Final Log file for 2Rb Inversion Signature

- Tejashwini Alalamath (2021)

---

1. Raw Data Preparation

(All processing done on unless otherwise mentioned)

The raw data was sourced from -

Indch - (Done by Aditi) :

```
scp
group_sshubha01@192.168.1.151:/home/internal/NGS/group_sshubha01/aditi/ref_genomes/
UCI/sorted_bt2_chr2UCI_IndChIllumina.bam .
```

Ste2* - (Done by Subha ma’am) *:*

fastq-dump --split-files SRR1168951 &

UCI1 - (Done by Saurabh)*:*

fastq-dump --split-files SRR11672501

UCI4 - (Done by Tejashwini)*:*

fastq-dump --split-files SRR11672504 (As mentioned in the email)

Fastq files are in the folder
/home/ss_group01/Tejashwini/TIGS/2Rb/

indch:
sorted_bt2_chr2UCI_IndChIllumina.bam

ste2: (Files further used for other processing by Aditi)
SRR1168951_1.fastq
SRR1168951_2.fastq

uci:
SRR11672501_1.fastq SRR11672501_2.fastq

Terminal output/nohup.out/log.out -
For IndCh - Nil
For STE2* -

```
2021-04-10T11:43:07 fastq-dump.2.9.2 sys: timeout exhausted while creating file within
network system module - Failed to Make Connection in KClientHttpOpen to
'sra-downloadb.be-md.ncbi.nlm.nih.gov:443'
2021-04-10T11:45:25 fastq-dump.2.9.2 sys: timeout exhausted while creating file within
network system module - Failed to Make Connection in KClientHttpOpen to
'sra-downloadb.be-md.ncbi.nlm.nih.gov:443'
2021-04-10T11:45:25 fastq-dump.2.9.2 err: timeout exhausted while creating file within
network system module - failed SRR1168951
```

```
=============================================================
An error occurred during processing.
A report was generated into the file '/home/ss_group01/ncbi_error_report.xml'.
If the problem persists, you may consider sending the file
to 'sra-tools@ncbi.nlm.nih.gov' for assistance.
=============================================================
```

* File should hopefully be completely downloaded

For UCI1-

```
2021-04-11T06:23:24 fastq-dump.2.9.2 sys: timeout exhausted while reading file within network system module - mbedtls_ssl_read returned -76 ( NET - Reading information from the socket failed )
2021-04-11T08:22:43 fastq-dump.2.9.2 sys: timeout exhausted while reading file within network system module - mbedtls_ssl_read returned -76 ( NET - Reading information from the socket failed )
2021-04-11T09:25:51 fastq-dump.2.9.2 sys: timeout exhausted while reading file within network system module - mbedtls_ssl_read returned -76 ( NET - Reading information from the socket failed )
.
.
.
.
2021-04-13T07:10:24 fastq-dump.2.9.2 sys: timeout exhausted while reading file within network system module - mbedtls_ssl_read returned -76 ( NET - Reading information from the socket failed )
2021-04-13T07:46:15 fastq-dump.2.9.2 sys: timeout exhausted while reading file within network system module - mbedtls_ssl_read returned -76 ( NET - Reading information from the socket failed )
Read 142580666 spots for SRR11672501
Written 142580666 spots for SRR11672501
```

For UCI4-
(*In the folder* **)**

```
Read 7528131 spots for SRR11672504
Written 7528131 spots for SRR11672504
```

Reference UCI file was obtained from Aditi

```
scp
group_sshubha01@192.168.1.151:/home/internal/NGS/group_sshubha01/aditi/ref_genomes/UCI/UCI2_3chrs_singleline.fa .
```

After which, chr2 was extracted : grep -A1 “>chr2” > UCI2_chr2.fa

VCF file generation

[The following script was modified as per the sample]

#Pipeline for generating vcf file

#1.Get the reference file ready
#bowtie2-build ../uci2_chr2_reference/UCI2_chr2.fa ../uci2_chr2_reference/UCI2_chr2_idx

#2.Map your reads to it

#bowtie2 -x ../uci2_chr2_reference/UCI2_chr2_idx -1 SRR1168951_1.fastq -2 SRR1168951_2.fastq -S SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st.sam

#3.Convert it to bam file

samtools view -bS SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st.sam -o SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st.bam

#4.Sort the bam file

samtools sort SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st.bam -o SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st_sorted.bam

