## 一、准备单个表达矩阵，同时制做表格B（datTraits.csv）

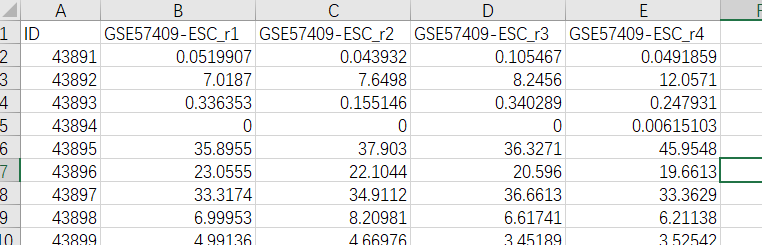
根据数据汇总的表格，挑选自己需要的数据库：



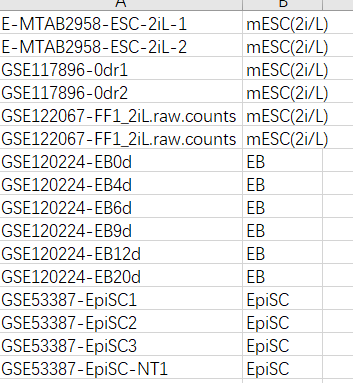
从文件夹内找到相应的excel或csv格式的一系列表格A，打开：

（1）删除非必须列：注意，表格内的样本不一定全部是我们想要的，故删除非必须的，留下我们感兴趣的样本列。

（2）统一整理样本名名称（序列号\_样本名称）：



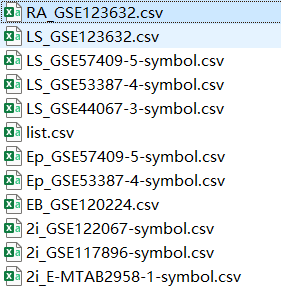
（3）同时将上述名称复制粘贴到另一个新建csv**表格B（datTraits.csv）**中：

（第一列为样本名称，第二列为分类名称）

如此操作方便之后查看时，方便对应到哪些样本。

（4）查看表格基因名是symbol还是ENSMUSG开头的

（5）表格A统一改名（名字为**“‘分类’\_’序列号’\_symbol’**”）另存为csv格式（这样可保证之前数据内容不要改动）。



## 二、批量表格，转换symbol

由于gene symbol即使同一个基因，也会有不同的名称，故我们统一转换为ENSMUSG开头的基因ID。此时，我们将文件名标有“symbol”单独纳入新建文件夹，统一将上述表格转换ID。

setwd("/disk\_HD\_1/wangjiaqi/R/task/symbol")

#step1:读取当前文件的所有的csv名字

temp = list.files(pattern="\*.csv")

library(clusterProfiler)

library(dplyr)

#step2:循环语句批量更换基因ID

for (i in 1:length(temp)){

#a是读取单个csv

assign("a",read.csv(temp[i],,sep = ","))

names(a)[1]<-"id"

gene<- bitr(a$id, fromType="SYMBOL", toType="ENSEMBL", OrgDb="org.Mm.eg.db")

names(gene)[1]<-"id"

#b是转换了ID的a

b<-inner\_join(a,gene,by="id") %>%

dplyr::select(ENSEMBL,everything())

b<-b[,-2]

#c是去除b中ENSEMBL列的重复项，并将第一列基因ID变成rownames，且去除基因ID（此操作是为了后续方便导入）

c<-distinct(b,ENSEMBL,.keep\_all = T)

rownames(c)<-c$ENSEMBL

c<-c[,-1]

write.csv(c,paste("new.",temp[i],sep = "")) #导出时，在之前的文件名之前加入“new”的名称方便识别。

}

## 三、合并表格,制作表格C (total.csv)

3.1将同一类型的表达矩阵先合并

3.11将同一类型的放同一新建文件夹：

3.12 在R上分别合并（以2i的合并为例）：

setwd("/disk\_HD\_1/wangjiaqi/R/task/**2i/**")

#step1:读取当前文件的所有的csv名字

temp = list.files(pattern="\*.csv")

library(clusterProfiler)

library(dplyr)

#step2：先读取第一个文件

assign("a",read.csv(temp[1],,sep = ","))

names(a)[1]<-"ID"

