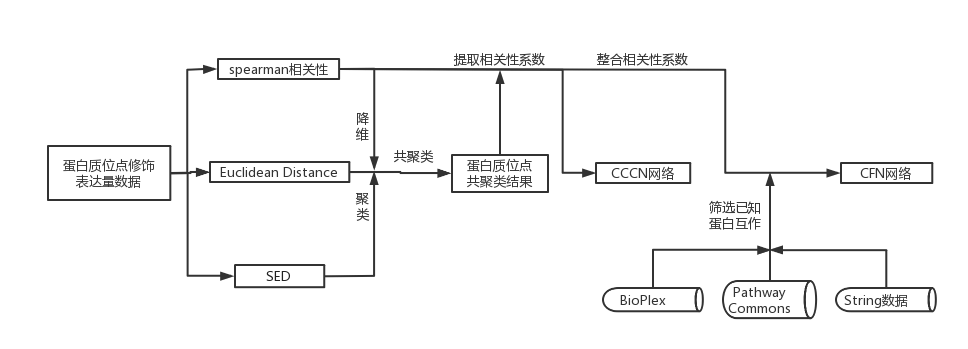
PTM修饰共聚类

1. 实验方法



使用R对癌症样本的PTM修饰位点数据计算成对完全欧氏距离（eu）、斯皮曼（sp）和混合斯皮曼-欧氏相似度（SED）。（gzdata.allt为PTM修饰位点数据）

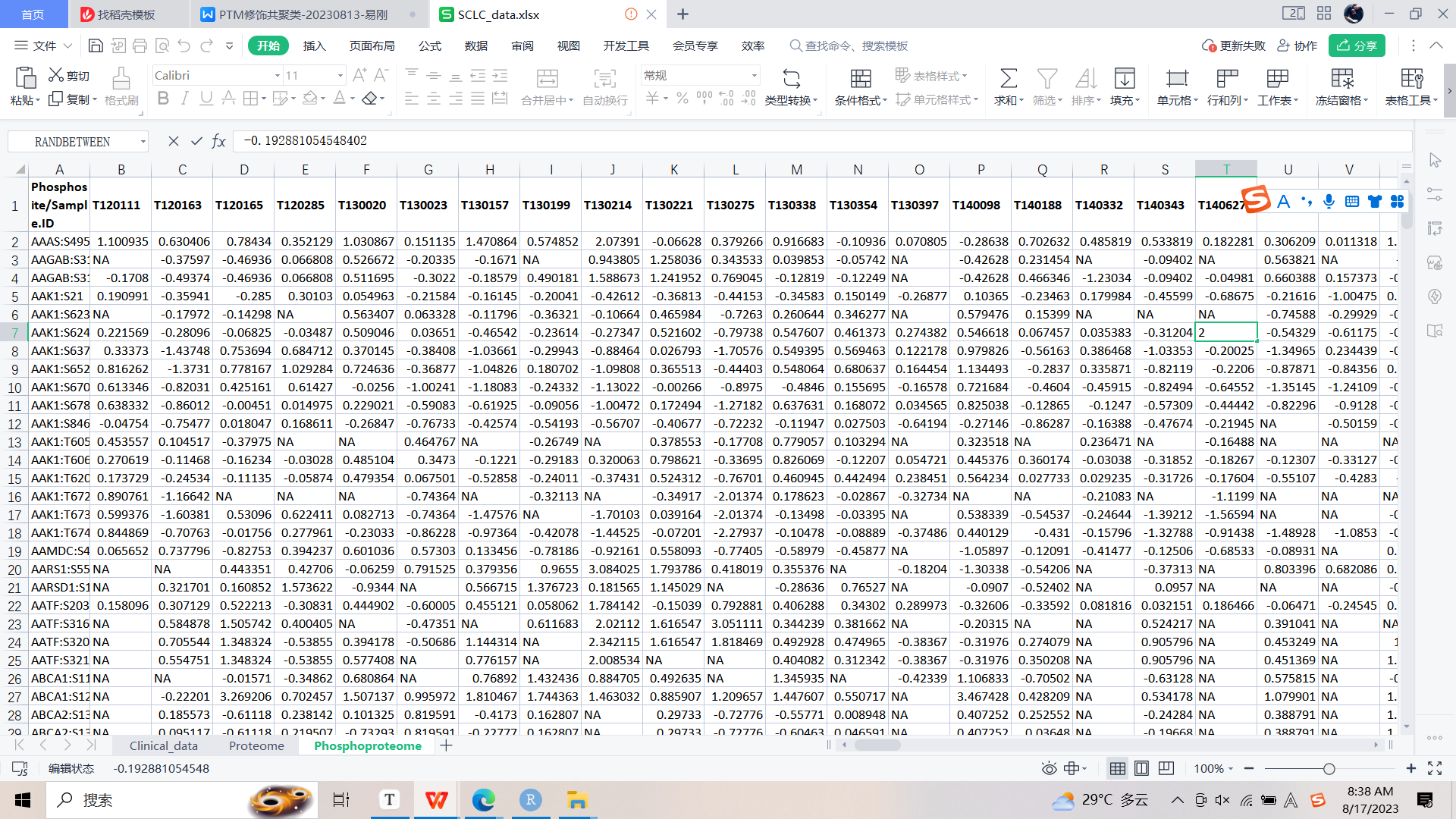
为了构建共聚类相关网络（CCCN），通过将共聚类修饰相互配对来构建邻接矩阵。Spearman相关值被用作网络边缘权重。

