based" to put it simply.

Windows users need to download an "emulator" of linux terminal such as "ubuntu 20.04" from sources such as the Microsoft store. (I use ubuntu 20.04.)

SAFETY WARNINGS

Can take time if you are a beginner

BEFORE STARTING

Read "A simplified workflow for the analysis of whole-genome sequencing data from mutant lines with an application to the nematode Pristionchus pacificus" by Christian Rödelsperger.

Installing tools 1h

1 **Rödelsperger's Whole Genome Sequence** workflow requires BWA, samtools, and bcftools which are tools or packages that can be readily downloaded and installed. These tools are "Linux/Mac based".

Windows users need to download an "emulator" of linux terminal such as "ubuntu 20.04" from sources such as the Microsoft store. (*e.g.* ubuntu 20.04.)

1.1 30m

For Windows 10:

1. Install “ubuntu 20.04 LTS” (free) from the Microsoft store
2. Once installed correctly
3. Open windows command shell and type “bash”; it may take a few moments for first time running.
4. $ sudo apt-get install bwa
5. $ sudo apt-get install samtools
6. $ sudo apt-get install bcftools
7. You are now free to continue.

For Mac users:

1. Open terminal/console
2. $ brew install bwa
3. $ brew install samtools
4. $ brew install bcftools
5. You are now free to continue.

1.2 Some computer systems may require admin or even root permissions in order to run the bash script. Use:

- sudo -i sudo -s

OR

- chmod -R 777 ./

Bash script can now be ran with full permissions. You must enter this request for each run.

Preparing the read files 4h

2 2h

Create a workspace folder and download the sequence files in the compressed format, **fq.gz**. Each genome should be at least two files from paired-end reads. BGI files may need to be concatenated into two files. Each compressed genome file could be up to 1 Gb.

3 5m

Decompress each file from the Terminal. On Mac OS10.12.x, open 'New Terminal at Folder' from your workspace. Each file should now end with **.gz,** ~3 Gb each.

- ls (list the files)
- gunzip -d filename (decompress)

3.1 If concatenating the files is necessary, perform the following and then run the alignment on combined_reads.fq. ~6 Gb each. 5m

- cat reads1.fq reads2.fq .... readsN.fq > combined_reads.fq

Example files used 5h

4 Download or move the reference sequences into the same folder. You will need an addition mutant genome if the mutations were generated on top of an existing mutant strain. The following files are needed/used for this example (**Rödelsperger's** workflow provides instructions for acquiring these files; With the only exception being the Automation_v1_3.sh bash script):

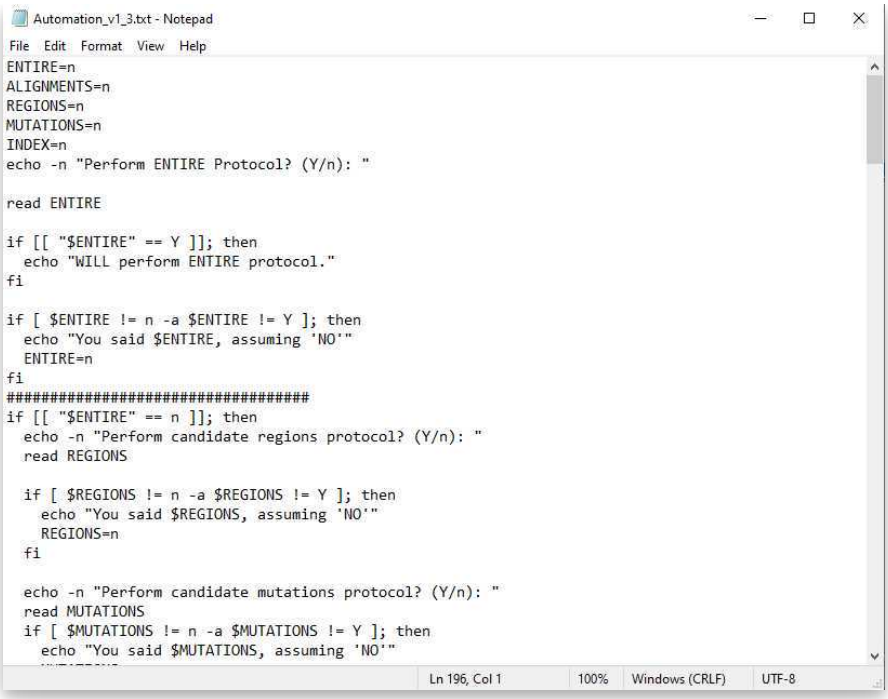
- El Paco Genome (Representative wild-type genome of *P. pacificus)*
- El_Paco_V3_gene_annotations_bcftools.gff3 (annotates the El Paco genome)
- SRR546092 (PS1843, Washington WILD mapping strain)
- ERR4367042 (*nhr-40* suppressor, SUP, pooled mapping panel)
- ERR4367073 (*nhr-40* suppressor mutant line (*tu515*))
- ERR4367101 (*nhr-40* mutant line (*tu505*))
- count_allele_frequencies.pl
- Automation_v1_3.sh **Automation_v1_3.sh**