#step3：将第一个文件分别单独和其他文件合并，判断是否有哪些文件使得交集基因较少（一般少于17000,则需要考虑是否需要这个样本？如果很重要，则需要从SRA数据）

for (i in 2:length(temp)){

assign("b",read.csv(temp[i],,sep = ","))

names(b)[1]<-"ID"

c<-inner\_join(a,b,by="ID") #注意第一个文件分别和其他文件合并是c<-inner\_join(a,b,by="ID")

assign(paste("G",i,sep = ""),c)

}

#发现G2交集只有14000，说明第二个基因集基因较少，故省去。

#step4：

for (i in 3:length(temp)){

i=3

assign("b",read.csv(temp[i],,sep = ","))

names(b)[1]<-"ID"

a<-inner\_join(a,b,by="ID") #注意所有文件合并是a<-inner\_join(a,b,by="ID")

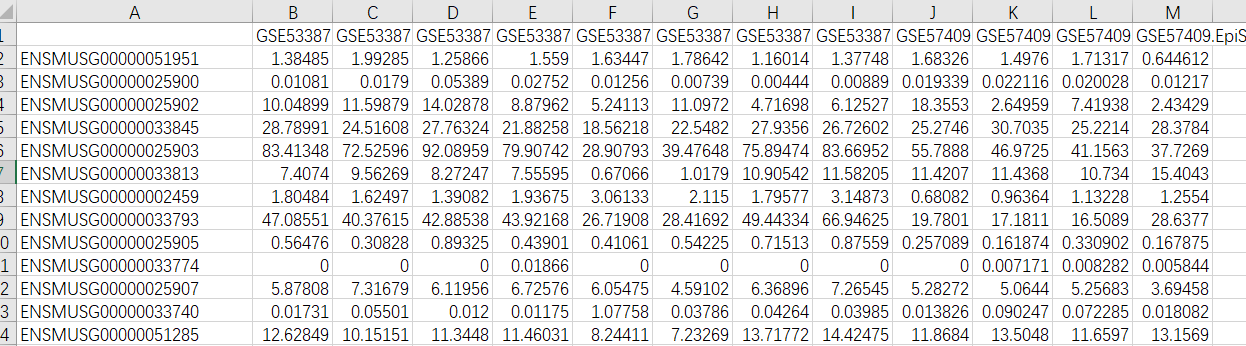
}

rownames(a)<-a$ID

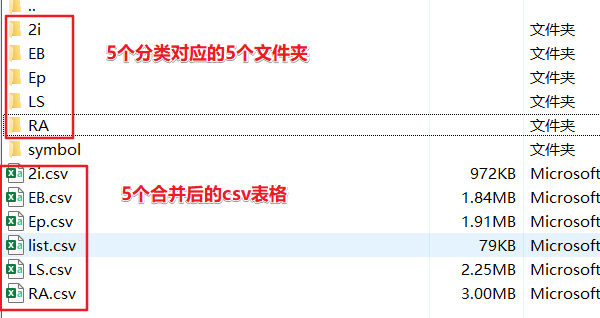
a<-a[,-1]

write.csv(a,"2i.csv")

最终每一个合并的表格为（以EpiSC为例，“Ep.csv”）：



最终效果：



3.2将每个大类的csv合并为一个**表格C（total.csv）**：

3.21新建一个文件夹total

3.22 R合并（和3.1代码几乎一样）：

setwd("/disk\_HD\_1/wangjiaqi/R/task/total/")

temp = list.files(pattern="\*.csv")

library(clusterProfiler)

library(dplyr)

#step2：先读取第一个文件

assign("a",read.csv(temp[1],,sep = ","))

names(a)[1]<-"ID"

#step3：将第一个文件分别单独和其他文件合并，判断是否有哪些文件使得交集基因较少（一般少于17000,则需要考虑是否需要这个样本？如果很重要，则需要从SRA数据）

for (i in 2:length(temp)){

assign("b",read.csv(temp[i],,sep = ","))

names(b)[1]<-"ID"

c<-inner\_join(a,b,by="ID") #注意第一个文件分别和其他文件合并是c<-inner\_join(a,b,by="ID")

assign(paste("G",i,sep = ""),c)