#5.Index it

samtools index SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st_sorted.bam

#6.Get the reference ready

#samtools faidx ../uci2_chr2_reference/UCI2_chr2.fa

#7.Penultimate step before vcf

samtools mpileup -vu -f ../uci2_chr2_reference/UCI2_chr2.fa -o SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st_sorted.mpileupvcf SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st_sorted.bam

#8.Finally call the vcf

bcftools call -vmO v -o SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st_sorted_mpileup.call.vcf SRR1168951_STE2_Illumina_against_UCI2_Chr2_1st_sorted.mpileupvcf

----

The final file path for the above set of vcf files is -
/home/ss_group01/Tejashwini/TIGS/2Rb_VCF

Further processing -
bgzip sorted_bt2_chr2UCI_IndChIllumina.call.vcf
tabix sorted_bt2_chr2UCI_IndChIllumina.call.vcf.gz

bcftools filter -i "DP>3 && QUAL>10" sorted_bt2_chr2UCI_IndChIllumina.call.vcf.gz -o IndCh_Illumina_bt_dp3_qual10.vcf.gz -O z
Other samples-
3 wild + 4 lab samples sourced from-

uci2.0 raw vcfs:
server: ss_group01@192.168.1.115
path: **/home/ss_group01/Jaysmita/TIGS/uci2.0/{population}/**
population=B ,L ,M ,TI, TII, TIII, TIV
(Individual samples were already filtered, they were merged and transported)

The raw files were processed in the following folder
/home/ss_group01/Tejashwini/TIGS/Raw_VCF_Processed/
1_Raw_Qual10_DP3
2_SNP_filtered
3_INDEL_filtered
4_Candidate_Strict_Check
5_Filtered_Datasets
6_Filter_Set2
Attempts
Filter.sh
nohup.out

Commands used - [SNP and INDEL filtering]

```
#!/bin/bash
for F in ~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/1_Raw_Qual10_DP3/*.vcf.gz
do
tabix -f -p vcf ${F}
done
#SNP filtering
for F in ~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/1_Raw_Qual10_DP3/*.vcf.gz
do
bcftools filter -i "TYPE='SNP'" ${F} -o "${F%_dp3_qual10.vcf.gz}_snp_filtered.vcf.gz" -O
z
done
#Move the files
mv
~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/1_Raw_Qual10_DP3/*_snp_filtered.vcf.gz
~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/2_SNP_filtered/
#Index it
for F in ~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/2_SNP_filtered/*.vcf.gz
do
tabix -f -p vcf ${F}
done
#Indel filtering
for F in ~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/1_Raw_Qual10_DP3/*.vcf.gz
do
bcftools filter -i "TYPE='INDEL'" ${F} -o "${F%_dp3_qual10.vcf.gz}_indel_filtered.vcf.gz"
-O z
```

```
done
#Move the files
mv
~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/1_Raw_Qual10_DP3/*_indel_filtered.vcf.g
z ~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/3_INDEL_filtered/
#Index it
for F in ~/Tejashwini/TIGS/2Rb_Attempt2/1_Raw_Data/3_INDEL_filtered/*.vcf.gz
do
tabix -f -p vcf ${F}
done
```