Name	Date modified	Type	Size
Automation_v1_3.sh	6/1/2021 7:43 AM	SH File	7 KB
count_allele_frequencies.pl	11/11/2020 10:17 PM	PL File	1 KB
El_Paco_Genome.fa	12/20/2020 7:17 PM	FA File	157,366 KB
El_Paco_V3_gene_annotations_bcftools.gff3	6/1/2021 2:02 PM	GFF3 File	35,363 KB
ERR4367042_1.fastq	12/22/2020 1:03 PM	FASTQ File	7,939,061 KB
ERR4367042_2.fastq	12/22/2020 1:04 PM	FASTQ File	7,939,061 KB
ERR4367073_1.fastq	12/19/2020 8:38 PM	FASTQ File	6,189,925 KB
ERR4367073_2.fastq	12/19/2020 8:37 PM	FASTQ File	6,183,771 KB
ERR4367101_1.fastq	12/20/2020 12:17 AM	FASTQ File	3,474,603 KB
ERR4367101_2.fastq	12/20/2020 12:17 AM	FASTQ File	3,474,603 KB
SRR546092_1.fastq	12/22/2020 1:07 PM	FASTQ File	2,015,137 KB
SRR546092_2.fastq	12/22/2020 1:07 PM	FASTQ File	2,015,138 KB

4.1 Acquiring Automation_v1_3.sh

Copy the following Automation_v1_3.sh Bash script into a text document (The code is located in the appendix section)

4.2 Name the script as desired and save the **.txt** file.



Example image of the bash script pasted into Notepad; Take note of the bottom right ("Windows (CRLF)") for the next steps. If CRLF is present, then steps 4.3 and 4.4 will be needed. If the bottom right states "Unix (LF)", then 4.3 and 4.4 can be skipped.

4.3 Open up bash terminal and navigate to where the script is located.

4.4 Most likely Windows users will be required to convert the .txt from "Windows (CRLF)" to "Unix (LF)" because bash terminal can only utilize bash scripts in Unix (LF). CRLF and LF are "control characters" or "bytecode" and are different ways of reading files.

To convert from CRLF to LF enter in a Bash terminal (Enter the name of the script inplace of "FILENAME"):

fromdos FILENAME.txt

Note: You may need to use **sudo apt install tofrodos**

Before "fromdos"

After "fromdos"

4.5 Finally change the file extension from **.txt** to **.sh**

The script should now be usable.

