**MED 263: Searching for disease mutations using DNA sequence data and bioinformatics**

**Download all data files needed for this practical from**

**https://github.com/vibansal/med263/raw/master/DATA/DATAFILES.zip**

The same files will also be needed for doing the homework exercises.

We will primarily be using the python pandas library to analyze data in this practical.

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

**1. Analysis of gene-specific variants using computational tools**

Loss-of-function (LoF) mutations in genes are expected to have a strong impact on gene function. In the lecture, we learned that LoF mutations in the MLL2 (also known as KMT2D) cause Kabuki syndrome, a severe multi-system childhood disease. In this exercise, we will analyze coding variants in this gene using computational tools that annotate variants with their predicted functional impact.

(i) The gnomAD database contains variant calls from exome and whole-genome sequencing of > 100,000 human individuals. Kabuki syndrome is a rare disease (prevalence of 1/30000) and the individuals in gnomAD are not expected to have this disease. We will analyze coding variants in this database to determine the frequency of LoF variants.

The gnomAD variant data (v2.1) for KMT2D can be browsed at the portal: <https://gnomad.broadinstitute.org/gene/ENSG00000167548?dataset=gnomad_r2_1>

The file ‘KMT2D.gnomad.variants.csv’ contains the variants from gnomAD (v1, 65,000 individuals) in a simple tabular format with annotations. The csv file can be viewed in a text editor or loaded into a pandas dataframe:

df=pd.read\_csv('KMT2D.gnomad.variants.csv',sep='\t')

df.head()

We can summarize the number of different types of variants (synonymous, missense and LoF) in this VCF file. Recall that LoF variants are of three types: stop\_gain, splice\_acceptor/splice\_donor and frameshift.

df['annotation'].value\_counts()

(iv) Missense variants in the KMT2D gene have also been found to be pathogenic for Kabuki syndrome. However, a significant fraction of individuals in the normal population are also carriers of missense variants in KMT2D. Polyphen2 and SIFT are two of the most widely used tools for prioritizing missense mutations. Polyphen2 categorizes missense mutations as 'probably damaging' (D), 'possibly damaging (P), and benign (B) while SIFT classifies them as 'deleterious' (D) and 'neutral' (N).

The file 'KMT2D.Kabuki.missense.csv' contains 57 missense variants (from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6488975/) identified in Kabuki syndrome individuals. The Polyphen2 and SIFT predictions for each variant are included as separate columns in this file. Similarly, the file 'KMT2D.gnomad.missense.csv' has the annotations for the 1578 missense variants from the file 'KMT2D.gnomad.variants.csv'

To evaluate the ability of these two methods to distinguish between pathogenic missense variants (found in Kabuki syndrome patients) and benign missense variants (found in the gnomAD database), we will count the number of missense variants predicted to be benign/deleterious in the two datasets:

controls=pd.read\_csv('KMT2D.gnomad.missense.csv',sep='\t')

cases=pd.read\_csv('KMT2D.Kabuki.missense.csv',sep='\t')

controls['SIFT'].value\_counts()

cases['SIFT'].value\_counts()

(v) We will use Fisher's exact test to determine if the computational classification of missense variants by SIFT can predict the pathogenicity of these variants. For this, we will tabulate the counts for SIFT predictions (D and T) in a 2x2 table ([[a,b],[c,d]]) and use the fisher\_exact function from Python scipy.stats:

N= controls['SIFT'].value\_counts().to\_list()

D= cases['SIFT'].value\_counts().to\_list()

from scipy.stats import fisher\_exact

fisher\_exact([D,N])

The fisher\_exact function returns two values: (i) the odds ratio and (ii) p-value (two-tailed probability of the odds-ratio being different than 1). Is the p-value significant? What is the interpretation of the odds ratio?

(vi) We will do the same analysis using Polyphen2 predictions and compare with the SIFT results.

**2. Variant filtering in multiple related individuals (rare disease)**

In the lecture, we talked about how DNA sequencing of related individuals can be used to find the genetic cause of rare diseases that affect individuals in a family. This requires prioritizing variants based on a combination of (i) sharing by affected individuals, (ii) population allele frequency and (iii) impact on gene function. In this exercise, we will use variants identified from exome sequencing of four individuals from a single family with a phenotype of early-onset glaucoma (eye disease) to search for the disease causing variants. This study was published a few years ago (<https://pmc.ncbi.nlm.nih.gov/articles/PMC8544095/>) and the exome sequence data for this study is publicly available (<https://www.ncbi.nlm.nih.gov/bioproject/PRJNA555016>).