通过将所有共簇的PTM合并到被修饰蛋白质的基因名称中，该PTM CCCN被用于构建Protein CCCN。

得到的蛋白质CCCN被用于创建已知PPI相互作用的聚类过滤网络（CFN），该网络通过蛋白质之间的相互作用来进行过滤，但具有共聚类修饰的蛋白质除外。PPI数据集由STRING、GeneMANIA、BioPlex、Pathway Commons和PhosphositePlus的激酶-底物数据所描述的物理相互作用组成，重点关注直接相互作用。

1. 实验步骤
2. 数据介绍

|  |  |
| --- | --- |
| LSCC（肺鳞癌） | 磷酸化修饰蛋白定量数据（癌症样本） |
| 泛素化修饰蛋白定量数据（癌症样本） |
| 乙酰化修饰蛋白定量数据（癌症样本） |
| SCLC（小细胞肺癌） | 磷酸化修饰蛋白定量数据 |
| 蛋白相互作用数据 | String数据库 |
| BioPlex |
| PathwayCommons11 |



1. LSCC数据处理及分析

2.1 数据预处理

从lscc\_merge\_information.csv文件中，根据三种P.value值（P\_all，P\_significance，P\_2-Fold）设置阈值（0.05），综合筛选出307个蛋白（P1\_Gene\_Name），涉及到4605个PTM修饰位点信息。

表1 Pvalue筛选蛋白数量表

|  |  |  |
| --- | --- | --- |
| ALL | Sig | Fold |
| 3916 | 1730 | 724 |

将4605个修饰位点映射到癌症细胞中的蛋白定量数据，由于存在多个PTM修饰共用一条定量数据的情况，最终得到了4449条数据。（注意：此处定量数据存在负值，是因为经log2转换，后续分析需将其还原，使用原始数据）

表2 不同PTM修饰的定量数据表

|  |  |  |
| --- | --- | --- |
| 磷酸化 | 泛素化 | 乙酰化 |
| 1696 | 1698 | 1055 |

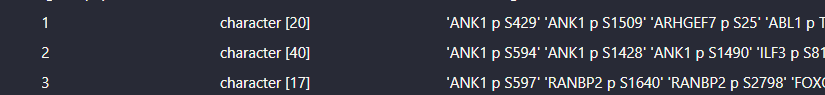
2.2降维并聚类

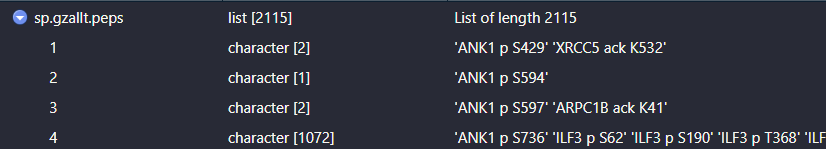
将映射得到的4449条数据，进行斯皮尔曼相关性计算，欧式距离计算以及SED计算，并使用Rtsne包对数据进行降维，并使用最小生成树方法进行聚类。