Initiating the script

5 After the files have been gathered and prepared. To initiate the script, in a bash terminal enter:

./Automation_v1_3.sh

You will be prompted a series of questions regarding what jobs you wish the script to handle for you. After all questions have been answered the script will commence running **Rödelsperger's** workflow and may take a few hours to complete.

```
root@DESKTOP-C2ARIQB: /mnt/c/Users/Eric/Desktop/Dr.Hong/ProtocolWork
This message is shown once a day. To disable it please create the
/home/wasabi/.hushlogin file.
wasabi@DESKTOP-C2ARIQB:/mnt/c/Users/Eric/Desktop/Dr.Hong/ProtocolWork$ sudo -i sudo -s
[sudo] password for wasabi:
root@DESKTOP-C2ARIQB:~# cd /mnt/c/Users/Eric/Desktop/Dr.Hong/ProtocolWork
root@DESKTOP-C2ARIQB:/mnt/c/Users/Eric/Desktop/Dr.Hong/ProtocolWork# ls
Automation_v1_3.sh  ERR4367073_1.fastq  ERR4367101_2.fastq                        SRR546092_1.fastq
ERR4367042_1.fastq  ERR4367073_2.fastq  El_Paco_Genome.fa                         SRR546092_2.fastq
ERR4367042_2.fastq  ERR4367101_1.fastq  El_Paco_V3_gene_annotations_bcftools.gff3  count_allele_frequencies.pl
root@DESKTOP-C2ARIQB:/mnt/c/Users/Eric/Desktop/Dr.Hong/ProtocolWork# ./Automation_v1_3.sh
Perform ENTIRE Protocol? (Y/n): Y
WILL perform ENTIRE protocol.
Enter representative genome (include file extension): El_Paco_Genome.fa
Index your representative genome? (Y/n): Y
Do you have suppressor reads? (Y/n): Y
Enter suppressor read 1 (include file extension): ERR4367073_1.fastq
Enter suppressor read 2 (include file extension): ERR4367073_2.fastq
Enter mutant read 1 (include file extension): ERR4367101_1.fastq
Enter mutant read 2 (include file extension): ERR4367101_2.fastq
Candidate mutations; Enter annotation file (include file extension): El_Paco_V3_gene_annotations_bcftools.gff3
Enter wild read 1 (include file extension): SRR546092_1.fastq
Enter wild read 2 (include file extension): SRR546092_2.fastq
Enter map read 1 (include file extension): ERR4367042_1.fastq
Enter map read 2 (include file extension): ERR4367042_2.fastq
Candidate regions; Enter 'Source Code 1' perl script (include file extension): count_allele_frequencies.pl
How many CPU threads to dedicate for alignments? (Minimum: 1; Max: 5): 3
[bwa_index] Pack FASTA... 1.27 sec
[bwa_index] Construct BWT for the packed sequence...
[BWTIncCreate] textLength=317000792, availableWord=34305004
```