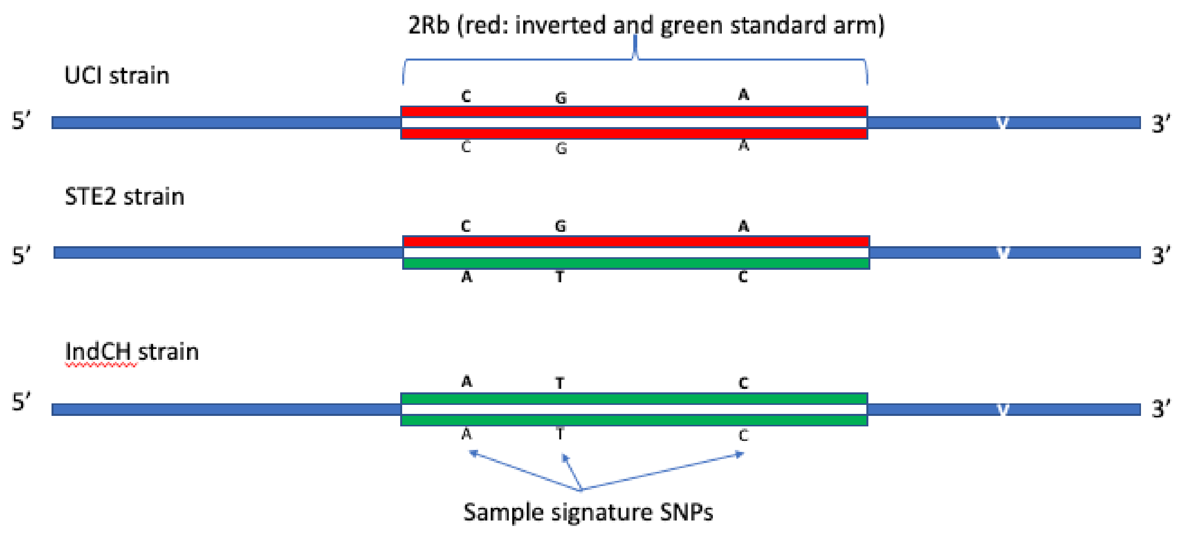
[Merging the files] - Similarly done for SNPs and INDELs
(Note: All combinations are not listed)

```
bcftools merge -m id B_L_M_TI_TII_TIII_TIV_raw_uci2_merged_dp3_qual10.vcf.gz
IndCh_Illumina_bt_dp3_qual10.vcf.gz STE2_dp3_qual10.vcf.gz
UCI2501_dp3_qual10.vcf.gz UCI2504_dp3_qual10.vcf.gz -o
Wild_Lab_IndCh_STE2_UCI_dp3_qual10.vcf.gz -O z

bcftools merge -m id TI_TII_TIII_TIV_raw_uci2_merged_dp3_qual10.vcf.gz
IndCh_Illumina_bt_dp3_qual10.vcf.gz STE2_dp3_qual10.vcf.gz
UCI2501_dp3_qual10.vcf.gz UCI2504_dp3_qual10.vcf.gz -o
Lab_IndCh_STE2_UCI_dp3_qual10.vcf.gz -O z

tabix Wild_Lab_IndCh_STE2_UCI_dp3_qual10.vcf.gz
tabix Lab_IndCh_STE2_UCI_dp3_qual10.vcf.gz
```

Candidate SNPs



(Diagram Credit - Subha ma'am)

Strategy

1. Aim : To find SNP/INDELs that characterize the 2Rb region.
2. Known information :
   UCI is homozygous inverted form, STE2 is heterozygous, IndCh is hypothesized to be homozygous uninverted (Standard)
3. The central dataset used here is a Matrix file. From the VCF file, a matrix is generated expressing the genotype of every sample, for a given chromosomal position.
   [0=Homozygous reference, 1 = Heterozygous alternate, 2 = Homozygous alternate]

Commands used - (Big codes attached below)
[Done for both SNPs and INDELs]

1. **Step-1**

Only merge the IndCh, STE2 and UCI vcf files.

```
bcftools merge -m id IndCh_Illumina_bt_snp_filtered.vcf.gz STE2_snp_filtered.vcf.gz
UCI2501_snp_filtered.vcf.gz UCI2504_snp_filtered.vcf.gz -o
IndCh_STE2_UCI1_UCI4_SNP.vcf.gz -O z
```

2. **Step-2**

Convert the Merged VCF file into a matrix file. *(Code below)*

3. **Step-3**
   Getting the Candidate SNPs

a) The file looks like this

Chrom.Pos	IndCh	STE2	UCI1	UCI4
chr2.17	2	0	2	0
chr2.110	0	0	1	0
chr2.121	1	0	1	0
chr2.128	0	1	0	1
chr2.238	2	1	1	2

b) We need columns which are 2 for IndCh, 1 for STE2 and 0 for UCI, hence we extract those rows only using the awk command.

-> We first copy the header in the other file

```
head -n1 Input/IndCh_STE2_UCI1_UCI4_SNP_012.tsv >
Candidate_SNP_ISU_Full_012.tsv
```

-> Filter it with awk

```
awk '($2==2 && $3==1 && $4==0 && $5==0)'
Input/IndCh_STE2_UCI1_UCI4_SNP_012.tsv >> Candidate_SNP_ISU_Full_012.tsv
```

-> Final Output

```
Chrom.Pos     IndCh   STE2  UCI1   UCI4
chr2.23707    2       1     0      0
chr2.24885    2       1     0      0
```

```
chr2.27398    2    1    0    0
chr2.33214    2    1    0    0
chr2.33895    2    1    0    0
chr2.40818    2    1    0    0
chr2.42653    2    1    0    0
```

-> Getting only SNP position list

```
cut -f1 Candidate_SNP_ISU_Full_012.tsv > Candidate_SNP_ISU_Full_list.txt
```

Using the Master merged file, positions of interest are extracted from the VCF file, which is subsequently converted to a matrix file and then using the txt file, the Candidate SNPs are extracted using grep.