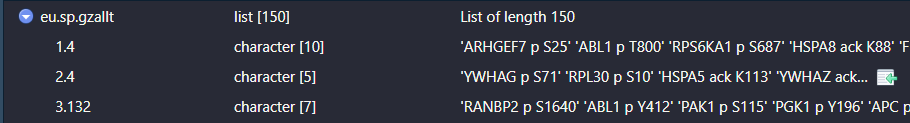
三种方法得到了三种聚类结果，我们需要得到共聚类的修饰位点信息。

首先将sp（斯皮尔曼）聚类结果与eu（欧式距离）聚类结果进行共聚类。

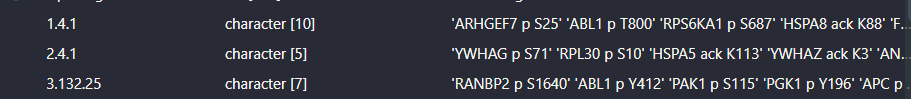
图A是eu聚类结果，图B是sp聚类结果。Eu的第一组聚类中共有20的元素（PTM修饰位点），sp的第四组聚类中共有1042个元素（PTM修饰位点），他们相同的PTM修饰位点信息共有10个，大于我们所设置的阈值（2），认为这10个PTM修饰位点是被sp和eu方法共聚类，存入一个新的list（eu.sp.gzallt）。

A 

B 

C 

将得到的新的list（eu.sp.gzallt）与sed聚类结果进行比对，得到一个新的共聚类list。



其中1.4.1代表着三种聚类方法中的组别，依次是eu（斯皮尔曼聚类结果）的第一组，sp（欧式聚类结果）的第四组，以及sed（聚类结果）的第一组。

共1579个PTM被共聚类成149组。

2.3 构建CCCN网络

在数据降维以及聚类中，我们得到了共聚类的蛋白质PTM位点信息，并以此为依据筛选出对应的PTM位点修饰蛋白的定量数据，构成一个新的矩阵数据。

此处还进行了进一步的筛选：我们计算出每一种PTM位点修饰蛋白的定量数据的有效值数量（样本数108减去NA值的数量），去除掉小于较小四分位数（后四分之一）的定量数据（阈值为56，即每行数据中至少有56个不是NA的有效值），构成一个新的数据矩阵。

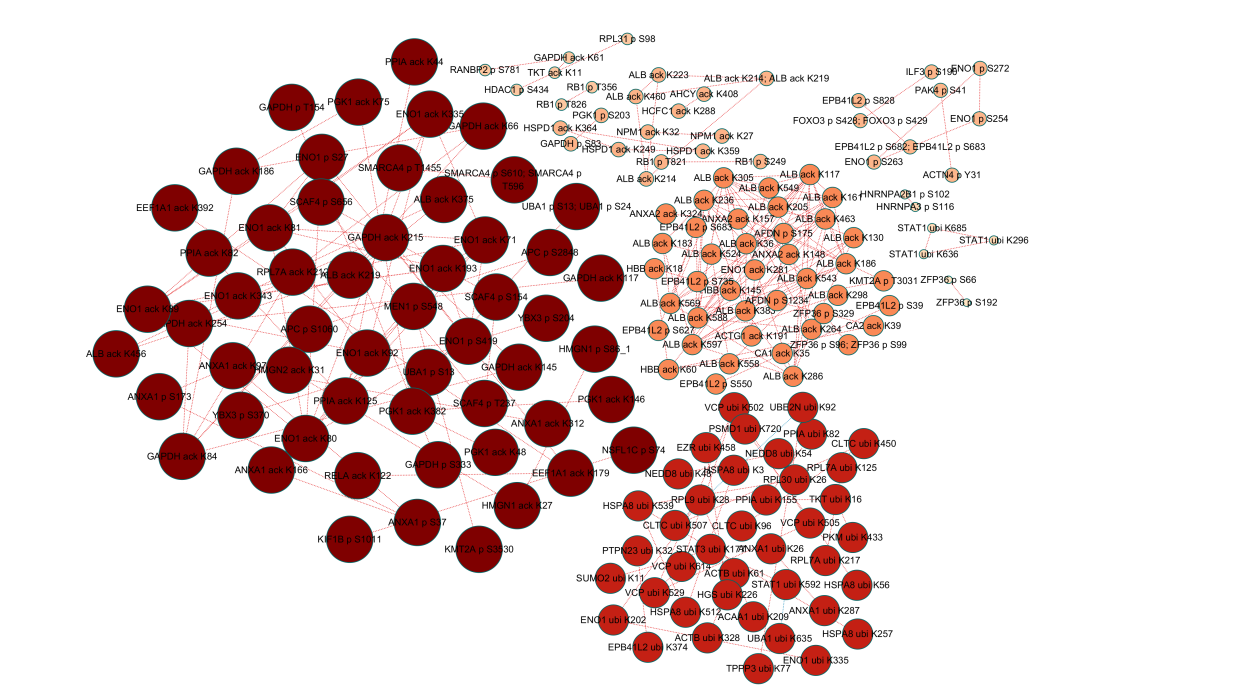
最终我们对1013个PTM位点构建CCCN，其中1013个PTM位点涉及227个蛋白质。