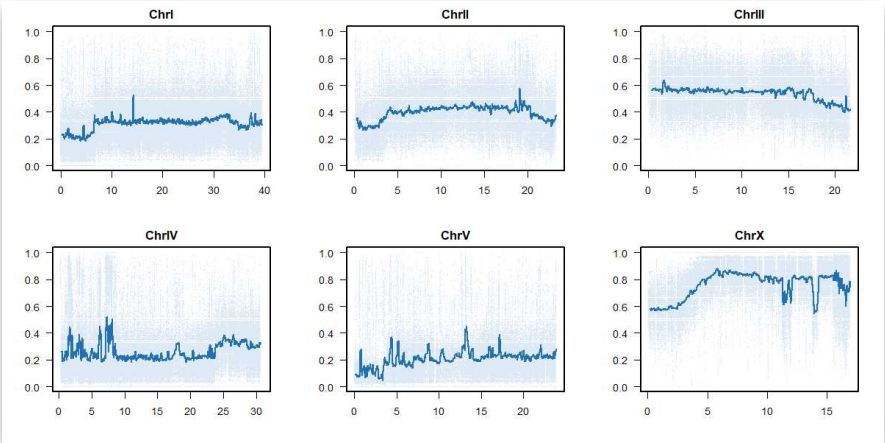
Results 1h

6 After script completion, these are the resulting files:

```
Automation_v1_3.sh     El_Paco_Genome.fa.sa                       SUPP_filtered1.vcf.gz
Candidate_Mut.txt      El_Paco_V3_gene_annotations_bcftools.gff3  SUPP_filtered1.vcf.gz.csi
ERR4367042_1.fastq     MAP.bam                                    SUPP_filtered2.vcf.gz
ERR4367042_2.fastq     MAP.bam.bai                                SUPP_filtered2_sorted.vcf.gz
ERR4367073_1.fastq     MUT.bam                                    SUPP_filtered2_sorted.vcf.gz.csi
ERR4367073_2.fastq     MUT.bam.bai                                SUPP_variants_annotated.vcf
ERR4367101_1.fastq     MUT.vcf.gz                                 WILD.bam
ERR4367101_2.fastq     MUT.vcf.gz.csi                             WILD.bam.bai
El_Paco_Genome.fa      SRR546092_1.fastq                          WILD.vcf
El_Paco_Genome.fa.amb  SRR546092_2.fastq                          count_allele_frequencies.pl
El_Paco_Genome.fa.ann  SUPP.bam                                   mapping_data.txt
El_Paco_Genome.fa.bwt  SUPP.bam.bai                               pileup_data.txt
El_Paco_Genome.fa.fai  SUPP.vcf.gz                                pos.txt
El_Paco_Genome.fa.pac  SUPP.vcf.gz.csi                            positions.txt
```

"mapping_data.txt" is the result of the "candidate regions" part of **Rödelsperger's** workflow to be taken to Rstudio. "Candidate_Mut.txt" is the **missense, stop_gained, synonymous, and INDEL** results from the "Candidate Mutations" part of **Rödelsperger's** workflow.

6.1 Processing "mapping_data.txt" through Rstudio yields (see **Rödelsperger's workflow** for the R scripts):



6.2 The script places "candidate mutations" into the "Candidate_Mut.txt" file:

This txt list can be imported into Excel or other spread sheet applications in a tab delineated format.

6.3 We recommend sequencing the genomes of multiple alleles in addition to chromosomal mapping to find likely phenotype-causing genes. To compare candidate mutation lists, use the following commands:

```
>perl extract_genes.pl Mut1_variants_annotated_all.vcf> set1.txt
>perl extract_genes.pl Mut2_variants_annotated_all.vcf> set2.txt
>perl compare_lists.pl set1.txt set2.txt
```

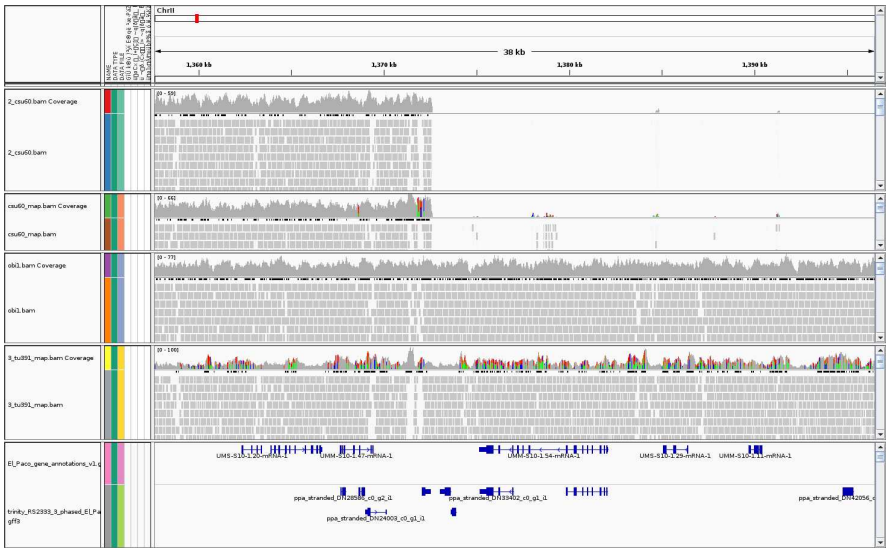
>perl extract_genes.pl Mut1_variants_annotated_all.vcf> set1.txt
>perl extract_genes.pl Mut2_variants_annotated_all.vcf> set2.txt
>perl compare_lists.pl set1.txt set2.txt

Visualizing Alignments 30m

7 Large insertions and deletions >1 kb will not be identified by the Candidate Mutation workflow. Instead, use the Integrative Genomics Viewer (IGV) to hone in on the chromosomal region you have identified by positional mapping.