The raw sequence data was mapped to the reference genome, variant calling was done using the GATK HaplotypeCaller tool and finally the variants were annotated using the Annovar tool. The variants and genotypes that overlap exons have been summarized in a tabular format in the file **"genotypes.coding.csv"**. Each variant has been annotated for its impact on genes and the allele frequency for each variant has also been obtained using data from the gnomAD database. The allele frequency information is provided for two populations: European (EUR\_AF, column 13) and South Asian (SAS\_AF, column 14). The four individuals correspond to:

\* the mother (affected), label S1 and column 6 in the table

\* child 1 (affected), label S2 and column 7 in the table

\* child 2 (unaffected), label S3 and column 8 in the table

\* child 3 (affected), label S4 and column 9 in the table

df = pd.read\_csv('genotypes.coding.csv',sep='\t',na\_values=['.'])

df.fillna(0,inplace=True)

The command df.shape tells us that there are 25,034 variants in this file

df['impact'].value\_counts()

We want to search for variants that are shared by the three affected individuals (and not by the one unaffected individual), are rare in the population and also affect the protein sequence. We will use a population allele frequency threshold of 0.5% to filter out common variants.

a. First, we will filter out variants that are either silent or have unknown impact:

df1 = df[ (df['impact'] != 'synonymous SNV') & (df['impact'] != 'unknown')]

b. Next, we will only keep those variants (heterozygous genotype = 0/1) which are shared by the three affected individuals (S1, S2 and S4):

df2 = df1[(df1['S1'] == "0/1") & (df1['S2'] == "0/1") & (df1['S3'] == "0/0") & (df1['S4'] == "0/1")]

c. Finally, we will filter out variants with allele frequency higher than a threshold (0.005):

df3 = df2[(df2['EUR\_AF'] <= 0.005) & (df2['SAS\_AF'] <= 0.005) ]

How many candidate variants are left after applying the filters?

We can use the pandas unique() function to find the distinct candidate genes with the variants:

genes = df3['gene'].unique()

Next, we will determine if any of the candidate genes are already known to cause human diseases. For this, we will use the OMIM (https://omim.org) database that contains information about human genes and phenotypes. The file "omim.genes.disease" contains human genes and the corresponding phenotypes associated with them. We can search for each candidate gene in this table:

import subprocess

for gene in genes: subprocess.call('grep -w ' + gene + 'omim.genes.disease',shell=True)

Is there a gene that is known to cause an eye-related phenotype?

**3. Using gene expression data for analyzing disease genes**

Gene expression information can be used to prioritize genes for association with disease. The GTEx project (http://gtexportal.org/home/) has generated RNA-seq data using more than 50 different tissues and cell-lines from hundreds of individuals. Summary data (RPKM values per gene for each tissue) is available for download from the GTEX website. We will use this data to analyze gene expression in disease-associated genes. File with RPKM values for all ENSEMBL transcripts and 50+ tissues/cell lines in a table format: “GTEx\_Analysis\_v6p\_RNA-seq\_RNA-SeQCv1.1.8\_gene\_median\_rpkm.gct”

The first line of this file gives information about the tissues/cell-lines and each subsequent line has the expression information for an individual transcript.

We will first a few transformations to the datafile to convert into a tissue x gene matrix and load it into a dataframe:

df = pd.read\_csv('GTEx\_Analysis\_v6p\_RNA-seq\_RNA-SeQCv1.1.8\_gene\_median\_rpkm.gct',sep='\t')

df1=df.T

df1.to\_csv('expression.matrix',sep='\t',header=None)

df = pd.read\_csv('expression.matrix',sep='\t',header=1,index\_col=0)

(i) Plot the gene expression values for KMT2D from the data:

df['KMT2D'].plot(kind='bar')

plt.show()

As we saw in class, KMT2D is expressed at a high level across virtually all tissues which is consistent with the multi-organ phenotype associated with Kabuki syndrome.

(ii) KMT2D is the primary gene that is mutated in Kabuki syndrome (discussed in lecture). KDM6A is another gene that has been implicated in Kabuki syndrome (5-10% of individuals). This suggests that the genes should have a similar expression profile. We will calculate the Spearman rank correlation between the expression profiles of KMT2D and KDM6A:

df['KMT2D'].corr(df['KDM6A'],method='spearman')

The correlation between the expression values of the two genes is high. The expression values can be also be visually looked using a scatter plot:

plt.scatter(df['KMT2D'],df['KDM6A'])

(iii) We can also visualize the expression patterns of genes expressed at high level in a specific tissue using a heatmap.

genes=['CPA1','PRSS1','CELA2A','CTRB1','AMY2A','PNLIP','CTRC','CELA2B','CTRB2']

plt.figure(figsize=(12, 10))

sns.heatmap(df[genes], cmap="YlGnBu", annot=False, fmt="g",xticklabels=True,yticklabels=True)

plt.show()