}

#发现G2交集只有14000，说明第二个基因集基因较少，故省去。

#step4：

for (i in 2:length(temp)){

assign("b",read.csv(temp[i],,sep = ","))

names(b)[1]<-"ID"

a<-inner\_join(a,b,by="ID") #注意所有文件合并是a<-inner\_join(a,b,by="ID")

}

#rm(a)

a1<-distinct(a,ID,.keep\_all = T) #这里可能合并以后有重合的ID，用distinct函数去重

rownames(a1)<-a1$ID

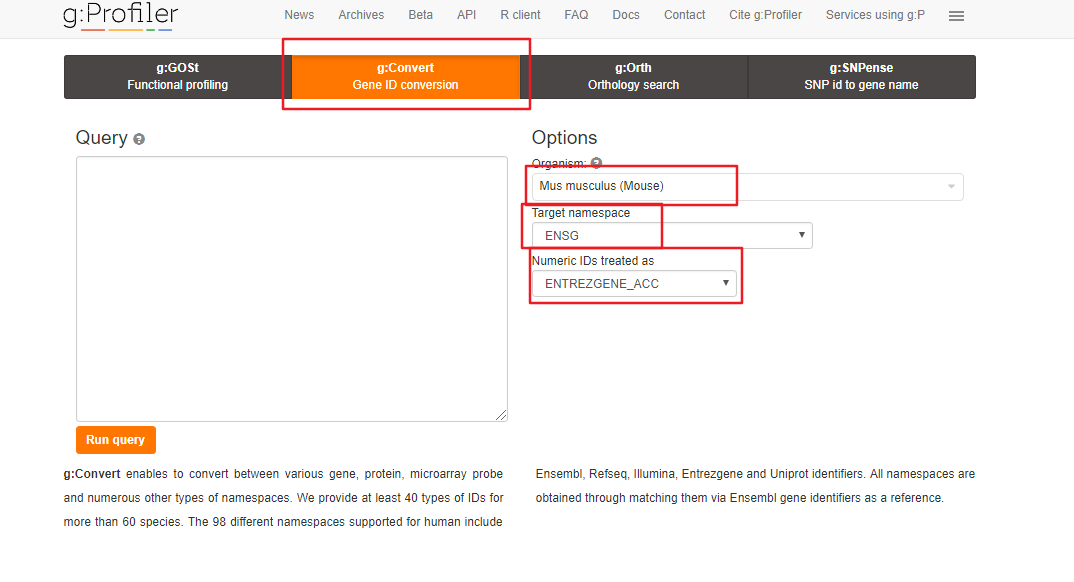
a1<-a1[,-1]

write.csv(a1,"total.csv")

3.23查看**表格C（total.csv）**的样本数和一开始**表格B（datTraits.csv）**是否是一致的，而且顺序必须要一致（顺序的混淆会导致后续分析的错位），在合并过程中由于有些样本的基因数过少，可能已经剔除掉，故需要在这里重新核对一遍。

## 四、准备表格D（list.csv）

WGCNA要求我们输入感兴趣的基因集，故我们需要从筛库或经过一定筛选后，提取出一个基因集。此时基因集的ID多为gene symbol，故我们同样需要转换ENSEMBL格式（ENSMUSG开头），此时我们建议用gprofiler的ID转换，此种转换基因数量损失量最小，故为了在分析前尽量减少我们的基因集的数量，故采取此方式，而上面所有表格的转换利用的R包（clusterProfiler），同样可实现，损失量相对高一点点，但是批量转换考虑效率问题，故不采取gprofiler网站的方式。



之后网站出结果后，另存为csv格式，提取出我们需要的ENSEMBL格式，导出为表格D（list.csv）.

