Perform differential expression analysis on fibrotic and non-fibrotic patients under 4 different treatments on HPC

Author

Sam (Cheng-Hsiang) Lu

Email: Cheng-Hsiang.Lu@cshs.org

Mentor

David Casero

Email: David.Casero@cshs.org

Background

Inflammatory bowel diseases(IBD)

Inflammatory bowel diseases (IBD) are a group of chronic conditions that cause inflammation and damage to the digestive tract. The two main types of IBD are Crohn's Disease and Ulcerative Colitis.

Crohn's disease can affect any part of the digestive tract, like small or large intestine, and can cause symptoms such as abdominal pain, diarrhea, weight loss, and fatigue. It can also cause complications such as fistulas (abnormal connections between different parts of the intestine) and strictures (narrowing of the intestine).

Ulcerative colitis, on the other hand, affects only the colon and rectum and causes symptoms such as bloody diarrhea, abdominal pain, and a frequent need to pass stools. It can also lead to complications such as inflammation of the skin, eyes, and joints.

Both Crohn's disease and ulcerative colitis are chronic conditions, meaning they can last for a lifetime and require ongoing treatment to manage symptoms and prevent complications.

Induced Pluripotent Stem Cells(iPSCs)

Induced pluripotent stem cells (iPSCs) are a type of stem cell that are generated in the laboratory by reprogramming adult cells, such as skin or blood cells, to a pluripotent state. A pluripotent state means that the cells have the potential to develop into any type of cell in the body, just like embryonic stem cells.

iPSCs offer several advantages as they can be generated from the patient's own cells, avoiding issues with immune rejection, ethical concerns and the need for embryos.

iPSCs can be used to study the underlying causes of diseases, test new drugs and therapies, and potentially generate replacement tissues or organs for transplantation.

Aim

In this project, we will be analyzing RNA-seq data from 19 samples, comprising of 10 samples with fibrotic complications and 9 non-fibrotic samples. Each sample has undergone two runs and 4 different treatments(untreated, TGFb, TNFa, and TGF-b+TNF-a), resulting in a total of 151 samples(1 library failed). We used induced pluripotent stem cells (iPSC) to differentiate into myofibroblasts and stimulated the system with different signals to observe its development. The objective is to investigate the effect of four different treatments: untreated, TGF-B, TNF-A, and TGF-B+TNF-A on the development of the system. In the end, we will perform differential expression analysis to identify the genes that are differentially expressed in fibrotic and non-fibrotic samples under 4 treatments.

Pipelines

In HPC

Convert 151 Fastq to Fasta files First, put all fasq.gz files into one folder and list all fastq files' name in fastqfiles.txt.

```
ls *q.gz > fastqfiles.txt
```

Cut redundant suffix "_R1_trimmed" and list all fastq files' name in libraryname.txt and preffix.txt.

```
ls *q.gz | cut -f 1 -d '.' | sed 's/_R1_trimmed//g'  >libraryname.txt
```

```
ls *q.gz | cut -f 1 -d '.' | sed 's/_R1_trimmed//g' > preffix.txt
```

Form a table with 3 columns: fastqfiles.txt libraryname.txt preffix.txt.

```
paste fastqfiles.txt libraryname.txt preffix.txt > tofastatable.txt
```

Create small-sized fasta-formatted files. To submit this job to the cluster on HPC, you need to read the file, library, and prefix. Once you have done that, run the script "generatefastaFromFastaqz" which combine the script "DCfastaqTofastaLibraryId.pl". This results in small-sized fasta-formatted files contain only one header and one sequence per read. You can find all scripts in the "scripts" folder.

```
cat tofastatable.txt | awk {print}' | while read file library preffix ; do qsub -cwd -o $PWI

done
```

This is what each fasta-formatted file would look like:

Generate auxiliary files and directories for each sample Put a list of names of all fasta files in the directory and save them in a text file named "fastafiles.txt".

```
ls *fasta.gz > fastafiles.txt
```

Cut the redundant suffix ".fasta.gz" from the names of all fasta files and generate a new list of file names with the suffix removed in a text file named "targetdirectories.GTF.txt".

```
cat fastafiles.txt |sed 's/.fasta.gz//g' > targetdirectories.GTF.txt
```

Create a separate directory for each sample listed in “fastafiles.txt”.

```
cat fastafiles.txt |while read line ; do mkdir ${line/.fasta\.gz/GTFpass1/} ; done
```

Form the submission script called “sendmyof” Add a shebang line at the beginning of your script file named “sendmyof” to indicate the interpreter that should be used to execute the script.

```
echo '#!/bin/bash/' > sendmyof
```

The command below runs the “generatesendscriptSingleGTFParam” script with several input parameters to map the RNA-seq data with STAR. The input parameters include the list of target directories containing the input data (“targetdirectories.GTF.txt”), the subdirectory name (“GTFpass1”), a parameter file containing settings for STAR alignment (“Parameters.txt”), a prefix for output files (“myof”), the path to the STAR index directory (“/home/luc/RNASEQ_MASTER/Hsapiens/GRC38/INDEXES/GRCh38.primary.33.basicselected.STAR2.7.
the path to the input data directory (“/home/luc/iPSC/MYOFIBROBLAST/”), the amount of free memory to use (“mem_free=32G”), and the number of threads to use (“8”). In the end, it will generate a “processLaneSingleGTF-Param” file and run the STAR package in each sample’s folder.

```
./generatesendscriptSingleGTFParam targetdirectories.GTF.txt GTFpass1 Parameters.txt myof /h
```

Change sendmyof into executable mode and run sendmyof.

```
chmod a+x sendmyof
. sendmyof
```

It will take less than one day to run through 151 samples and generate each sample a folder which contain every output from STAR.

Create a table summarizing the mapping statistics for each sample Change directory into one sample file which ends with “GTFpass1”. Extract the first column from the mapping statistics file and store it in “temp2.txt”.

```
grep "|" 008iP22TGFbM_S71GTFpass1/008iP22TGFbM_S71GTFpass1Log.final.out | cut -f 1 -d "|" |
```

The first column from the mapping statistics file.

```
                                Started job on |
                            Started mapping on |
                                    Finished on |
       Mapping speed, Million of reads per hour |

                           Number of input reads |
                     Average input read length |
                                  UNIQUE READS:
                   Uniquely mapped reads number |
                        Uniquely mapped reads % |
                          Average mapped length |
                       Number of splices: Total |
            Number of splices: Annotated (sjdb) |
                       Number of splices: GT/AG |
                       Number of splices: GC/AG |
                       Number of splices: AT/AC |
              Number of splices: Non-canonical |
                     Mismatch rate per base, % |
                        Deletion rate per base |
                       Deletion average length |
                       Insertion rate per base |
                     Insertion average length |
                               MULTI-MAPPING READS:
        Number of reads mapped to multiple loci |
             % of reads mapped to multiple loci |
        Number of reads mapped to too many loci |
             % of reads mapped to too many loci |
                                 UNMAPPED READS:
Number of reads unmapped: too many mismatches |
     % of reads unmapped: too many mismatches |
          Number of reads unmapped: too short |
               % of reads unmapped: too short |
              Number of reads unmapped: other |
                   % of reads unmapped: other |
                                 CHIMERIC READS:
                     Number of chimeric reads |
                          % of chimeric reads |
```

Create an empty temporary file for storing intermediate results.

```
rm tempprev.txt
touch tempprev.txt
```

Extract the total mapped reads from each subsequent mapping statistics file and combine with previous results.

```
ls *pass1/*final.out | while read line ; do
grep "|" $line | cut -f 2  > temp.txt
paste tempprev.txt temp.txt > tempnew.txt
mv tempnew.txt tempprev.txt
done
```

Remove the first column and write the final results to a file called "mappingstatsFirstpass.txt".

```
cut -f 2- tempprev.txt | awk 'NR>3 {print}'  > tempnew.txt
mv tempnew.txt tempprev.txt
paste temp2.txt tempprev.txt > mappingstatsFirstpass.txt
```

The mappingstatsFirstpass.txt would look like this:

Mapping speed, Million of reads per hour			452.92	344.45	460.17	487.96	331.34	441.99	560.78	487.93	532.33	351.69	381.99	447.59	463.27
Number of input reads	46046803		40855908		43588067		47982561		39392754		33886137		72122988		53266027
Average input read length	75	75	75	75	75	75	75	75	75	75	75	75	75	75	75
Uniquely mapped reads number	42647348		36668193		39982295		43570276		36214207		31289999		66061689		48190947
Uniquely mapped reads %	92.62%	89.75%	91.73%	90.80%	91.93%	92.34%	91.60%	90.47%	93.77%	90.29%	90.29%	90.70%	92.04%	91.95%	90.83%
Average mapped length	75.21	75.21	75.21	75.20	75.20	75.20	75.19	75.21	75.20	75.20	75.20	75.24	75.24	75.23	75.22
Number of splices: Total	14107761		11743321		12025357		12627415		11851597		10196588		20128263		13942120
Number of splices: Annotated (sjdb)	14029840		11673663		11945482		12536747		11787471		10140891		19995363		
Number of splices: GT/AG	14015583		11657575		11928403		12523568		11777228		10127320		19961004		13827884
Number of splices: GC/AG	78050	74078	84368	87757	62434	59661	145139	96183	139381	88755	60185	79428	94829	86337	89305
Number of splices: AT/AC	7048	6261	6785	8291	6179	5252	12249	9152	15682	8499	6008	7801	9790	8299	7743
Number of splices: Non-canonical		7080	5407	5801	7799	5756	4355	9871	8901	10933	5153	3764	6609	7788	5706
Mismatch rate per base, %	0.27%	0.28%	0.29%	0.28%	0.28%	0.28%	0.31%	0.28%	0.29%	0.29%	0.29%	0.28%	0.27%	0.28%	0.30%
Deletion rate per base	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%	0.02%
Deletion average length	1.47	1.47	1.47	1.47	1.48	1.48	1.47	1.46	1.47	1.48	1.50	1.48	1.48	1.47	1.51
Insertion rate per base	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%	0.01%
Insertion average length	1.61	1.52	1.43	1.43	1.63	1.60	1.47	1.43	1.62	1.42	1.38	1.44	1.59	1.53	1.42
Number of reads mapped to multiple loci			2398878	3287402	2498197	3093230	2029231	1701575	4120922	3568843	3900452	2580456	1853980	3009727	2483263
% of reads mapped to multiple loci	5.21%		8.05%	5.73%	6.45%	5.15%	5.02%	5.71%	6.70%	4.80%	5.64%	6.11%	6.49%	4.84%	5.20%
Number of reads mapped to too many loci			222693	207795	314715	343251	343822	264218	498200	397171	262921	547542	254116	316223	407743
% of reads mapped to too many loci	0.48%		0.51%	0.72%	0.72%	0.87%	0.78%	0.69%	0.75%	0.32%	1.20%	0.84%	0.68%	0.79%	0.50%
Number of reads unmapped: too many mismatches			1269	1089	1322	1446	1105	1001	2452	1565	1799	1824	1173	1452	1228
% of reads unmapped: too many mismatches			0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Number of reads unmapped: too short	759644		674719	771365	950053	788663	614921	1410179	1081555	902701	1286763	815091	960438	1181777	994451
% of reads unmapped: too short	1.65%		1.65%	1.77%	1.98%	2.00%	1.81%	1.96%	2.03%	1.09%	2.81%	2.69%	2.07%	2.30%	2.32%	2.55%
Number of reads unmapped: other			16971	16710	20173	24305	15726	13423	29546	25946	17237	24268	22820	24661	12953	12190
% of reads unmapped: other	0.04%		0.04%	0.05%	0.05%	0.04%	0.04%	0.04%	0.05%	0.02%	0.05%	0.08%	0.05%	0.03%	0.03%	0.03%
Number of chimeric reads	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
% of chimeric reads	0.00%		0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%

Counts Generate a directory called "COUNTS" and copy all gene count files to this folder and then clean all file names.

```
mkdir COUNTS
cp *pass1/*PerGene* COUNTS/
```

```
ls *tab|while read line ; do mv $line ${line/GTFpass1ReadsPerGene.out/} ; done
```

For each count file, extract and create the five count tables.

```
ls *.tab | while read line ; do
echo $line
cat $line | awk 'NR==3{print}' | cut -f 2- > ${line/tab/nofeature\.tab}
cat $line | awk 'NR==4{print}' | cut -f 2- > ${line/tab/ambiguous\.tab}
cat $line | awk 'NR>4{print}' | cut -f 2 > ${line/tab/nostrand\.tab}
cat $line | awk 'NR>4{print}' | cut -f 3 > ${line/tab/sense\.tab}
cat $line | awk 'NR>4{print}' | cut -f 4 > ${line/tab/antisense\.tab}
done
```

Make a Geneid list from one of the count tables as "countsannot_GRCh38.primary.Selected.Geneid.txt".

```
ls 008iP22TGFbM_S71.tab | head -1 | while read line; do
cut -f 1 $line | awk 'NR>4{print}' > countsannot_GRCh38.primary.Selected.Geneid.txt
done
```

Create a file listing the names of all samples as "RBarretTNFATGFBsamples.txt".

```
ls *.sense.tab | sed 's/.sense.tab//g' | tr -s " " "\n" | sed 's/_1//g' > RBarretTNFATGFBsam
```

Make count tables for sense, anti-sense, nostrand, ambiguous, and nofeature reads.

```
****
# combine all sense counts into RBarretTNFATGFB_sense.ALL.cnt
paste *.sense.tab > RBarretTNFATGFB_sense.ALL.cnt

# combine all antisense counts into RBarretTNFATGFB_antisense.ALL.cnt
paste *.antisense.tab > RBarretTNFATGFB_antisense.ALL.cnt

# combine all nostrand counts into RBarretTNFATGFB_nostrand.ALL.cnt
paste *.nostrand.tab > RBarretTNFATGFB_nostrand.ALL.cnt

# combine all ambiguous counts into RBarretTNFATGFB_ambiguous.cnt
cat *ambiguous.tab > RBarretTNFATGFB_ambiguous.cnt

# combine all nofeature counts into RBarretTNFATGFB_nofeature.cnt
cat *nofeature.tab > RBarretTNFATGFB_nofeature.cnt
```

Next, I am going to use "RBarretTNFATGFB_antisense.ALL.cnt" file for the further analysis.

In MATLAB

Transfer data and import annotation Transfer the counts, annotation, and mappability data to your local laptop.

```
RBarretTNFATGFBCnt = textread('RBarretTNFATGFB_antisense.ALL.cnt','');
RBarretsamplesTNFATGFB = textread('RBarretTNFATGFBsamples.txt','%s');
RBarretsampleskeysTNFATGFB = textread('samplekeys_Sam.txt','%s');

% calculate the sum of the counts in RBarretTNFATGFBCnt, divides the result by 1000000, and
RBarretTNFATGFBmeta_seqdepth=round(sum(RBarretTNFATGFBCnt)/1000000);
```

You can find these annotation in the "mappability and R code" folder.

```
Gencode_33_Selected_MappSS=textread('mappability and R code/gencode.v33.Selected.ReadsPerGen
Gencode_33_Selected_MappUS=textread('mappability and R code/gencode.v33.Selected.ReadsPerGen
Gencode_33_Selected_Geneid=textread('mappability and R code/gencode.v33.annotation.Selected.
Gencode_33_Selected_Biotype=textread('mappability and R code/gencode.v33.annotation.Selected
Gencode_33_Selected_Genename=textread('mappability and R code/gencode.v33.annotation.Selecte
```

Compile counts First, initialize a new variable called RBarretTNFATGFBTPM with the same count data as RBarretTNFATGFBCnt. Then, iterates over each gene in the count data matrix. For each gene, the corresponding row in RBarretTNFATGFBTPM is updated by dividing the count data by the read counts from the "Gencode_33_Selected_MappSS", multiplying by 1000, and storing the result in RBarretTNFATGFBTPM.

Finally, iterates over each sample in the TPM data matrix. For each sample, the corresponding column in RBarretTNFATGFBTPM is updated by dividing the values in the column by the sum of the values in the column, multiplying by 1,000,000, and storing the result in RBarretTNFATGFBTPM. This step **normalizes the TPM values** across samples and scales the resulting values to TPM.

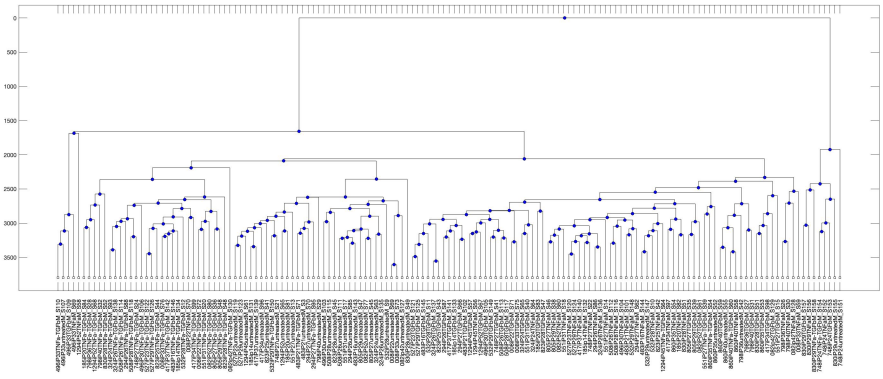
```
RBarretTNFATGFBTPM = RBarretTNFATGFBCnt;
for i=1:size(RBarretTNFATGFBCnt,1)
% divid the gene count matrix RBarretTNFATGFBCnt by Gencode_33_Selected_MappSS matrix, which
RBarretTNFATGFBTPM(i,:) = RBarretTNFATGFBCnt(i,:)/Gencode_33_Selected_MappSS(i)*1000;
end
% set any NaN or Inf values resulting from the normalization process to 0
RBarretTNFATGFBTPM(isnan(RBarretTNFATGFBTPM)) = 0;
RBarretTNFATGFBTPM(isinf(RBarretTNFATGFBTPM)) = 0;
for i=1:size(RBarretTNFATGFBTPM,2)
% scale the TPM values so that the sum of expression values across each sample of the matrix
RBarretTNFATGFBTPM(:,i) = RBarretTNFATGFBTPM(:,i)/sum(RBarretTNFATGFBTPM(:,i))*1000000;
end
```

Make the first dendrogram Make a dendrogram to visualize the relationships among samples in the RBarretTNFATGFB dataset based on their gene expression profiles.