在第二步 数据降维以及聚类 中，我们使用了皮尔逊计算相关性系数。根据筛选出的PTM修饰位点信息，提取相关性系数，构成新的相关性系数矩阵（gzallt.cccn）。

2.3.1 PTM CCCN

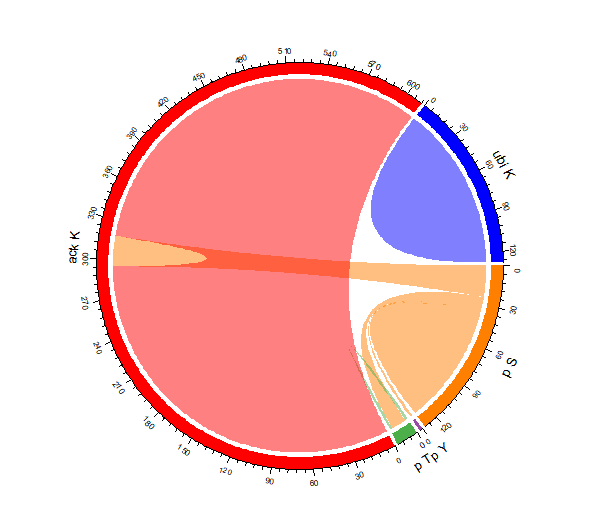
将该矩阵数据转换成三列，第一、第二列均为Name，第三列为Weigh（相关性系数），第四列根据Weigh大小，认为是Positive correlation（Weigh > 0.5），negative correlation（Weigh < -0.5）以及correlation。并以此提交到Cytoscape构建CCCN网络。（左图）

提取Positive correlation与negative correlation的位点互作信息，重新构建网络，发现呈现Positive correlation的位点信息较多。（右图）



共175个PTM修饰位点，222个相互作用，涉及61个蛋白质。Node颜色深浅以及大小显示相互作用数目的变化。

从表格及图中，我们可以看到，大部分的作用关系都是修饰类型内部的相互作用，而且修饰类型之间的相互作用，主要发生在丝氨酸磷酸化与其他修饰（苏氨酸磷酸化、赖氨酸乙酰化以及酪氨酸磷酸化）



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | p S | p Y | p T | ack K | ubi K |
| p S | 23 | 1 | 6 | 12 | 0 |
| p Y |  | 0 | 0 | 0 | 0 |
| p T |  |  | 1 | 0 | 0 |
| ack K |  |  |  | 147 | 0 |
| ubi K |  |  |  |  | 32 |

统计每种蛋白出现的次数，我们发现ALB、ENO1和GAPDH蛋白在CCCN网络中的占比较大。

|  |  |  |  |
| --- | --- | --- | --- |
| Gene | Count | Gene | Count |
| ALB | 178 | GAPDH | 27 |
| ENO1 | 68 | VCP | 10 |