-> **bcftools view** -r chr2:55000000-72000000 ${F} > "${F%_uci2_merged_snp_filtered.vcf.gz}_uci2_merged_SNP_Chr_2R_55million_to_72million_filtered.vcf"

Convert it to a matrix file and use grep

-> **grep -Fwf** Candidate_SNP_ISU_Full_list.txt Lab_uci2_merged_SNP_Chr_2Rb_012.tsv > Output_Candidate_SNP.tsv

--
The above output file is then used to plot heatmaps/PCA/tSNE/KMeans,etc or carry out DESeq filtering
--

DESeq Filtering (All codes attached below)

1. In order to filter the sample cluster-wise, we needed to find which samples belong to which **cluster**, for which a **neighbour joining tree** *(using the code below)* was constructed and the samples and cluster names were manually noted down.

2. The following steps were carried out to re-order the file as input for DESeq

   cut -f1,3,8,14,17,20,22,26,27,28,36,37,38,40,41,48,52,57,60 Candidate_Lab_uci2_2rb_snp_012.tsv > ste2.tsv

   cut --complement -f1,3,8,14,17,20,22,26,27,28,36,37,38,40,41,48,52,57,60 Candidate_Lab_uci2_2rb_snp_012.tsv > indch_snp.tsv

   paste ste2_snp.tsv indch_snp.tsv > Reordered_Candidate_Lab_uci2_2rb_snp_012.tsv

3. The above file was then used as an input for DESeq. *(The code is given below)*

Miscellaneous Codes:

1. **Matrix_012.py**

```
#Python code to make a 0,1,2 matrix

#Importing all the packages required to extract variants from the vcf file
import numpy as np
import pandas as pd
import allel; print('scikit-allel', allel.__version__)
import os

print("Loading vcf file")

#Note: The scikit-allel has some inbuilt functions to analyse variant files

#Now, we input the vcf file (bgzipped/indexed or just vcf) over here
#The callset object returned by read_vcf() is a Python dictionary (dict). It contains
several NumPy arrays, each of which can be accessed via a key.
callset = allel.read_vcf('input.vcf.gz', fields='*')

#Here, we are inputting the Chrom and Pos into a list (Revise python) using the callset
key of variants/CHROM and variants/POS
chrom = callset['variants/CHROM']
pos = callset['variants/POS']

#Later- Read numpy array vs others. Note numpy arrays help in large data
manipulation
#Now, we are creating a genotype array using the key 'calldata/GT'.
allel.GenotypeArray is the inbuilt function telling its a genotype array
gt = allel.GenotypeArray(callset['calldata/GT'])
#Now, in this array whatever is equal to -1, change it to 0 (Say Reference) [It will be -1 if
there was no call found at that position]
gt[gt == -1] = 0

#[This genotype array is our file, hence gt.shape gives the information / structure of the
file ]
print("Structure of VCF File : ",gt.shape)

#What remains is assigning the names. For this, we create a blank list called cp and
from the previously created lists, we merge both the columns (chrom and pos)
cp = []
```

```
for i in range(len(chrom)):
        tmp=chrom[i] + '.' + str(pos[i])
        cp.append(tmp)

#(Check) The genotype array is in the format 0/0, 1/1, etc. These step resolves it to a
single digit
#Transform the genotype data into a 2-dimensional matrix where each cell has the
number of non-reference alleles per call
gt_012 = gt.to_n_alt(fill=0)  #converting to 012

#New array where you convert it to a data frame
gt_012_df = pd.DataFrame(gt_012)

print("DataFrame built, Shuffling columns")

#We now add the names in proper order.
gt_012_df['Chrom.Pos'] = cp
cols = list(gt_012_df.columns)
cols = [cols[-1]] + cols[:-1]
gt_012_df = gt_012_df[cols]

print("Writing onto an output file 012.csv")

###Change output file name
gt_012_df.to_csv("output.tsv", index=None, sep='\t')

#manually changed header using shell or use system call

os.system("sed -i "1s/.*/Chrom.Pos\tIndCh\tSTE2\tUCI1\tUCI4/" output.tsv")
```