1. I generates a random selection of 1,000 genes from the TPM data matrix.
2. Calculates the pairwise distances between the selected genes.
3. Creates a for loop that iterates 9,999 times. For each iteration, a new random selection of 1,000 genes is generated, and the pairwise distances between these genes are added to the previous 'thisdist' calculation.
4. Converts the one-dimensional distance vector 'thisdist' into a distance matrix 'thisdistmat' using the 'squareform' function.
5. Generates a hierarchical clustering tree based on the distance matrix 'thisdistmat'.

Overall, I perform a clustering analysis on a subset of genes in the RBarretTNFATGFB dataset to visualize the relationships among samples based on their gene expression profiles.

```
thisrand = unique(randi([1 size(RBarretTNFATGFBTPM,1)],1,1000));
thisdist = pdist(RBarretTNFATGFBTPM(thisrand,:)');
for i=1:9999
thisrand = unique(randi([1 size(RBarretTNFATGFBTPM,1)],1,1000));
thisdist = thisdist+pdist(RBarretTNFATGFBTPM(thisrand,:)');
end
thisdistmat = squareform(thisdist/10000);
thistree = seqlinkage(thisdistmat,'average', RBarretsamplesTNFATGFB)
plot(thistree, 'ORIENTATION', 'top')
```



All biotypes counts percents This part of codes is performing all biotypes counts percents across multiple samples.

```
% use the unique function and stored the allbiotypes variable.
allbiotypes = unique(Gencode_33_Selected_Biotype);

% create A cell array allbiotypeslength to store the lengths of each biotype name.
allbiotypeslength = cell(length(allbiotypes),1);

% two new matrices, allbiotypescounts and allbiotypescountspercents, are initialized with ze
allbiotypescounts = zeros(length(allbiotypes),size(RBarretTNFATGFBCnt,2));
allbiotypescountspercents = zeros(length(allbiotypes),size(RBarretTNFATGFBCnt,2));

for i=1:length(allbiotypes)
%  finds all the indices of Gencode_33_Selected_Biotype that match the current biotype. Ther
temp = strmatch(allbiotypes{i}, Gencode_33_Selected_Biotype);
% Stored the length of the temp vector represents the number of genes with the current bioty
allbiotypeslength{i} = length(temp);
if length(temp)>1
% Sum the expression values for all genes with the current biotype across all samples. The r
allbiotypescounts(i,:) = sum(RBarretTNFATGFBCnt(strmatch(allbiotypes{i}, Gencode_33_Selected
allbiotypescountspercents(i,:) = allbiotypescounts(i,:)./sum(RBarretTNFATGFBCnt)*100;
end
end

dlmwrite('allbiotypescountspercents.txt', allbiotypescountspercents,'delimiter','\t')
writetable(cell2table(allbiotypes),'allbiotypes.txt','WriteVariableNames',0)
```

Using Excel, create a spreadsheet using "allbiotypes.txt" and "allbiotypescountspercents.txt", and calculate the minimum, maximum, and average values for each biotype. You can access my completed spreadsheet here. Notably, protein_coding genes exhibit an average of 98.65% among the various biotypes, consistent with my expectations.

Biotypes	min	max	average
IG_C_gene	0.000000000	0.000446580	0.000013340
IG_D_gene	0.000000000	0.000000000	0.000000000
IG_J_gene	0.000000000	0.000013813	0.000000235
IG_V_gene	0.000000000	0.000073058	0.000010604
Mt_rRNA	0.228350000	1.165600000	0.487934371
Mt_tRNA	0.000141280	0.002049800	0.000492376
TEC	0.011838000	0.029033000	0.018976556
TR_C_gene	0.000000000	0.002667100	0.000287637
TR_D_gene	0.000000000	0.000011542	0.000000580
TR_J_gene	0.000000000	0.000076445	0.000009219
TR_V_gene	0.000000000	0.000061956	0.000011904
lncRNA	0.537580000	1.612500000	0.825408079
miRNA	0.002352100	0.014811000	0.004202784
misc_RNA	0.000641580	0.002443400	0.001048973
protein_coding	**97.675000000**	**99.144000000**	**98.658132450**
rRNA	0.000000000	0.000028828	0.000005066
ribozyme	0.000000000	0.000007357	0.000000293
sRNA	0.000000000	0.000004680	0.000000135
scRNA	0.000000000	0.000000000	0.000000000
scaRNA	0.000002616	0.000324330	0.000036518
snRNA	0.000262070	0.001464200	0.000543351
snoRNA	0.001727500	0.004335800	0.002883131
vaultRNA	0.000000000	0.000000000	0.000000000

Protein coding genes The "allbiotypes.txt" file contains multiple biotypes. For the next step, I will only retain the "protein_coding" biotype.

```
allbiotypes=unique(Gencode_33_Selected_Biotype);
% finds the 15th unique value, which is protein_coding, of Gencode_33_Selected_Biotype in th
proteincodingindx = strmatch(allbiotypes{15}, Gencode_33_Selected_Biotype);
biotypeindx = proteincodingindx;

% creates an array additionalgenes contains the indices of genes that have certain prefixes
additionalgenes = [strmatch('MT-',Gencode_33_Selected_Genename) ; strmatch('H1',Gencode_33_S
```

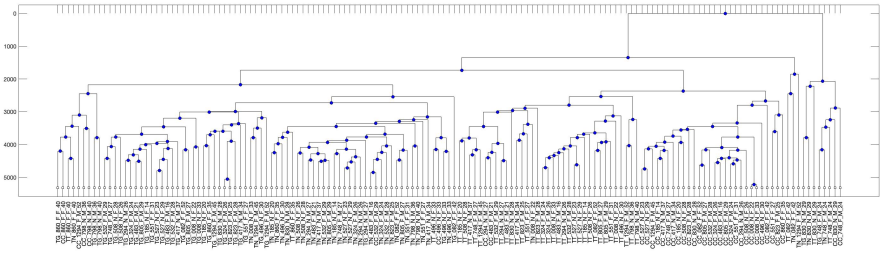
```
% creates an array nonadditionalgenes with the same length as the Gencode_33_Selected_Genena
nonadditionalgenes = 1:length(Gencode_33_Selected_Genename);
% remove the indices of genes in additionalgenes from the nonadditionalgenes array.
nonadditionalgenes(additionalgenes) = [];

% mappableindx contains the indices of elements in the Gencode_33_Selected_MappSS array that
mappableindx = find(Gencode_33_Selected_MappSS>50);


% a new variable finalIndexGeneric which is the intersection of three other variables: bioty
finalIndexGeneric = intersect(biotypeindx,intersect(nonadditionalgenes,mappableindx));
% find the indices of rows in RBarretTNFATGFBCnt that have a sum greater than 150.
countindx = find(sum(RBarretTNFATGFBCnt')'>150);

% update finalIndexGeneric to be the intersection of finalIndexGeneric and countindx.
finalIndexGeneric=intersect(finalIndexGeneric,countindx);


% create a new variable RBarretTNFATGFBCnt_GMask which is a subset of RBarretTNFATGFBCnt cor
RBarretTNFATGFBCnt_GMask = RBarretTNFATGFBCnt(finalIndexGeneric,:);

Gencode_33_Selected_Geneid_GMask = Gencode_33_Selected_Geneid(finalIndexGeneric);
Gencode_33_Selected_Genename_GMask = Gencode_33_Selected_Genename(finalIndexGeneric);
Gencode_33_Selected_MappSS_GMask = Gencode_33_Selected_MappSS(finalIndexGeneric);
Gencode_33_Selected_MappUS_GMask = Gencode_33_Selected_MappUS(finalIndexGeneric);


% normalize the expression data like we do previously
RBarretTNFATGFBExpression_GMask = RBarretTNFATGFBCnt_GMask;
for i=1:size(RBarretTNFATGFBExpression_GMask,2)
RBarretTNFATGFBExpression_GMask(:,i) = RBarretTNFATGFBCnt_GMask(:,i)/sum(RBarretTNFATGFBCnt_
end
for i=1:size(RBarretTNFATGFBExpression_GMask)
RBarretTNFATGFBExpression_GMask(i,:) = RBarretTNFATGFBExpression_GMask(i,:)/Gencode_33_Selec
end
RBarretTNFATGFBExpression_GMask(isnan(RBarretTNFATGFBExpression_GMask)) = 0;
RBarretTNFATGFBExpression_GMask(isinf(RBarretTNFATGFBExpression_GMask)) = 0;

% RBarretTNFATGFBCPM_GMask contains the expression data normalized only by CPM, using the sa
RBarretTNFATGFBCPM_GMask = zeros(size(RBarretTNFATGFBCnt_GMask));
for i=1:size(RBarretTNFATGFBCnt_GMask,2)
RBarretTNFATGFBCPM_GMask(:,i) = RBarretTNFATGFBCnt_GMask(:,i)/sum(RBarretTNFATGFBCnt_GMask(:
end

% RBarretTNFATGFBTPM_GMask contains the expression data normalized only by TPM.
RBarretTNFATGFBTPM_GMask = RBarretTNFATGFBCnt_GMask;
for i=1:size(RBarretTNFATGFBCnt_GMask,1)
RBarretTNFATGFBTPM_GMask(i,:) = RBarretTNFATGFBCnt_GMask(i,:)/Gencode_33_Selected_MappSS_GMa
```

```
end
RBarretTNFATGFBTPM_GMask(isnan(RBarretTNFATGFBTPM_GMask)) = 0;
RBarretTNFATGFBTPM_GMask(isinf(RBarretTNFATGFBTPM_GMask)) = 0;
for i=1:size(RBarretTNFATGFBTPM_GMask,2)
RBarretTNFATGFBTPM_GMask(:,i) = RBarretTNFATGFBTPM_GMask(:,i)/sum(RBarretTNFATGFBTPM_GMask(
end
```

