**PROJECT’s full code**

**#Process of snp.tab files from each snippy output directory in R.**

**# Set up working directory#**

setwd("path\_to\_snp.tab\_files")

**#read library readr#**

library(readr)

**#Due to large number of snp.tab files, we have processed initially each project accession number that we have retrieved from ENA separately.**

**#READ & PROCESS SNP.TAB FILES FROM DATASET\_1#**

**#create a vector/list from all files#**

files <- list.files("path\_to\_snp.tab\_files of each project accession\_number”)

**#annotated with dataset extension#**

files <- paste("additional\_path\_to\_dataset1\_folder/", files, sep = "")

**#read data frames -Read snp.tab files from dataset1#**

allFiles.list <- lapply(files, read\_delim, delim = '\t', col\_names = TRUE)

**#combine all data frames by combining the rows, remove duplicate headers#**

df <- do.call("rbind", allFiles.list)

**#remove indels**

df <- df[ grep("del", df$TYPE, invert = TRUE) , ]

df <- df[ grep("ins", df$TYPE, invert = TRUE) , ]

df <- df[ grep("ins", df$EFFECT, invert = TRUE) , ]

df <- df[ grep("del", df$EFFECT, invert = TRUE) , ]

**#create new dataframe with the columns you need- here we have kept column position, ref &alt ,Ftype, effect, locus\_tag, gene and product#**

df = df[,c(2,4,5,7,11,12,13,14)]

**#remove vector/list after reading the initial dataframes#**

rm(allFiles.list,files)

**#process dataset1#**

**# load dplyr library**

library(dplyr)

**# group  per position & locustag#**

**# add an extra column, summarising the cumulative number of nucleotide variations in specific position #**

dataset1 <- df %>% group\_by(POS, LOCUS\_TAG) %>%  summarise(SNPs = n())

**#remove interim dataframe#**

rm(df)

**# Read & process the rest of datasets in the same way#**

**# We have read & processed the rest of the 16 datasets in the same way#**

**#Further analysis of datasets(trim/bind/groupby)#**

**#trim datasets# for example for dataset1 and dataset2 see below#**

**# keep column position and summary of SNPs**

data\_a <- dataset1[,c(1,3)]

**# keep column position and summary of SNPs**

data\_b <- dataset2[,c(1,3)]

**#Do the same for rest of the datasets**

**# remove all non –trimmed datasets**

rm(dataset1, dataset2, …)

**#rbind trimmed datasets#**

data <- rbind(data\_a, data\_b, …)

**# remove trimmed datasets#**

rm(data\_a, data\_b, …)

**#combined dataframes-group by POS and summarise nucleotide variants#**

library(dplyr)

data <- data %>% group\_by(POS) %>%

  summarise(SNPs = sum(SNPs))

**#load intersected file. The intersected file takes into account all the overlapping genomic positions in H37Rv and was created in bash shell with the command bedtools intersect#**

**# read & view intersected file #**

intersected <- read\_delim("~/TB\_bioinformatics/data/intersected1","\t", escape\_double = FALSE, trim\_ws = TRUE)

View(intersected)

**#remove the duplicate and empty columns from intersected file#**

intersected <- intersected[,c(1,3,5,6,7,8,9,10)]

**#name columns in dataframe intersected#**

names(intersected) = c("GENOME","POS","START","STOP", "FTYPE", "LOCUS\_TAG", "GENE","PRODUCT")

**#create a dataframe with the same number of positions as H37Rv#**

wholegenome <- data.frame(POS = seq(1,4411532))

**#merge intersected with wholegenome dataframe by the column position#**

intersected <- merge(intersected, wholegenome, by = "POS", all = TRUE)

**#remove wholegenome dataframe#**

rm(wholegenome)

**# Remove NA values from all the columns of intersected dataframe (in this case it’s the non coding positions) & replace with appropriate names#**

intersected$GENOME[[is.na](http://is.na/)(intersected$GENOME)] <- "NC\_000962"

intersected$START[[is.na](http://is.na/)(intersected$START)] <- "NON\_CODING"

intersected$STOP[[is.na](http://is.na/)(intersected$STOP)] <- "NON\_CODING"

intersected$FTYPE[[is.na](http://is.na/)(intersected$FTYPE)] <- "NON\_CODING"

intersected$LOCUS\_TAG[[is.na](http://is.na/)(intersected$LOCUS\_TAG)] <- "NON\_CODING"

intersected$GENE[[is.na](http://is.na/)(intersected$GENE)] <- "NON\_CODING"

intersected$PRODUCT[[is.na](http://is.na/)(intersected$PRODUCT)] <- "NON\_CODING"

**#Merge intersected dataframe with data dataframe by column position#**

intersected\_final <- merge(intersected, data, by = "POS", all = TRUE)

**# replace NA in SNP column with zero#**

intersected\_final$SNPs[[is.na](http://is.na/" \t "_blank)(intersected\_final$SNPs)] <- 0