---

2. **PCA.py**

```
#!/usr/bin/python
import pandas as pd
import os

df=pd.read_csv("TEST_sample.tsv",sep="\t") #Input file
df_t=df.transpose() #Transposing

df_t.to_csv("python_trans.tsv",sep="\t") #Writing onto another file

#---------------------------------Linux commands---------------------------------------------
os.system("sed -i '1d' python_trans.tsv") #The column header maybe repeated, hence
the step
```

```
#Removing first column which had individual sample lanes
os.system('cut -f2- python_trans.tsv > no_samples.tsv') # no_samples.tsv is the
transposed file with no sample names

#Adding a column of population names to the beginning of the file
os.system('paste Populations.txt no_samples.tsv > final.tsv') #Pasting the files

#Adding SNP names to a file #The first row will have the names of the SNPs
os.system("head -1 no_samples.tsv > features.tsv")

#2D PCA
import matplotlib.pyplot as plt
import pandas as pd

feat=pd.read_csv("features.tsv",sep="\t")
snp=[] #Empty list
print("Step-1")
for col in feat.columns:
	snp.append(str(col))#Time consuming step
df = pd.read_csv("final.tsv",sep="\t")#Reading into a dataframe
from sklearn.preprocessing import StandardScaler
features=snp
x = df.loc[:, features].values
y = df.loc[:,['Populations']].values
x = StandardScaler().fit_transform(x)
print("Step-2")
from sklearn.decomposition import PCA #PCA calculations
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents, columns = ['principal
component 1', 'principal component 2'])
finalDf = pd.concat([principalDf, df[['Populations']]], axis = 1)
variance=pca.explained_variance_ratio_ #To get variance on the x and y axis, note
down
print(variance)
finalDf.to_csv("final_df.tsv",sep="\t")

#Visualization done on the local system by inputing the variance values obtained
from the above output

import matplotlib.pyplot as plt
import pandas as pd
```

```
finalDf=pd.read_csv("final_df_2L_deseq_all.tsv",sep="\t")
variance=[0.33548281,0.10954588]
 #Enter values found on server
pc1="Principal Component 1 ("+str(round((variance[0]*100),2))+"%)"
pc2="Principal Component 2 ("+str(round((variance[1]*100),2))+"%)"
fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel(pc1, fontsize = 17)
ax.set_ylabel(pc2, fontsize = 17)
ax.set_ylim([-10, 100])
ax.set_title('2 component PCA', fontsize = 20)
targets = ['B', 'L', 'M','TI','TII','TIII','TIV','I', 'S', 'U']
colors = ['r', 'g', 'b','c','m','y','lime','maroon','k','navy']
for target, color in zip(targets,colors):
        indicesToKeep = finalDf['Populations'] == target
        ax.scatter(finalDf.loc[indicesToKeep, 'principal component 1']
        , finalDf.loc[indicesToKeep, 'principal component 2']
        , c = color
        , s = 50)
ax.legend(targets)
ax.grid()
plt.show()
```

3. **Heatmap.py**

```
import numpy as np
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt

df=pd.read_csv('input.tsv', sep='\t')
df.index=df['Chrom.Pos']
del df['Chrom.Pos']
#Transpose it to have the populations mentioned at the side
df2=df.T
#Cluster the similar ones
sns.clustermap(df2, cmap='YlGnBu')
plt.show()

#Extra commands
#df3 = sns.clustermap(df2)
#sns.heatmap(df3,cmap='YlGnBu')
```