You will need to load the following files to IGV (using a csu60 example):

- El Paco Genome (Representative genome of *P. pacificus)*
- El_Paco_V3_gene_annotations_bcftools.gff3 (annotates the El Paco genome)
- *Ppa-hsd-2(csu60)* whole genome sequence .bam file (*csu60* raw reads are on Figshare)
- Indexed *Ppa-hsd-2(csu60)* whole genome sequences (or, run >Samtools index csu60.bam to obtain the .bai file)



Note: Additional tracks show other genome alignments in addition to csu60 onto the El Paco reference genome for PS312.

Appendix Automation v1.3 bash script

8

```
ENTIRE=n
ALIGNMENTS=n
REGIONS=n
MUTATIONS=n
INDEX=n
echo -n "Perform ENTIRE Protocol? (Y/n): "

read ENTIRE

if [[ "$ENTIRE" == Y ]]; then
  echo "WILL perform ENTIRE protocol."
fi

if [ $ENTIRE != n -a $ENTIRE != Y ]; then
  echo "You said $ENTIRE, assuming 'NO'"
  ENTIRE=n
fi
####################################
if [[ "$ENTIRE" == n ]]; then
  echo -n "Perform candidate regions protocol? (Y/n): "
  read REGIONS

  if [ $REGIONS != n -a $REGIONS != Y ]; then
    echo "You said $REGIONS, assuming 'NO'"
    REGIONS=n
  fi

  echo -n "Perform candidate mutations protocol? (Y/n): "
  read MUTATIONS
  if [ $MUTATIONS != n -a $MUTATIONS != Y ]; then
    echo "You said $MUTATIONS, assuming 'NO'"
    MUTATIONS=n
  fi
fi
####################################
if [ $ENTIRE == Y -o $ALIGNMENTS == Y -o $REGIONS == Y -o $MUTATIONS == Y ]; then
  echo -n "Enter representative genome (include file extension): "
```

```
  read GENOME
  echo -n "Index your representative genome? (Y/n): "
  read INDEX
  if [ $INDEX != Y -a $INDEX != n ]; then
    echo "You said $INDEX, assuming 'NO'"
    echo "Will NOT index genome."
    INDEX=n
  fi
fi
####################################
if [ $ENTIRE == Y -o $MUTATIONS == Y -o $ALIGNMENTS == Y ]; then
  echo -n "Do you have suppressor reads? (Y/n)"
  read SUPPREAD
  if [ $SUPPREAD == Y ]; then
    echo -n "Enter suppressor read 1 (include file extension): "
    read SUPP_1
    echo -n "Enter suppressor read 2 (include file extension): "
    read SUPP_2
  fi
  if [ "$SUPPREAD" != Y -a $SUPPREAD != n ]; then
    echo "No suppressor reads. Only mutant reads will be used."
    SUPPREAD=n
  fi
  echo -n "Enter mutant read 1 (include file extension): "
  read MUT_1
  echo -n "Enter mutant read 2 (include file extension): "
  read MUT_2
fi
if [ "$ENTIRE" == Y -o $MUTATIONS == Y ]; then
  echo -n "Candidate mutations; Enter annotation file (include file extension): "
  read ANNOTATION
fi
if [ "$ENTIRE" == Y -o $REGIONS == Y -o $ALIGNMENTS == Y ]; then
  echo -n "Enter wild read 1 (include file extension): "
  read WILD_1
  echo -n "Enter wild read 2 (include file extension): "
  read WILD_2

  echo -n "Enter map read 1 (include file extension): "
  read MAP_1
  echo -n "Enter map read 2 (include file extension): "
  read MAP_2

  if [ "$ENTIRE" == Y -o $REGIONS == Y ]; then
    echo -n "Candidate regions; Enter 'Source Code 1' perl script (include file extension): "
    read PERL
  fi
fi
####################################
if [ $ENTIRE == Y -o $ALIGNMENTS == Y -o $REGIONS == Y -o $MUTATIONS == Y ]; then
  echo -n "How many CPU threads to dedicate for alignments? (Minimum: 1; Max: 5): "
  read THREAD
fi
##################################################################################
#####
if [[ "$INDEX" == Y ]];then
  bwa index $GENOME
fi
######################## REGIONS
#######################################################
if [ "$ENTIRE" == Y -o $REGIONS == Y ]; then
```

```