Dendrogram with only protein coding genes Perform hierarchical clustering on a subset of the gene expression data stored in the variable RBarretTNFATGFBTPM_GMask with only protein coding genes.

```
thisrand = unique(randi([1 size(RBarretTNFATGFBTPM_GMask,1)],1,1000));
thisdist = pdist(RBarretTNFATGFBTPM_GMask(thisrand,:)');
for i=1:9999
thisrand = unique(randi([1 size(RBarretTNFATGFBTPM_GMask,1)],1,1000));
thisdist = thisdist+pdist(RBarretTNFATGFBTPM_GMask(thisrand,:)');
end
thisdistmat = squareform(thisdist/10000);
thistree = seqlinkage(thisdistmat,'average', RBarretsampleskeysTNFATGFB)
plot(thistree,'ORIENTATION','top')
```

Basically, the plot is clustered by their treatments.



The percent of the top 100 genes Calculates the top 100 expressed genes in each sample based on their transcript per million (TPM) values in the RBarretTNFATGFBTPM_GMask matrix.

```
% iterates over 151 samples, it first sorts the TPM values of all genes in descending order
yall=[];
for i=1:151
[x y]=sort(RBarretTNFATGFBTPM_GMask(:,i),'descend');
yall=unique([y(1:100); yall]);
top100percent(i)=sum(RBarretTNFATGFBTPM_GMask(y(1:100),i))/1000000;
end
```

After calculating the sum of the percent of the top 100 genes, it is 58.25%.

In the end, we store the Gencode_33_Selected_Geneid_GMask.txt, Gencode_33_Selected_Genename_GMask.txt, Gencode_33_Selected_MappSS_GMask.txt, RBarretTNFATGFBTPM_GMask.txt, and RBarretTNFATGFBCnt_GMask.txt for ours further analysis in R.

```
writetable(cell2table(Gencode_33_Selected_Geneid_GMask),'Gencode_33_Selected_Geneid_GMask.tx
writetable(cell2table(Gencode_33_Selected_Genename_GMask),'Gencode_33_Selected_Genename_GMas
dlmwrite('Gencode_33_Selected_MappSS_GMask.txt', Gencode_33_Selected_MappSS_GMask,'delimiter
dlmwrite('RBarretTNFATGFBTPM_GMask.txt', RBarretTNFATGFBTPM_GMask,'delimiter','\t')
dlmwrite('RBarretTNFATGFBCnt_GMask.txt', RBarretTNFATGFBCnt_GMask,'delimiter','\t')
```

In R

Install packages First, install packages BiocManager, BiocLite, IHW, DESeq2, and ggplot2. Then, read in RBarretTNFATGFBCnt_GMask.txt, RBarretTNFATGFBsamples.txt, samplekeys_Sam.txt, and Gencode_33_Selected_Genename_GMask.txt.

```
setwd("/Users/LuC/Desktop/Cedars-Sinai/PROJECTS/IBD_RNASeq/RBARRETTNFATGFB/")#setwd("/Users/
```

Form samplekeys_Sam.tab Separate samplekeys_Sam.txt by "_" to get samplekeys_Sam.tab before next step. Here are my code in terminal.

```
#In terminal
#Create an empty file to store the output
touch samplekeys_Sam.tab

#Loop over the sample names and split them by "_"
for sample in $(cat samplekeys_Sam.txt); do
    IFS=_ read -r col1 col2 col3 col4 col5 <<< "$sample"
    echo -e "$col1\t$col2\t$col3\t$col4\t$col5" >> samplekeys_Sam.tab
done
```

Generate a sampleTableTNFATGFB The sampleTableTNFATGFB contains Treatment, Line, Pheno, Sex, Pass, Factor, and Batch.

```
sampleTableTNFATGFB = read.table("samplekeys_Sam.tab")rownames(sampleTableTNFATGFB)<-sampleK
```

	Treatment	Line	Pheno	Sex	Pass	Factor	Batch
TG_008_N_F_22	TG	8	N	F	22	TG_N	8_22
TT_008_N_F_22	TT	8	N	F	22	TT_N	8_22
TN_008_N_F_22	TN	8	N	F	22	TN_N	8_22
CC_008_N_F_22	CC	8	N	F	22	CC_N	8_22
TG_008_N_F_33	TG	8	N	F	33	TG_N	8_33
TT_008_N_F_33	TT	8	N	F	33	TT_N	8_33
TN_008_N_F_33	TN	8	N	F	33	TN_N	8_33
CC_008_N_F_33	CC	8	N	F	33	CC_N	8_33
TG_082_F_F_52	TG	82	F	F	52	TG_F	82_52
TT_082_F_F_52	TT	82	F	F	52	TT_F	82_52
TN_082_F_F_52	TN	82	F	F	52	TN_F	82_52
CC_082_F_F_52	CC	82	F	F	52	CC_F	82_52

DESeq2 package The RBarretTNFATGFBCntGMaskBatch is created with the **Batch** information specified in the design formula, while the RBarretTNFATGFBCntGMaskFactor is created with the **treatment and phenotype** information specified in the design formula.

Next, the DESeq function is used to estimate size factors and dispersion values for the DESeqDataSet objects.

Finally, the varianceStabilizingTransformation function is used to perform variance stabilizing transformation on the DESeqDataSet objects. This transformation is important for reducing the effect of noise and heteroscedasticity in the data, making it more suitable for downstream analyses such as differential gene expression analysis.

```
RBarretTNFATGFBCntGMaskBatch <- DESeqDataSetFromMatrix(RBarretTNFATGFBCntGMask, colData= sam
```

PCA

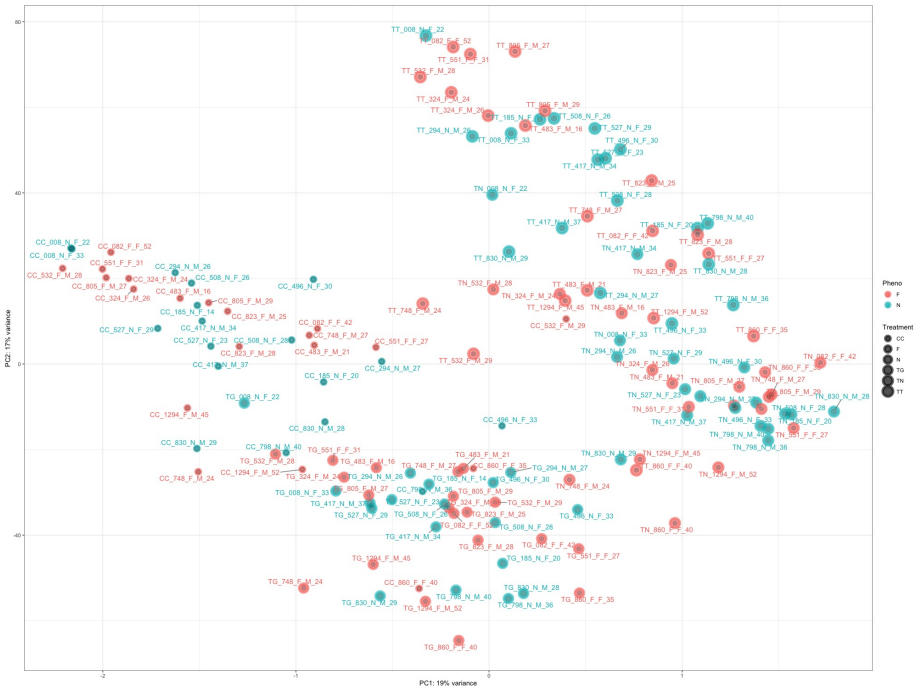
```
#perform principal component analysis (PCA) on the variance-stabilized counts data pcabatch
#give the percentage of variance explained by each principal componentpercentVarbatch <- rou
#pcabatch$rotation is a matrix containing the loadings of the principal components.aloadbatc

#normalize the loadings in aloadbatch so that each column (i.e., PC) sums to 1. aloadrelativ
#pcabatch$x is a matrix containing each sample's coordinate on each principal componentpcaba
#center the PC1 scores in pcabatchR to have a mean of 0. This is done so that the PC1 variab
```

PCA plots

```
ggplot(pcabatchR, aes(PC1, PC2, color= Pheno)) +  geom_point(aes(size= Treatment),alpha=0.6,
```

The plot shows the relationship between **PC1 and PC2** colored by **Pheno** variable, with the point size indicating the **Treatment** variable.