4. **tSNE.py**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.manifold import TSNE
import seaborn as sns

df=pd.read_csv('Only_Lab_Candidate_Deseq_001.tsv', sep='\t')
df.index=df['Chrom.Pos']
del df['Chrom.Pos']
df.index.names = [None]
df2=df.T
tsne=TSNE(random_state=0)
tsne_results=tsne.fit_transform(df2)
#tsne_results
tsne_results=pd.DataFrame(tsne_results,columns=['tsne1','tsne2'])
colour=['blue','blue','blue','blue','blue','blue','blue','blue','blue','blue','blue','blue','blue','blue','blu
e','red','red','red','red','red','red','red','red','red','red','red','red','red','red','yellow','yellow','yellow','
yellow','yellow','yellow','yellow','yellow','yellow','yellow','yellow','yellow','yellow','yellow','yellow','ye
llow','yellow','green','green','green','green','green','green','green','green','green','green','green','g
reen','green','green','green','green']
plt.scatter(tsne_results['tsne1'],tsne_results['tsne2'],color = colour)
plt.title('Candidate_Lab_uci2_2rb_snp_001.tsv\nNo.of SNPs=....')
plt.xlabel('tsne1')
plt.ylabel('tsne2')
plt.show()
```

5. **KMeans.py**

```
from sklearn.decomposition import PCA
import pandas as pd
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

d=pd.read_csv('Lab_with_ISU_test.tsv', sep='\t')
d.index=d['Chrom.Pos']
del d['Chrom.Pos']
d.index.names = [None]
df2=d.T
features = list(df2.columns)[:-1]
data=df2[features]
```

```
clustering_kmeans = KMeans(n_clusters=2, precompute_distances="auto",
n_jobs=-1)
data['clusters'] = clustering_kmeans.fit_predict(data)
#/home/ta/.local/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:786:
FutureWarning: 'precompute_distances' was deprecated in version 0.23 and will
be removed in 1.0 (renaming of 0.25). It has no effect
#  warnings.warn("'precompute_distances' was deprecated in version "
#/home/ta/.local/lib/python3.8/site-packages/sklearn/cluster/_kmeans.py:792:
FutureWarning: 'n_jobs' was deprecated in version 0.23 and will be removed in
1.0 (renaming of 0.25).
#  warnings.warn("'n_jobs' was deprecated in version 0.23 and will be"

reduced_data = PCA(n_components=2).fit_transform(data)
results = pd.DataFrame(reduced_data,columns=['pca1','pca2'])
sns.scatterplot(x="pca1", y="pca2", hue=data['clusters'], data=results)
#<AxesSubplot:>
plt.title('K-means Clustering with 2 dimensions')
#Text(0.5, 1.0, 'K-means Clustering with 2 dimensions')
plt.show()
```

6. **DESeq.r**

```
library(DESeq2)
library(DESeq)

#put 012 tsv file path/name
print("Step 1 - Reading the table and arrangement")
data=read.table("Reordered_Candidate_Lab_uci2_2rb_indel_012.tsv",header=T,sep=
"\t")
rownames(data)=data$Chrom.Pos
data$Chrom.Pos=NULL
##change according to population name and size

print("Step 2 - Defining the groups")
#UCI2.0 reference
group=c(rep('S',19),rep('I',43))

print("Step 3 - Putting it in CountDataSet")
countDataSet<- newCountDataSet(data,group)
#countDataSet<-estimateSizeFactors(countDataSet, type="poscount")
countDataSet<-estimateSizeFactors(countDataSet)
#countDataSet<- estimateSizeFactors( countDataSet, locfunc=genefilter::shorth )
#sizeFactors(countDataSet) <- 1
countDataSet<-estimateDispersions(countDataSet,fitType="local")
```

```
###for each population combination and p-value of choice
print("Step 4 - Generating the output for negative binomial test")

DEVal=nbinomTest(countDataSet,"I","S")
p1=subset(DEVal,pval<0.0005)
write.table(p1,"I_S_0005.txt",quote=F)
```

7. **Prep.sh**

```
#Script
cat I_S_0005.txt | grep -v "baseMean" | sort -u -k1 -n | cut -f2 -d " " > I_S_pval_0005.txt

head -n1 Candidate_Lab_uci2_2rb_indel_012.tsv >>
Lab_Candidate_indel_2rb_DESeq_filtered_pval_0005.tsv
#Note here the txt file only has the chromosomal positions, hence we can extract the
positions from any master file. Here, the main master file is used instead of the
re-ordered one, to ease further downstream processing of plots.
grep -Fwf I_S_pval_0005.txt Candidate_Lab_uci2_2rb_indel_012.tsv >>
Lab_Candidate_indel_2rb_DESeq_filtered_pval_0005.tsv
```

8. **Make_tree.r**