  bwa mem -t $THREAD -o WILD.sam $GENOME $WILD_1 $WILD_2
  bwa mem -t $THREAD -o MAP.sam $GENOME $MAP_1 $MAP_2

  samtools view -S -b MAP.sam > MAP_unsorted.bam
  samtools view -S -b WILD.sam > WILD_unsorted.bam

  samtools sort -o MAP.bam MAP_unsorted.bam
  samtools sort -o WILD.bam WILD_unsorted.bam

  samtools index MAP.bam
  samtools index WILD.bam

  rm MAP_unsorted.bam MAP.sam
  rm WILD_unsorted.bam WILD.sam

  bcftools mpileup -Ou -f $GENOME WILD.bam | bcftools call -mv -Ov -o WILD.vcf

  awk '{if($6>100) print}' WILD.vcf | grep -v INDEL |awk '{print $1 "\t" $2}' > positions.txt

  samtools mpileup -f $GENOME -l positions.txt MAP.bam > pileup_data.txt

  perl $PERL pileup_data.txt > mapping_data.txt
  echo "Candidate regions finished; Look for 'mapping_data.txt' file."
fi
############################## MUTATIONS
###########################################
if [ $SUPPREAD == Y ]; then
  bwa mem -t $THREAD -o SUPP.sam $GENOME $SUPP_1 $SUPP_2
  bwa mem -t $THREAD -o MUT.sam $GENOME $MUT_1 $MUT_2

  samtools view -S -b SUPP.sam > SUPP_unsorted.bam
  samtools view -S -b MUT.sam > MUT_unsorted.bam

  samtools sort -o SUPP.bam SUPP_unsorted.bam
  samtools sort -o MUT.bam MUT_unsorted.bam

  samtools index SUPP.bam
  samtools index MUT.bam

  rm SUPP_unsorted.bam SUPP.sam
  rm MUT_unsorted.bam MUT.sam

  bcftools mpileup -Ou -f $GENOME SUPP.bam | bcftools call -mv -Ou | bcftools view -i '%QUAL>=20' -Oz > SUPP.vcf.gz

  bcftools mpileup -Ou -f $GENOME MUT.bam | bcftools call -mv -Ou | bcftools view -i '%QUAL>=20' -Oz > MUT.vcf.gz

  bcftools index SUPP.vcf.gz

  bcftools index MUT.vcf.gz

  bcftools filter -O z SUPP.vcf.gz > SUPP_filtered1.vcf.gz

  bcftools index SUPP_filtered1.vcf.gz

  bcftools isec -C SUPP_filtered1.vcf.gz MUT.vcf.gz > pos.txt

  bcftools filter -O z -R pos.txt SUPP_filtered1.vcf.gz > SUPP_filtered2.vcf.gz

  bcftools sort -O z SUPP_filtered2.vcf.gz > SUPP_filtered2_sorted.vcf.gz

  bcftools index SUPP_filtered2_sorted.vcf.gz
```

```
  bcftools csq -p s -f $GENOME -g $ANNOTATION SUPP_filtered2_sorted.vcf.gz >   SUPP_variants_annotated.vcf

  grep missense SUPP_variants_annotated.vcf > Candidate_Mut.txt
  grep stop_gained SUPP_variants_annotated.vcf >> Candidate_Mut.txt
  grep synonymous SUPP_variants_annotated.vcf >> Candidate_Mut.txt
  grep INDEL SUPP_variants_annotated.vcf >> Candidate_Mut.txt
  echo "Candidate mutations protocol finished; Look for 'Candidate_Mut.txt' file"
fi

if [ $SUPPREAD == n ]; then
  bwa mem -t $THREAD -o MUT.sam $GENOME $MUT_1 $MUT_2
  samtools view -S -b MUT.sam > MUT_unsorted.bam
  samtools sort -o MUT.bam MUT_unsorted.bam
  samtools index MUT.bam
  rm MUT_unsorted.bam MUT.sam
  bcftools mpileup -Ou -f $GENOME MUT.bam | bcftools call -mv -Ou | bcftools view -i '%QUAL>=20' -Oz > MUT.vcf.gz
  bcftools index MUT.vcf.gz
  bcftools csq -p s -f $GENOME -g $ANNOTATION MUT.vcf.gz > MUT.vcf
  grep missense MUT.vcf > Candidate_Mut.txt
  grep stop_gained MUT.vcf >> Candidate_Mut.txt
  grep synonymous MUT.vcf >> Candidate_Mut.txt
  grep INDEL MUT.vcf >> Candidate_Mut.txt
fi
##################################################################################
#####
if [[ $ENTIRE == n ]]; then

  if [[ "$REGIONS" == n ]]; then
    echo "Candidate regions protocol was not performed."
  fi
  if [[ "$MUTATIONS" == n ]]; then
    echo "Candidate mutations protocol was not performed."
  fi
fi
if [ $ENTIRE == n -a $REGIONS == n -a $MUTATIONS == n -a $INDEX == n -a $ALIGNMENTS == n ];then
 echo "Nothing was accomplished this day."
else
 echo "Something was accomplished!"
fi
```