接下来我们需要将total.csv取list.csv中的基因集：

setwd("/disk\_HD\_1/wangjiaqi/R/WGCNA/")

library(clusterProfiler)

library(dplyr)

#step1:导入list.csv

gene<-read.csv("list.csv")

names(gene)[1]<-"ID"

#step2：导入合并后的表达矩阵

total<-read.csv("total.csv")

names(total)[1]<-"ID"

#step3：合并

a<-inner\_join(total,gene,by="ID")

write.csv(a,"expreSet.csv")

得到**表格E（expreSet.csv）**

至此，我们WGCNA的两个输入文件得到了，分别是**expreSet.csv、datTraits.csv。**

**第二步 WGCNA:**

setwd("/home/wangjiaqi/data\_HD\_1/R/WGCNA/")

**#step1:导入datExpr和datTraits**

library(dplyr)

##1.1将data0整理为表达矩阵，样本名和基因名均为colnames和rownames

data0<-read.csv("expreSet.csv")

data0<-data0[,-1]

data0<-distinct(data0,ID,.keep\_all = T)

rownames(data0)<-data0$ID

data0<-data0[,-1]

##1.2 data0倒置后，导入datExpr

data01 = t(data0)

datExpr<-data01

write.csv(datExpr,"datExpr.csv")

##1.3导入datTraits

datTraits<-read.csv("datTraits.csv",header = F)

**#step2:确定最佳beta值**

#BiocManager::install("WGCNA") #安装WGCNA

library(WGCNA)

powers = c(c(1:10), seq(from = 12, to=20, by=2))

sft = pickSoftThreshold(datExpr, powerVector = powers, verbose = 5)

# Plot the results:

##sizeGrWindow(9, 5)

par(mfrow = c(1,2));

cex1 = 0.9;

# Scale-free topology fit index as a function of the soft-thresholding power

plot(sft$fitIndices[,1], -sign(sft$fitIndices[,3])\*sft$fitIndices[,2],

xlab="Soft Threshold (power)",ylab="Scale Free Topology Model Fit,signed R^2",type="n",

main = paste("Scale independence"));

text(sft$fitIndices[,1], -sign(sft$fitIndices[,3])\*sft$fitIndices[,2],

labels=powers,cex=cex1,col="red");

# this line corresponds to using an R^2 cut-off of h

abline(h=0.90,col="red")

# Mean connectivity as a function of the soft-thresholding power

plot(sft$fitIndices[,1], sft$fitIndices[,5],

xlab="Soft Threshold (power)",ylab="Mean Connectivity", type="n",

main = paste("Mean connectivity"))

text(sft$fitIndices[,1], sft$fitIndices[,5], labels=powers, cex=cex1,col="red")

**#step3:一步法构建共表达矩阵**

#对datExpr.csv的表格属性设置为“数字”后，再次读取

datExpr<-read.csv("datExpr.csv")

rownames(datExpr)<-datExpr$X

datExpr<-datExpr[,-1]

net = blockwiseModules(

datExpr,

power = sft$powerEstimate,

maxBlockSize = 6000,

TOMType = "unsigned", minModuleSize = 30,

reassignThreshold = 0, mergeCutHeight = 0.25,

numericLabels = TRUE, pamRespectsDendro = FALSE,

saveTOMs = TRUE,

saveTOMFileBase = "AS-green-FPKM-TOM",

verbose = 3

)

table(net$colors)

**#step4: 模块可视化**

# Convert labels to colors for plotting

mergedColors = labels2colors(net$colors)

table(mergedColors)

# Plot the dendrogram and the module colors underneath

plotDendroAndColors(net$dendrograms[[1]], mergedColors[net$blockGenes[[1]]],

"Module colors",

dendroLabels = FALSE, hang = 0.03,

addGuide = TRUE, guideHang = 0.05)

## assign all of the gene to their corresponding module

## hclust for the genes.

**#step5:模块和性状的关系**

#明确样本数和基因数

nGenes = ncol(datExpr)

nSamples = nrow(datExpr)

names(datTraits)[2]<-"subtype"

design=model.matrix(~0+ datTraits$subtype)

colnames(design)=levels(datTraits$subtype)

moduleColors <- labels2colors(net$colors)

# Recalculate MEs with color labels

MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes

MEs = orderMEs(MEs0); ##不同颜色的模块的ME值矩阵(样本vs模块)

moduleTraitCor = cor(MEs, design , use = "p");

moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples)

sizeGrWindow(10,6)

# Will display correlations and their p-values

textMatrix = paste(signif(moduleTraitCor, 2), "\n(",

signif(moduleTraitPvalue, 1), ")", sep = "");

dim(textMatrix) = dim(moduleTraitCor)

par(mar = c(6, 8.5, 3, 3));