```
library(ape)
#creating unrooted NJ tree
data2=read.table("Candidate_Lab_uci2_2rb_snp_012.tsv",header=T)
rownames(data2)=data2$Chrom.Pos
data2$Chrom.Pos=NULL
data2=t(data2)
stree=nj(dist.gene(data2))
write.tree(phy=stree, file="Candidate_Lab_uci2_2rb_snp.newick")
```

9. **SNP_Block.py**

```
#!/usr/bin/python

import numpy as np
import pandas as pd
import os

#To write later
os.system('cut -f1 Lab_samples.tsv > Pos.tsv')
os.system('cut -f2 -d "." Pos.tsv > Pos_3782.tsv)

df=pd.read_csv("Pos_3782.tsv",sep="\t")
pos1 = df['Pos'].tolist() #OR pos1 = df['Position'].tolist(),have to do edit to header in this case
```

```
mat=[]
test=[]
z=len(pos1)-1

#Code still has a certain degree of redundancy

for a in range(z):
   print("Iteration="+ str(a+1))
   ie1=pos1[a]
   m=ie1 + 1500
   print(m)
   res = [x for x in range(z) if pos1[x]<m]
   g=res[-1]
   test=pos1[a:g]
   if len(test)>6:
          mat.append(test)
          #print(mat)
   else:
          print("Moving on")


print("Final Matrix")
print(mat)

#Mat_df=pd.DataFrame(mat)
#Mat_df.to_csv("Pos_3782_SNP_Blocks.tsv", index=None, sep='\t')
#print(Mat_df)

#Now to remove redundancy

result = []

for d in mat:
         d = set(d)

         matched = [d]
         unmatched = []
         # first divide into matching and non-matching groups
         for g in result:
         if d & g:
         matched.append(g)
         else:
         unmatched.append(g)
         # then combine all matching groups into one group
```

```
        # while leaving unmatched groups intact
        result = unmatched + [set().union(*matched)]

print(result)
```

10. For Exonic SNPs

```
java -jar /home/ssubha/Tejashwini/TIGS/Software/snpEff/snpEff.jar eff -c
/home/ssubha/Tejashwini/TIGS/Software/snpEff/snpEff.config -v anstephv2
UCIwg_IndChStdwg.call.vcf > UCIwg_IndChStdwg_annotated_anstephv2.vcf


java -jar /home/ssubha/Tejashwini/TIGS/Software/snpEff/SnpSift.jar filter "(QUAL>=30)"
UCIwg_IndChStdwg_annotated_anstephv2.vcf >
UCIwg_IndChStdwg_annotated_anstephv2_qual30.vcf

java -jar /home/ssubha/Tejashwini/TIGS/Software/snpEff/SnpSift.jar filter
"ANN[*].EFFECT has 'missense_variant'"
UCIwg_IndChStdwg_annotated_anstephv2_qual30.vcf >
UCIwg_IndChStdwg_annotated_anstephv2_qual30_missense.vcf

java -jar /home/ssubha/Tejashwini/TIGS/Software/snpEff/SnpSift.jar filter
"ANN[*].EFFECT has 'synonymous_variant'"
UCIwg_IndChStdwg_annotated_anstephv2_qual30.vcf >
UCIwg_IndChStdwg_annotated_anstephv2_qual30_synonymous.vcf
```

Getting the final output file--------------------(Basic Linux Commands)------------------------

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ ls
Candidate_missense_variant.txt  Candidate_synonymous_variant.txt
Chr2Rb_Candidate_22081_Annotated.vcf  genes.gff  Log_file_On_Commands_Used
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$
```

Now the 2Rb annotated file has been filtered for missense and synonymous variants, however they are two separate files and therefore we need to merge them. Here, we have extracted the positions from those two files. We will merge and sort them into one position file, using this we will extract the Candidate Annotated missense and synonymous variants from the master file and write them to another file.