**#remove intersected dataframe#**

rm(intersected)

**#create final dataframe, but exclude miscellaneous RNAs, mobile elements, repeat regions, transposases and non coding regions#**

df\_NC <- intersected[!intersected$GENE=="NON\_CODING",]

df\_NO <- df\_NC[!df\_NC$FTYPE=="tRNA"

&!df\_NC$FTYPE=="repeat\_region"&!df\_NC$FTYPE=="mobile\_element"&!df\_NC$FTYPE=="ncRNA"&!df\_NC$FTYPE=="misc\_RNA"&!df\_NC$FTYPE=="rRNA&!df\_NC$PRODUCT=="transposase"&!df\_NC$PRODUCT=="IS2-like transposase"&!df\_NC$PRODUCT=="transposase fusion protein",]

**# Remove PE-PPE genes**

df\_NO <- df\_NO[ grep("PPE", df\_NO$PRODUCT, invert = TRUE) , ]

df\_NO <- df\_NO[ grep("PE", df\_NO$PRODUCT, invert = TRUE) , ]

df\_NO <- df\_NO[ grep("PE\_PGRS", df\_NO$PRODUCT, invert = TRUE) , ]

**#GROUP by Locus\_tag to sum the snps per locus tag#**

df\_final <- df\_NO %>% group\_by(LOCUS\_TAG, GENE, PRODUCT,START, STOP) %>% # group per LOCUS\_TAG #

summarise(SNPs = sum(SNPs))

**#Normalise by gene length#**

**#subset start –stop column#**

subfinal<- df\_final[,c(4,5)]

**#make start and stop column numeric#**

subfinal$START <- as.numeric(as.character(subfinal$START))

subfinal$STOP <- as.numeric(as.character(subfinal$STOP))

**##subtract stop-start column to find every gene’s length#**

subfinal$LENGTH <- (subfinal$STOP-subfinal$START)

**#merge subfinal with with df\_final dataframe#**

subfinal2 <- merge(df\_final, subfinal, by = "START", all = TRUE)

**#remove 2nd stop column#**

subfinal3 <- subfinal2[,c(1,2,3,4,5,6,8)]

**#rename columns appropriately#**

names(subfinal3) <- c("START","LOCUS\_TAG","GENE","PRODUCT","STOP","SNP","LENGTH")

**#divide number of nucleotide variants/gene’s length to normalise value by gene’s length#**

final <- transform(subfinal3, RATIO= SNP / LENGTH)

**# sort by lowest mutation/length ratio**

final\_sorted <- final[order(final$RATIO),]

**#Summary statistics & 5th-95th quantile#**

summary(final\_sorted)

quantile(final\_sorted$RATE, probs = seq(0, 1, by= 0.05))

**#Validation of results#**

**#The R script below ran on a linux server (in R console), because computationally was very difficult to be done in R studio. The files used for the validation were the files contained in each snippy directory containing the depth in each genomic position. Each file was manipulated in bash shell and contained 1 column(depth column). The first file in the directory used contained 2 columns (position and depth)#**

**#set working directory#**

setwd("/path\_to\_files\_containing\_depth&position")

**#create a list vector with file names#**

filenames <- list.files(full.names=TRUE)

**#read all files from list#**

allFiles.list <- lapply(filenames, read.csv,header = TRUE)

**#column bind all dataframes. Note that the first dataframe on the list has 2 columns (position &depth). The rest have only one column(depth). The df dataframe will have the number of lines of H37Rv and 8536 columns(1st column with have the genomic positions)**

df <- do.call("cbind", allFiles.list)

**#subset each gene on 5th percentile e.g csb gene from the large df file #**

csb <- subset(df, POS > 967896 & POS < 968306)

**#check the sum of genomic positions equal or above depth of 10- keep column 1(position) stable as first column#**

csb$above10 =apply(csb[,-1], 1, function(x) sum(abs(x) >= 10))

**#keep datframe without the column with the sum >10 depth#**

csb\_1 <- csb[,c(1:8536)]

**#check the sum of genomic positions below depth of 10- keep column 1(position) stable as first column#**

csb\_1$below10 = apply(csb\_1[,-1], 1, function(x) sum(abs(x) < 10))

**#rename csb to csb\_new-it will contain >10 depth column#**

csb\_new <-csb[,c(1,8537)]

**#rename csb\_1 dataframe to csb\_1\_new –will contain below 10 depth column#**

csb\_1\_new <- csb\_1[,c(1,8537)]

**#merge csb\_new & csb\_new\_1 dataframes to create dataframe with >10 and <10 depth column#**

csb\_merge <- merge(csb\_1\_new,csb\_new, by="POS", all=TRUE)

**#remove files you don’t need#**

rm(csb\_1,csb,csb\_new , csb\_1\_new )

write.csv(csb\_merge,"/reads/tsv\_files/all\_final/test/csb\_merge", row.names = FALSE)