Four distinct groups were formed based on their treatment: the CC group (untreated) is located in the right corner, the TG group (treated with TGF-b) is located at the bottom, the TN group (treated with TNF-a) is located in the right corner, and the TT group (treated with both TGF-b and TNF-a) is located at the top. These groups were differentiated based on PC1, which accounted for 19% of the variance.



####

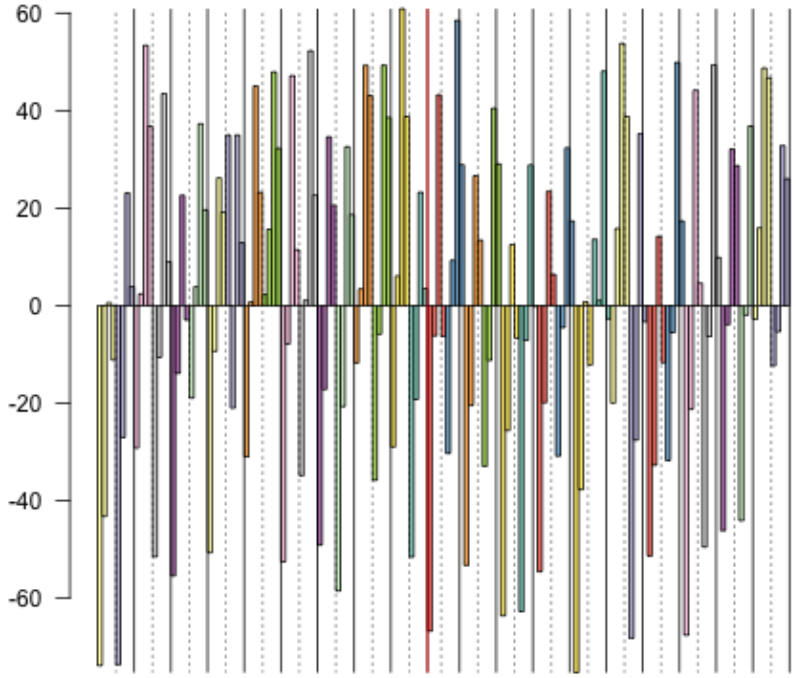
A series of bar plots (one for each principal component)

Each bar plot represents the loadings of all samples on a given principal component.

```
#the resulting vector coul will contain 12 colors from the "Set3" palette.library(RColorBrew
```

A red line is drawn at position 72 in order to separate the non-fibrotic group and the fibrotic group. Within each patient, the treatment order would be CC, TG, TN, TT. The color of each bar represents the batch of the sample, with a unique color assigned to each batch. The vertical lines on the plot indicate the position of specific loadings, with thin and thick lines indicating different positions.

For example, this is PC_1.png. From this plot, you can see that the highest sxpression is closely related with TNF-a. As for the first patient, compaire to the contorl(untreated), the TNF-a group is much higher and the TT group (TGF-b+TNF-a) is not that high.



PCA rank matrix Take csv files and converts it to the txt files with the second column onwards. It does this by first removing the first row using awk, replacing multiple commas with tabs using tr, and removing the first column using cut.

```
#In terminal
cat aloadrelativeMask_batchmodel_filtered.csv | awk 'NR>1{print}' | tr -s "," "\t" | cut -f
cat pca_batchmodel_x.csv | awk 'NR>1{print}' | tr -s "," "\t" | cut -f 2- > pca_batchmodel_x
```

Read in the preprocessed data files created in the previous steps and store them in variables pcabatch_samples and pca_loadings, respectively.

```
#In Matlab
pcabatch_samples = textread('pca_batchmodel_x.clean.txt','');
pca_loadings = textread('aloadrelativeMask_batchmodel_filtered.clean.txt','');
```

Sort the three columns of pca_loadings in descending order and store the sorted values in variables x1, x2, and x3, and the corresponding indices in y1, y2, and y3.

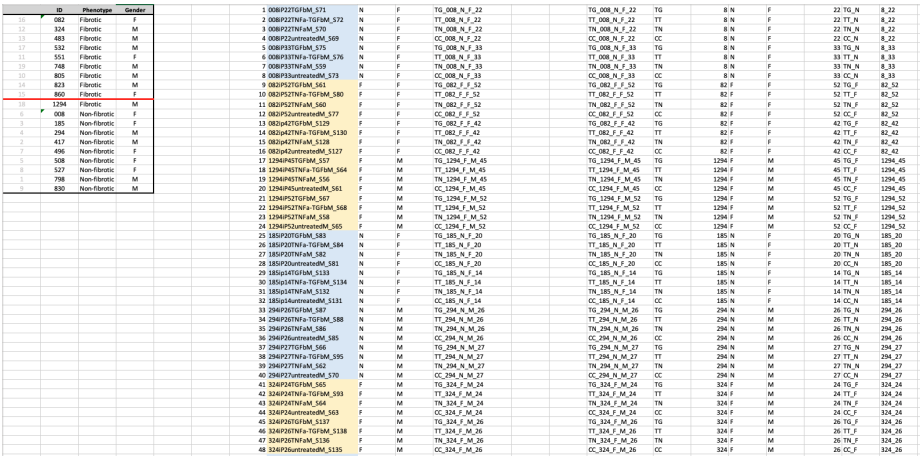
```
%%%%%%%% PCA SUPP

%x = pca_loading number, y = its index

[x1 y1]=sort(pca_loadings(:,1),'descend');
[x2 y2]=sort(pca_loadings(:,2),'descend');
[x3 y3]=sort(pca_loadings(:,3),'descend');
```

Determine the rank of each row in the original order for the first three principal components and store the ranks in a matrix pcarankmatrix.

```
[x y z]=intersect(1:length(y1),y1);
pcarankmatrix(:,1)=z;
[x y z]=intersect(1:length(y2),y2);
pcarankmatrix(:,2)=z;
[x y z]=intersect(1:length(y3),y3);
pcarankmatrix(:,3)=z;

%contains the rank of each feature in the original order for the first three principal compo

dlmwrite('pcarankmatrix.txt', pcarankmatrix,'delimiter','\t')
```

To efficiently manage our data with a single glance, I have organized it into an Excel spreadsheet using a combination of command line, Excel, and R.

Spreadsheet The first sheet (patients) built on excel contains patients order, patients id, phenotypes, and sex. You can visit the sheet by clicking here.



	ID	Phenotype	Gender
10	082	Fibrotic	F
12	324	Fibrotic	M
13	483	Fibrotic	M
17	532	Fibrotic	M
11	551	Fibrotic	F
19	748	Fibrotic	M
10	805	Fibrotic	M
14	823	Fibrotic	M
15	860	Fibrotic	F
19	1294	Fibrotic	M
6	008	Non-fibrotic	F
3	185	Non-fibrotic	F
4	294	Non-fibrotic	M
2	417	Non-fibrotic	M
7	496	Non-fibrotic	F
5	508	Non-fibrotic	F
8	527	Non-fibrotic	F
1	798	Non-fibrotic	M
9	830	Non-fibrotic	M