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cat
Candidate_synonymous_variant.txt Candidate_missense_variant.txt | sort -n | uniq >
Synonymous_Missense_Position_Combined.txt
```

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ grep -Fwf
Synonymous_Missense_Position_Combined.txt Chr2Rb_Candidate_22081_Annotated.vcf >
Chr2Rb_Candidate_22081_Annotated_Synonymous_Missense.vcf


ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ mv
Chr2Rb_Candidate_22081_Annotated_Synonymous_Missense.vcf
Chr2Rb_Candidate_Annotated_Synonymous_Missense.vcf


ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f4 -d "|"
Chr2Rb_Candidate_Annotated_Synonymous_Missense.vcf | uniq >
Chr2Rb_Candidate_Annotated_Synonymous_Missense_Geneid_list.txt
#STEP repeated
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final/Int_files$ cut -f4 -d
"|" ../Chr2Rb_Candidate_Annotated_Synonymous_Missense.vcf >
Chr2Rb_Candidate_Annotated_Synonymous_Missense_Geneid_list.txt


#Now the geneid list is to extract gene names from the gff file
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ grep -Fwf
Chr2Rb_Candidate_Annotated_Synonymous_Missense_Geneid_list.txt genes.gff >
Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff

#STEP RE-DONE
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final/Int_files$ grep
-Fwf Chr2Rb_Candidate_Annotated_Synonymous_Missense_Geneid_list.txt genes.gff >
Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff

ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final/Int_files$ rm
../Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff

ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final/Int_files$ mv
Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff ../


#Now we have to create a grand file which will merge the two information

#Preparation for file-1
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f1,2,4,5
Chr2Rb_Candidate_Annotated_Synonymous_Missense.vcf > Vcf_info_set1
```

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f2,3,4,10,11
-d "|" Chr2Rb_Candidate_Annotated_Synonymous_Missense.vcf > Vcf_info_set2

ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ paste
Vcf_info_set1 Vcf_info_set2 > File1
```

#Just Checking
Realized that vcf file will obviously have more variants than the gene file because one gene can have multiple variants

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f2 -d "="
Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff | wc -l
1187
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f2 -d "="
Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff >
Makerid_gff
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ less Makerid_gff
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ gedit Makerid_gff
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ mv Makerid_gff
Makerid_gff.txt
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ grep -Fwf
Makerid_gff.txt File1 | wc -l
1740
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ grep -Fwf
Makerid_gff.txt
Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff | wc -l
1187
```

#Preparation for File-2

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f2,5,9 -d "="
Chr2Rb_Candidate_Annotated_Synonymous_Missense_MAKER_GENES.gff > Gene_info1

ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f1 -d ";"
Gene_info1 > Gene_info2
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f2 -d "="
Gene_info1 > Gene_info3
```

#Clean up the sides

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ gedit Gene_info2
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ gedit Gene_info3

ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ paste Gene_info2
Gene_info3 > File2
```

#Before proceeding ahead, open file1 and change header, replace "|" with "\t"
#Make sure both the files are tab separated (check in text editor) and there is no space in the gene name

# Used awk to merge the files with the help of a key-value structure. With just the gene id as reference

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ awk
'FNR==NR{a[$1]=$2;next}{if(a[$7]==""){a[$7]=0};
        print $1,$2,$3,$4,$5,$6,$7,$8,$9,a[$7]}' File2 File1 | wc -l
1741
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ awk
'FNR==NR{a[$1]=$2;next}{if(a[$7]==""){a[$7]=0};
        print $1,$2,$3,$4,$5,$6,$7,$8,$9,a[$7]}' File2 File1 >
Chr2Rb_CANDIDATE_SNPs_Annotated_Missense_Synonymous_Gene_Name.tsv
```

##
##
Getting the Chr.Pos file to get NJ tree and PCA plot

```
ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ cut -f1,2 -d " "
Chr2Rb_CANDIDATE_SNPs_Annotated_Missense_Synonymous_Gene_Name.tsv >
Chr_Pos_Chr2Rb_CANDIDATE_SNPs_Annotated_Missense_Synonymous_Gene_Name.txt

ta@ta:~/Application/Internship/Work_Outputs/TIGS/July/29July21/Final$ gedit
Chr_Pos_Chr2Rb_CANDIDATE_SNPs_Annotated_Missense_Synonymous_Gene_Name.txt
```

11. Other useful commands used -

```
awk '/^>/ {printf("\n%s\n",$0);next; } { printf("%s",$0);}  END {printf("\n");}' < 'chr2
60911248..60920133 (- strand).fasta' > 'chr2 60911248..60920133 (-
strand)_single_line.fasta'
bcftools isec -p output/ A.vcf.gz B.vcf.gz
bcftools query -f '%POS\n' 0000.vcf | wc -l
bcftools query -l input.vcf
sed -i 's/_/./g' Missense.txt
bgzip your.vcf
tabix -p vcf your.vcf
tabix your.vcf.gz chr1:10,000,000-20,000,000
```