Appendix Compare mutation lists 15m

9 Perl scripts for comparing related mutant genomes. 15m

extract_genes.pl

```
#!/usr/bin/perl

use strict;

open(IN, $ARGV[0] );
while(){
   if (/BCSQ=/){
 s/.+BCSQ=//g;  ## remove first part of the line
 s/\s+.+//g;   ## remove second part of the line
 my ($type, $id) = split( /\|/, $_ );  ## split by "|" symbol
 unless($type eq "synonymous" || $type eq "intron" ){
    print "$id\n";
 }
```

```
    }

  }
  close(IN);
```

9.1 compare_lists.pl

```
#!/usr/bin/perl

use strict;

my $usage = "
checks for common entries in the lists
> program list1.txt list2.txt [-a/-n]\n ";



if( $#ARGV == -1 ){ print $usage; exit 0;}

my @list1 = `less $ARGV[0]`;
my $list2;
my $printCom = 1;
my $all = 0;
$printCom = 0  if( $ARGV[2] eq "-n" );
$all = 1  if( $ARGV[2] eq "-a" );

open( IN , $ARGV[1]);
while(my $l = ){

 $l =~ s/\s+//g;
 $list2->{ $l } = 1;
}

my $commons = 0;
$list2 =~ s/\t/ /g;


foreach my $element (@list1){
 chomp $element;
 $element =  $element;
 #print "$element\n";
 my $found = 0;
 $found = 1 if( exists $list2 ->{$element} ) ;
 unless( $all ){
    if( exists $list2 ->{$element} ){
  $commons++;
  print $element."\n" if( $printCom );
    }
    else{
  print $element."\n"  unless( $printCom );
    }
 }
 if( $all ){
  print "$element\t$found\n";
 }
}
print STDERR "non-coms $printCom\n " unless( $printCom );
print STDERR "Commons: $commons\n";
```