1	008P22TGFbM_571	N	F	TG_008_N_F_22		TG_008_N_F_22	TG	8	N	F	22	TG_N	8_22
2	008P22TNFa-TGFbM_572	N	F	TT_008_N_F_22		TT_008_N_F_22	TT	8	N	F	22	TT_N	8_22
3	008P22TNFaM_570	N	F	TN_008_N_F_22		TN_008_N_F_22	TN	8	N	F	22	TN_N	8_22
4	008P22untreatedM_569	N	F	CC_008_N_F_22		CC_008_N_F_22	CC	8	N	F	22	CC_N	8_22
5	008P33TGFbM_575	N	F	TG_008_N_F_33		TG_008_N_F_33	TG	8	N	F	33	TG_N	8_33
6	008P33TNFa-TGFbM_576	N	F	TT_008_N_F_33		TT_008_N_F_33	TT	8	N	F	33	TT_N	8_33
7	008P33TNFaM_559	N	F	TN_008_N_F_33		TN_008_N_F_33	TN	8	N	F	33	TN_N	8_33
8	008P33untreatedM_573	N	F	CC_008_N_F_33		CC_008_N_F_33	CC	8	N	F	33	CC_N	8_33
9	082P52TGFbM_561	F	F	TG_082_F_F_52		TG_082_F_F_52	TG	82	F	F	52	TG_F	82_52
10	082P52TNFa-TGFbM_580	F	F	TT_082_F_F_52		TT_082_F_F_52	TT	82	F	F	52	TT_F	82_52
11	082P52TNFaM_560	F	F	TN_082_F_F_52		TN_082_F_F_52	TN	82	F	F	52	TN_F	82_52
12	082P52untreatedM_577	F	F	CC_082_F_F_52		CC_082_F_F_52	CC	82	F	F	52	CC_F	82_52
13	082p42TGFbM_5129	F	F	TG_082_F_F_42		TG_082_F_F_42	TG	82	F	F	42	TG_F	82_42
14	082p42TNFa-TGFbM_5130	F	F	TT_082_F_F_42		TT_082_F_F_42	TT	82	F	F	42	TT_F	82_42
15	082p42TNFaM_5128	F	F	TN_082_F_F_42		TN_082_F_F_42	TN	82	F	F	42	TN_F	82_42
16	082p42untreatedM_5127	F	F	CC_082_F_F_42		CC_082_F_F_42	CC	82	F	F	42	CC_F	82_42
17	1294P45TGFbM_557	F	M	TG_1294_F_M_45		TG_1294_F_M_45	TG	1294	F	M	45	TG_F	1294_45
18	1294P45TNFa-TGFbM_564	F	M	TT_1294_F_M_45		TT_1294_F_M_45	TT	1294	F	M	45	TT_F	1294_45
19	1294P45TNFaM_556	F	M	TN_1294_F_M_45		TN_1294_F_M_45	TN	1294	F	M	45	TN_F	1294_45
20	1294P45untreatedM_561	F	M	CC_1294_F_M_45		CC_1294_F_M_45	CC	1294	F	M	45	CC_F	1294_45
21	1294P52TGFbM_567	F	M	TG_1294_F_M_52		TG_1294_F_M_52	TG	1294	F	M	52	TG_F	1294_52
22	1294P52TNFa-TGFbM_568	F	M	TT_1294_F_M_52		TT_1294_F_M_52	TT	1294	F	M	52	TT_F	1294_52
23	1294P52TNFaM_558	F	M	TN_1294_F_M_52		TN_1294_F_M_52	TN	1294	F	M	52	TN_F	1294_52
24	1294P52untreatedM_565	F	M	CC_1294_F_M_52		CC_1294_F_M_52	CC	1294	F	M	52	CC_F	1294_52
25	185P20TGFbM_583	N	F	TG_185_N_F_20		TG_185_N_F_20	TG	185	N	F	20	TG_N	185_20
26	185P20TNFa-TGFbM_584	N	F	TT_185_N_F_20		TT_185_N_F_20	TT	185	N	F	20	TT_N	185_20
27	185P20TNFaM_582	N	F	TN_185_N_F_20		TN_185_N_F_20	TN	185	N	F	20	TN_N	185_20
28	185P20untreatedM_581	N	F	CC_185_N_F_20		CC_185_N_F_20	CC	185	N	F	20	CC_N	185_20
29	185p14TGFbM_5133	N	F	TG_185_N_F_14		TG_185_N_F_14	TG	185	N	F	14	TG_N	185_14
30	185p14TNFa-TGFbM_5134	N	F	TT_185_N_F_14		TT_185_N_F_14	TT	185	N	F	14	TT_N	185_14
31	185p14TNFaM_5132	N	F	TN_185_N_F_14		TN_185_N_F_14	TN	185	N	F	14	TN_N	185_14
32	185p14untreatedM_5131	N	F	CC_185_N_F_14		CC_185_N_F_14	CC	185	N	F	14	CC_N	185_14
33	294P26TGFbM_587	N	M	TG_294_N_M_26		TG_294_N_M_26	TG	294	N	M	26	TG_N	294_26
34	294P26TNFa-TGFbM_588	N	M	TT_294_N_M_26		TT_294_N_M_26	TT	294	N	M	26	TT_N	294_26
35	294P26TNFaM_586	N	M	TN_294_N_M_26		TN_294_N_M_26	TN	294	N	M	26	TN_N	294_26
36	294P26untreatedM_585	N	M	CC_294_N_M_26		CC_294_N_M_26	CC	294	N	M	26	CC_N	294_26
37	294P27TGFbM_566	N	M	TG_294_N_M_27		TG_294_N_M_27	TG	294	N	M	27	TG_N	294_27
38	294P27TNFa-TGFbM_595	N	M	TT_294_N_M_27		TT_294_N_M_27	TT	294	N	M	27	TT_N	294_27
39	294P27TNFaM_562	N	M	TN_294_N_M_27		TN_294_N_M_27	TN	294	N	M	27	TN_N	294_27
40	294P27untreatedM_570	N	M	CC_294_N_M_27		CC_294_N_M_27	CC	294	N	M	27	CC_N	294_27
41	324P24TGFbM_565	F	M	TG_324_F_M_24		TG_324_F_M_24	TG	324	F	M	24	TG_F	324_24
42	324P24TNFa-TGFbM_593	F	M	TT_324_F_M_24		TT_324_F_M_24	TT	324	F	M	24	TT_F	324_24
43	324P24TNFaM_564	F	M	TN_324_F_M_24		TN_324_F_M_24	TN	324	F	M	24	TN_F	324_24
44	324P24untreatedM_563	F	M	CC_324_F_M_24		CC_324_F_M_24	CC	324	F	M	24	CC_F	324_24
45	324P26TGFbM_5137	F	M	TG_324_F_M_26		TG_324_F_M_26	TG	324	F	M	26	TG_F	324_26
46	324P26TNFa-TGFbM_5138	F	M	TT_324_F_M_26		TT_324_F_M_26	TT	324	F	M	26	TT_F	324_26
47	324P26TNFaM_5136	F	M	TN_324_F_M_26		TN_324_F_M_26	TN	324	F	M	26	TN_F	324_26
48	324P26untreatedM_5135	F	M	CC_324_F_M_26		CC_324_F_M_26	CC	324	F	M	26	CC_F	324_26

The second sheet (allbiotypes_percents) includes the names and percentages of all biotypes, along with their respective minimum, maximum, and average values, providing us with a comprehensive overview. You can visit the sheet by clicking here.

```
#In terminal
paste allbiotypes allbiotypescountspercents > combine_allbiotypes_percents.txt

#In R#allbiotypes_percentssheet2_1 <- list("Biotypes")sheet2_2 <- sampleKeyTNFATGFBcombined_
#add their respective minimum, maximum, and average values on Excel
```

Biotypes	min	max	average	TG_008_N_F_22	TT_008_N_F_22	TN_008_N_F_22	CC_008_N_F_22	TG_008_N_F_33	TT_008_N_F_33	TN_008_N_F_33	CC_008_N_F_33
IG_C_gene	0.000000000	0.000446580	0.000013340	0.000000000	0.000000000	0.000000000	0.000000000	0.000003012	0.000011577	0.000000000	0.000000000
IG_D_gene	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
IG_J_gene	0.000000000	0.000013813	0.000000235	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
IG_V_gene	0.000000000	0.000073058	0.000010604	0.000012755	0.000010996	0.000008912	0.000002556	0.000015061	0.000008269	0.000010462	0.000011520
Mt_rRNA	0.228350000	1.165600000	0.487934371	0.495080000	0.584550000	0.439920000	0.826140000	0.508050000	0.550660000	0.508120000	0.859590000
Mt_tRNA	0.000141280	0.002049800	0.000492376	0.000214290	0.000274910	0.000199040	0.000345110	0.000240970	0.000219970	0.000237130	0.000336390
TEC	0.011838000	0.029033000	0.018976556	0.017090000	0.016613000	0.016773000	0.021052000	0.017133000	0.019370000	0.017489000	0.021066000
TR_C_gene	0.000000000	0.002667100	0.000287637	0.000002551	0.000000000	0.000005942	0.000000000	0.000018073	0.000011577	0.000087182	0.000009216
TR_D_gene	0.000000000	0.000011542	0.000000580	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
TR_J_gene	0.000000000	0.000076445	0.000009219	0.000000000	0.000000000	0.000000000	0.000010225	0.000009036	0.000000000	0.000006975	0.000002304
TR_V_gene	0.000000000	0.000061956	0.000011904	0.000000000	0.000008247	0.000005942	0.000010225	0.000003012	0.000013231	0.000006975	0.000016129
lncRNA	0.537580000	1.612500000	0.825408079	0.832110000	0.782160000	0.759180000	1.060300000	0.858370000	0.828130000	0.792760000	1.102900000
miRNA	0.002352100	0.014811000	0.004202784	0.004000100	0.004865900	0.004536300	0.003474100	0.003707900	0.004582900	0.005147200	0.003327100
misc_RNA	0.000641580	0.002443400	0.001048973	0.001140300	0.001300300	0.001265500	0.001250100	0.001015100	0.001283400	0.001227500	0.001101300
protein_codi	**97.675000000**	**99.144000000**	**98.658132450**	**98.647000000**	**98.607000000**	**98.775000000**	**98.084000000**	**98.608000000**	**98.592000000**	**98.671000000**	**98.008000000**
rRNA	0.000000000	0.000028828	0.000005066	0.000000000	0.000008247	0.000008912	0.000000000	0.000003012	0.000000000	0.000010462	0.000002304
ribozyme	0.000000000	0.000007357	0.000000293	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
sRNA	0.000000000	0.000004680	0.000000135	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000001654	0.000000000	0.000000000
scRNA	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
scaRNA	0.000002616	0.000324330	0.000036518	0.000012755	0.000024742	0.000062386	0.000040902	0.000024097	0.000031424	0.000027898	0.000052994
snRNA	0.000262070	0.001464200	0.000543351	0.000413270	0.000519580	0.000534740	0.000682550	0.000590380	0.000583820	0.000606780	0.000580630
snoRNA	0.001727500	0.004335800	0.002883131	0.002719400	0.002650100	0.002857900	0.003037000	0.002970000	0.003486400	0.002978100	0.003205000
vaultRNA	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000