# Display the correlation values within a heatmap plot

labeledHeatmap(Matrix = moduleTraitCor,

xLabels = colnames(design),

yLabels = names(MEs),

ySymbols = names(MEs),

colorLabels = FALSE,

colors = greenWhiteRed(50),

textMatrix = textMatrix,

setStdMargins = FALSE,

cex.text = 0.5,

zlim = c(-1,1),

main = paste("Module-trait relationships"))

**#step6:导出每个模块所对应的基因集**

**library(stringr)**

**a<-rownames(moduleTraitCor) #提取模块颜色**

**b<-str\_replace(a,"ME","") #去除前缀"ME"**

**for (i in 1:length(b)){**

**module = b[i] # Select module**

**# Select module probes**

**probes = colnames(datExpr) ## 我们例子里面的probe就是基因名**

**inModule = (moduleColors==module);**

**c = probes[inModule]; #c为每个模块的基因**

**write.csv(c,paste(b[i],".csv",sep=""))**

**}**

**热图（表达谱）绘制：**

setwd("/disk\_HD\_1/wangjiaqi/R/heatmap")

#Step1:导入数据

#1.1导入datTraits，并获得样本间的分组样本

datTraits<-read.csv("datTraits.csv",header = F)

summary(datTraits$V2)

colcolor<-rainbow(5, start = 0, end = .3)

ColSideColors<-c(rep(colcolor[1],4),rep(colcolor[2],6),rep(colcolor[3],12),rep(colcolor[4],16),rep(colcolor[5],9))

#1.2导入表达矩阵

data<-read.csv("datExpr.csv")

rownames(data)<-data$X

data<-data[,-1]

data1<-t(data)

#1.3批量导入module基因集，合并得到gene

setwd("/disk\_HD\_1/wangjiaqi/R/WGCNA/module/") #将所有的module放在一个新建文件夹里

temp = list.files(pattern="\*.csv")

library(stringr)

for (i in 1:length(temp)){

assign("a",read.csv(temp[i]))

b<-str\_replace(temp[i],".csv","")

c<-as.data.frame(as.character(a$x))

names(c)[1]<-"ID"

c$type<-b

assign(paste("G",i,sep=""),c)

}

G<-rbind(G1,G2,G3,G4,G5,G6,G7,G8,G9)

G$type<-as.factor(G$type)

gene<-as.character(G$ID)

#Step2：获得表达矩阵以及分组情况

#2.1获得表达矩阵的expreSet

setwd("/disk\_HD\_1/wangjiaqi/R/heatmap")

library(dplyr)

expreSet<-data1[rownames(data1) %in% gene,]

#--导出导入的目的是为了将expreSet的rownames放在第一列上

write.csv(expreSet,"expreSet.csv")

expreSet<-read.csv("expreSet.csv")

#--End

names(expreSet)[1]<-"ID"

#2.2获得含有ID、module分组情况的expreSet0，并提取基因module分组情况

expreSet0<-inner\_join(expreSet,G,by="ID")

dim(expreSet0)

expreSet01<-arrange(expreSet0,type)#根据type对expreSet0排序

rowcolor<-as.character(expreSet01$type)#获取基因module分组情况

#2.3整理expreSet01，重新获得表达矩阵expreSet02

expreSet02<-expreSet01[c(2: (length(expreSet01)-1))]

rownames(expreSet02)<-expreSet01$ID

expreSet03<-as.matrix(expreSet02) #热图的输入形式是numeric matrix

#Step3:画热图

library("gplots")

heatmap.2(expreSet03,

scale = "row",

Colv = NA,Rowv = NA,# 决定是否要将row按照系统发生树cluster的结果重新排序，注意要和dendrogram一致。

#margins = c(20,20),

col=greenred(75), #色块颜色的选择

density.info="none",

trace="none",# trace可以给每个色块中添加一条线，与行平行或者与列平行。其与色块中心的距离代表了这个值被显示的比例。

dendrogram = "none", # 生成row/col的系统发生树

ColSideColors=ColSideColors,

#RowSideColors =rep("yellow",nrow(expreSet))

RowSideColors = rowcolor

)

dev.off()