The third sheet is the main sheet that includes Genename, Geneid, Mapp, PC1, PC2, PC3, and patient's TPM values.

```
#In terminal

paste Gencode_33_Selected_Genename_GMask.txt Gencode_33_Selected_Genename_GMask.txt Gencode_
#In R
```

```
#spreadsheet
sheet3_1 <- list("Genename","Genename","Geneid","Mapp","PC1","PC2","PC3")sheet3_2<- sampleKe
```

You can sort this sheet with PC1, PC2, and so on to see the corelation between the treatment and the expression level in each gene.

Genename	Genename	Geneid	Mapp	PC1	PC2	PC3	TG_008_N_F_22	TT_008_N_F_22	TN_008_N_F_22	CC_008_N_F_22	TG_008_N_F_33	TT_008_N_F_33	TN_008_N_F_33	CC_008_N_F_33	TG_082_F_F	TT_082_F_F	TN_082_F_F	CC_082_F_F	TG_082_F_F	TT_082_F_F	TN_082_F_F	CC_082_F_F
CXCL6	CXCL6	ENSG00000124875.10	1596.00	68.00	1.00	887.00	32.65	10587.00	5693.30	31.87	32.82	4913.60	870.45	33.79	13.20	4292.20	149.95	124.15	1.46	4126.20	943.95	7.07
CXCL8	CXCL8	ENSG00000169429.11	1868.00	1.00	2.00	389.00	0.24	4089.30	1728.20	0.40	0.29	1642.70	747.49	0.36	3.21	2633.80	804.62	7.38	0.67	8435.60	3672.60	0.48
CXCL1	CXCL1	ENSG00000163739.5	1075.00	111.00	3.00	1782.00	15.88	5467.70	2412.60	18.21	12.37	1673.10	351.36	19.33	17.31	2740.30	143.20	76.39	3.44	11957.00	1262.50	32.73
C3	C3	ENSG00000125730.17	5851.00	21.00	4.00	153.00	0.50	944.99	236.21	0.64	0.09	310.99	23.94	0.71	0.10	316.76	2.27	0.09	1.99	117.31	11.12	18.46
ABI3BP	ABI3BP	ENSG00000154175.17	9390.00	1164.00	5.00	1275.00	10.58	412.98	49.70	48.11	10.23	659.79	59.99	49.43	7.72	621.97	40.02	65.88	0.69	15.85	4.66	3.53
COMP	COMP	ENSG00000105664.11	2779.00	147.00	6.00	2037.00	49.71	0.04	67.18	0.08	34.12	0.19	102.56	0.00	54.69	0.00	33.59	0.00	134.30	0.11	12.07	0.12
ELN	ELN	ENSG00000049540.17	5524.00	5644.00	7.00	1132.00	33.42	3.41	15.19	0.64	13.08	0.78	5.12	0.59	64.26	0.31	1.65	0.48	1012.30	12.42	20.23	11.67
CXCL3	CXCL3	ENSG00000163734.4	975.00	118.00	8.00	795.00	3.01	1973.70	799.40	1.76	3.72	585.33	93.03	0.89	0.50	514.44	11.66	1.84	1.16	3789.20	322.23	4.70
SOD2	SOD2	ENSG00000112096.18	13791.00	56.00	9.00	1172.00	12.85	1272.70	798.31	23.34	8.93	913.94	202.64	22.62	8.31	1386.80	84.31	19.33	9.63	780.30	230.05	43.23
CXCL5	CXCL5	ENSG00000163735.7	2337.00	81.00	10.00	550.00	0.34	869.40	225.78	0.64	0.17	151.01	14.67	0.17	0.00	95.85	2.82	0.05	0.31	1967.60	87.29	0.81
CCL5	CCL5	ENSG00000271503.6	1372.00	3.00	11.00	321.00	0.16	295.41	223.72	0.00	0.29	549.51	161.46	0.07	0.14	456.50	171.04	0.00	0.04	94.00	833.98	0.00
PTX3	PTX3	ENSG00000163661.4	1781.00	3156.00	12.00	4555.00	1386.10	9236.90	3935.60	2258.20	184.73	6763.80	768.90	2271.60	55.27	3225.10	140.89	150.15	41.85	468.01	43.16	1158.50
CCL2	CCL2	ENSG00000108691.9	1970.00	4.00	13.00	172.00	1.72	1043.10	834.13	3.22	1.09	1280.90	806.48	3.09	4.20	1968.40	1513.90	5.13	0.30	542.81	292.03	3.17
CXCL2	CXCL2	ENSG00000081041.9	1016.00	333.00	14.00	1285.00	1.45	904.33	356.30	4.76	1.59	304.28	32.38	3.23	0.68	368.55	3.36	2.07	1.00	1236.60	140.94	9.89
VCAM1	VCAM1	ENSG00000162694.12	3283.00	591.00	15.00	1695.00	82.51	1391.60	462.26	286.87	24.01	2666.10	431.79	294.77	18.17	2822.90	343.05	256.84	7.87	238.71	61.08	99.14
MX1	MX1	ENSG00000157601.14	4555.00	413.00	16.00	495.00	1.94	278.74	62.79	2.31	2.09	116.67	6.18	2.21	0.61	487.83	9.96	15.53	1.48	37.89	22.66	19.59
IFI6	IFI6	ENSG00000126709.15	921.00	135.00	17.00	636.00	197.01	11438.00	3923.40	263.68	106.44	4058.10	387.41	265.10	102.71	11933.00	705.00	290.36	68.81	1217.60	878.34	214.17
TPK3	TPK3	ENSG00000073282.13	6942.00	1074.00	18.00	434.00	0.13	74.20	3.17	0.37	0.54	76.34	8.95	0.22	0.10	139.07	0.62	3.43	0.09	0.18	0.08	0.02
ICAM1	ICAM1	ENSG00000090339.9	3063.00	20.00	19.00	1288.00	5.06	1045.60	430.16	27.88	2.54	1143.40	413.10	29.38	27.86	1830.80	1044.70	129.72	7.40	482.19	926.08	50.29
BST2	BST2	ENSG00000130303.13	900.00	145.00	20.00	390.00	12.06	1170.20	201.45	14.94	9.55	531.86	71.06	12.88	0.55	1587.70	7.71	5.38	2.72	54.49	50.72	27.43
OAS3	OAS3	ENSG00000111331.13	7694.00	127.00	21.00	471.00	1.50	325.87	39.64	2.71	0.54	65.66	9.59	1.97	0.12	190.81	11.51	2.29	0.73	23.47	22.70	8.72
TRPA1	TRPA1	ENSG00000104321.11	5092.00	88.00	22.00	1008.00	0.04	57.40	30.22	0.04	0.03	98.37	10.63	0.00	0.10	35.51	0.33	0.02	0.93	359.15	55.77	3.51
MMP1	MMP1	ENSG00000196611.5	1864.00	17.00	23.00	236.00	3.15	377.17	978.50	0.52	3.24	141.31	83.54	1.24	0.05	105.16	2.07	0.00	0.26	2582.70	252.83	0.54
OAS1	OAS1	ENSG00000089127.13	4894.00	316.00	24.00	494.00	0.25	83.85	21.32	1.32	0.33	38.14	2.10	1.84	0.10	149.43	3.93	2.78	1.11	14.71	15.05	14.70
BOKR2	BOKR2	ENSG00000168398.6	6478.00	5115.00	25.00	2145.00	3.46	299.40	167.19	35.96	3.17	169.33	146.70	35.20	1.84	550.72	12.32	119.07	9.39	297.21	165.28	64.21
TNFAIP2	TNFAIP2	ENSG00000185213.9	4879.00	47.00	26.00	768.00	2.90	480.19	114.66	7.49	2.45	324.42	59.14	7.25	8.45	493.27	60.84	6.93	0.72	108.12	66.09	2.27
PTGES	PTGES	ENSG00000148344.11	1648.00	13160.00	27.00	927.00	22.78	937.77	605.45	120.14	6.60	201.08	56.81	130.62	1.55	477.40	9.66	25.17	3.77	398.15	38.91	38.84
OAS2	OAS2	ENSG00000111335.12	6417.00	205.00	28.00	304.00	0.05	56.65	6.50	0.07	0.10	27.13	0.60	0.08	0.05	109.23	0.55	0.15	0.10	1.54	1.64	0.82
TNFAIP3	TNFAIP3	ENSG00000118503.15	5175.00	16.00	29.00	1485.00	11.01	508.75	471.41	12.45	12.07	527.70	324.42	12.31	5.71	743.53	304.41	9.60	7.48	586.30	359.82	18.54
COL10A1	COL10A1	ENSG00000123500.10	3719.00	426.00	30.00	258.00	14.14	0.84	4.44	0.23	22.50	0.71	19.39	0.29	0.45	0.00	0.28	0.00	0.22	2.40	4.16	0.00
C7	C7	ENSG00000112936.19	5583.00	1080.00	31.00	8010.00	35.94	325.09	80.53	57.83	26.44	99.34	16.41	57.58	77.74	1392.20	45.82	500.07	1.89	0.91	0.22	76.14
EPSTI1	EPSTI1	ENSG00000133106.14	3387.00	1787.00	32.00	1636.00	16.06	351.20	178.25	30.64	6.82	211.47	49.20	26.46	4.30	277.19	18.16	38.45	1.66	25.39	13.08	10.65
ANO51	ANO51	ENSG00000111201.12	5932.00	978.00	33.00	1324.00	85.09	2.89	35.76	0.63	241.05	9.27	50.30	0.80	219.43	1.02	58.02	1.15	156.78	3.19	5.04	13.22
SLC7A2	SLC7A2	ENSG00000003989.17	8019.00	162.00	34.00	1464.00	10.18	203.92	79.47	60.58	1.72	103.64	82.66	56.99	1.31	94.47	279.42	50.21	5.15	405.05	797.25	37.41
CRLF1	CRLF1	ENSG00000006016.11	1713.00	139.00	35.00	1053.00	104.67	0.13	228.89	0.63	52.76	0.46	230.10	0.62	103.95	0.19	138.65	0.74	776.71	8.46	777.11	1.70
SLC39A8	SLC39A8	ENSG00000138821.13	5772.00	338.00	36.00	634.00	4.03	522.69	96.33	2.46	3.16	143.62	19.82	2.39	0.84	233.80	5.71	3.36	1.61	31.61	11.27	2.46
BIRC3	BIRC3	ENSG00000023445.15	7397.00	8.00	37.00	468.00	0.20	43.90	23.12	0.22	0.05	47.92	19.87	0.16	0.12	49.16	19.24	0.23	0.08	30.37	29.72	0.44
CLDN11	CLDN11	ENSG00000013297.11	1477.00	44.00	38.00	3725.00	318.25	852.33	339.26	2145.20	54.92	748.84	39.03	2166.10	29.52	300.67	14.63	1372.10	18.26	277.48	6.36	1461.40
CXCL10	CXCL10	ENSG00000169245.6	1076.00	336.00	39.00	406.00	0.00	69.21	10.80	0.00	0.00	107.42	16.86	0.00	0.00	365.14	1.06	0.20	0.00	12.87	6.02	0.00
IFIT3	IFIT3	ENSG00000119917.14	2424.00	3432.00	40.00	1342.00	24.02	301.95	73.64	59.05	8.70	180.05	25.11	53.31	5.28	700.96	21.53	96.16	2.61	47.62	45.89	38.15
IFI44L	IFI44L	ENSG00000137959.16	6048.00	539.00	41.00	423.00	4.36	198.48	48.30	4.59	2.29	96.52	5.01	5.24	2.67	304.69	7.35	3.97	0.24	11.16	7.05	2.29
IL32	IL32	ENSG00000008517.16	2856.00	2.00	42.00	1031.00	0.20	194.89	142.13	0.87	5.08	451.84	329.95	0.27	3.48	492.91	463.17	4.06	4.91	114.73	305.85	3.75
LOXL4AD4	LOXL4AD4	ENSG00000138675.18	11732.00	686.00	43.00	981.00	0.65	0.02	0.59	0.02	4.03	0.36	7.21	0.01	19.99	0.12	16.14	3.37	24.01	0.46	14.29	0.25
SPX	SPX	ENSG00000134548.11	2268.00	1773.00	44.00	4460.00	41.82	662.12	115.00	146.01	12.20	111.61	32.33	143.45	0.30	89.01	2.96	4.22	0.42	1.83	0.46	1.18
CTSS	CTSS	ENSG00000163131.11	3937.00	94.00	45.00	604.00	0.37	86.80	24.61	0.90	0.20	84.66	16.63	1.25	0.93	129.91	15.59	2.78	0.59	79.83	36.88	1.16
CHI3L2	CHI3L2	ENSG00000064886.14	3501.00	224.00	46.00	1133.00	0.19	31.15	33.51	0.46	0.15	28.93	7.60	0.58	0.06	59.92	0.81	0.42	0.17	7.14	1.01	0.73
IL6	IL6	ENSG00000136244.12	1764.00	42.00	47.00	1049.00	0.26	432.09	40.78	0.00	0.23	63.80	1.31	0.22	0.67	117.20	13.02	0.12	0.27	330.18	47.31	0.51
TNFRSF1B	TNFRSF1B	ENSG00000028137.19	3154.00	4819.00	48.00	608.00	3.01	88.89	28.39	6.97	1.03	89.14	5.92	7.44	0.39	88.43	0.64	3.78	1.00	27.89	2.10	9.13
IFIT1	IFIT1	ENSG00000185745.10	4607.00	3926.00	49.00	1883.00	35.58	148.21	65.03	30.02	10.67	70.47	10.24	28.47	1.79	223.42	3.21	34.19	1.32	12.06	9.07	40.66

Future works

References

1. Dovrolis, N., et al., Co-expression of fibrotic genes in inflammatory bowel disease; A localized event? Front Immunol, 2022. 13: p. 1058237.
2. Edgar, R.D., et al., Culture-Associated DNA Methylation Changes Impact on Cellular Function of Human Intestinal Organoids. Cell Mol Gastroenterol Hepatol, 2022. 14(6): p. 1295-1310.
3. Ihara, S., Y. Hirata, and K. Koike, TGF-beta in inflammatory bowel disease: a key regulator of immune cells, epithelium, and the intestinal microbiota. J Gastroenterol, 2017. 52(7): p. 777-787.
4. Lindeboom, R.G., et al., Integrative multi-omics analysis of intestinal organoid differentiation. Mol Syst Biol, 2018. 14(6): p. e8227.
5. Ma, Q., et al., OrganoidDB: a comprehensive organoid database for the multi-perspective exploration of bulk and single-cell transcriptomic profiles of organoids. Nucleic Acids Res, 2023. 51(D1): p. D1086-D1093.
6. Wang, Q., et al., Applications of human organoids in the personalized treatment for digestive diseases. Signal Transduct Target Ther, 2022. 7(1): p. 336.
7. Corsini, N.S. and J.A. Knoblich, Human organoids: New strategies and methods for analyzing human development and disease. Cell, 2022. 185(15): p. 2756-2769.

8. Ingber, D.E., Human organs-on-chips for disease modelling, drug development and personalized medicine. Nat Rev Genet, 2022. 23(8): p. 467-491.
9. Brooks, I.R., et al., Functional genomics and the future of iPSCs in disease modeling. Stem Cell Reports, 2022. 17(5): p. 1033-1047.
10. D'Alessio, S., et al., Revisiting fibrosis in inflammatory bowel disease: the gut thickens. Nat Rev Gastroenterol Hepatol, 2022. 19(3): p. 169-184.
11. Carcamo-Orive, I., et al., Analysis of Transcriptional Variability in a Large Human iPSC Library Reveals Genetic and Non-genetic Determinants of Heterogeneity. Cell Stem Cell, 2017. 20(4): p. 518-532 e9.
12. Gleeson, J.P., et al., Development of Physiologically Responsive Human iPSC-Derived Intestinal Epithelium to Study Barrier Dysfunction in IBD. Int J Mol Sci, 2020. 21(4).
13. Workman, M.J., et al., Modeling Intestinal Epithelial Response to Interferon-gamma in Induced Pluripotent Stem Cell-Derived Human Intestinal Organoids. Int J Mol Sci, 2020. 22(1).