ALB、ENO1和GAPDH是一些常见的蛋白质，它们在细胞中具有重要的功能和作用。

ALB是一种血浆蛋白质，在体内有多种重要功能。它起着维持血浆渗透压、维持正常的胶体渗透压、调节体内酸碱平衡以及输送和保护其他分子的作用。此外，ALB还参与免疫反应、抗炎过程和抗氧化反应等。

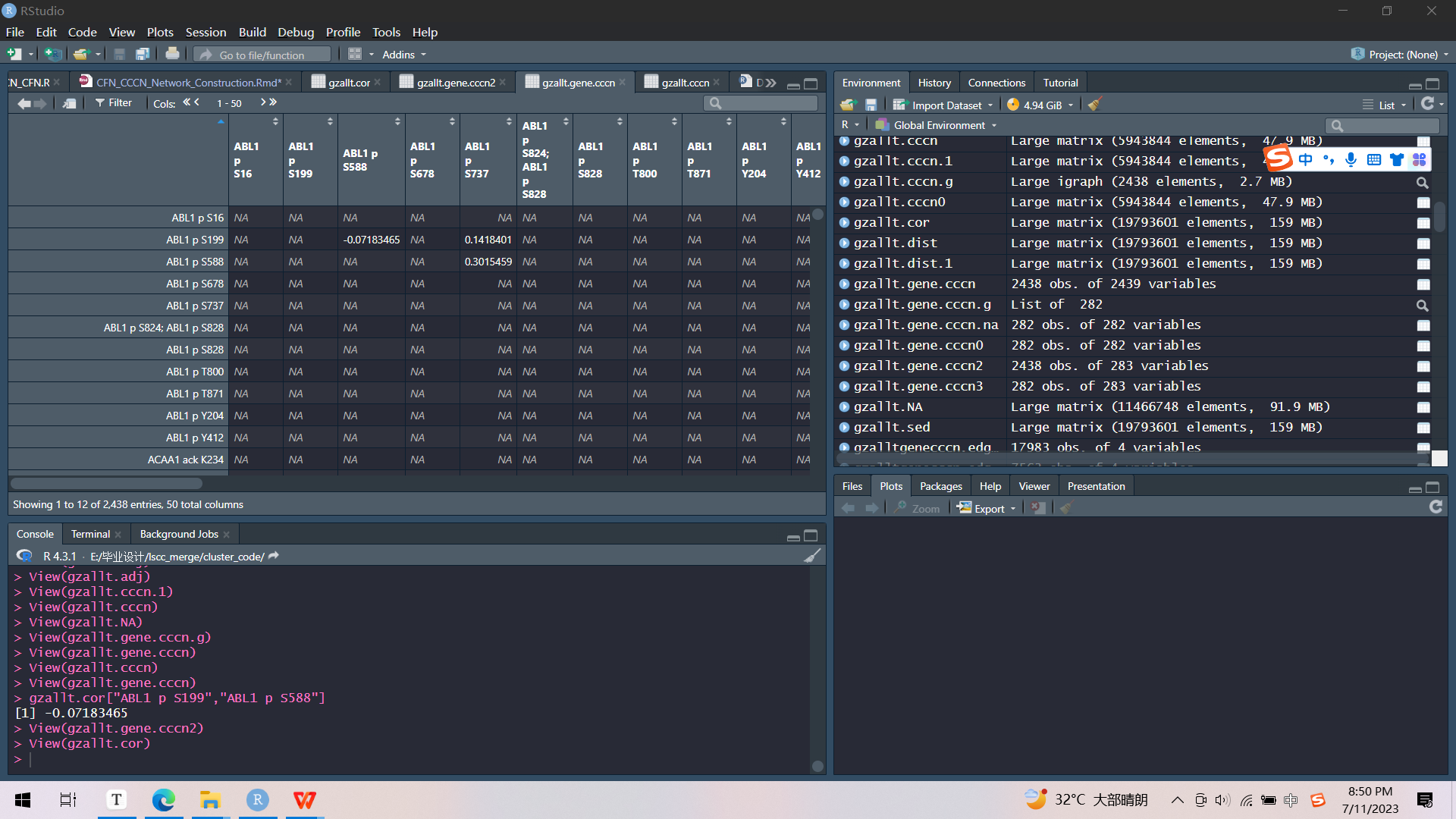
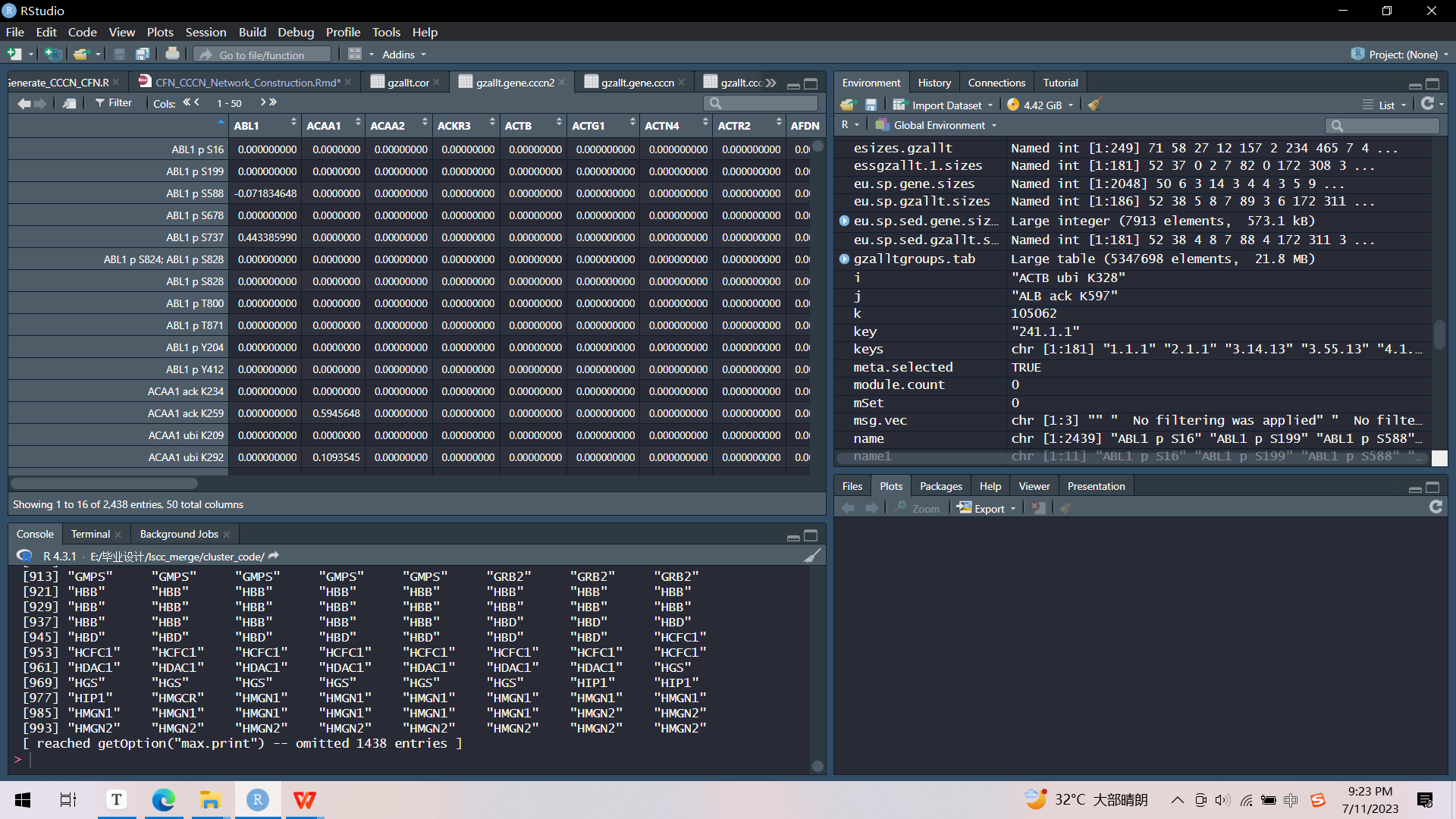
ENO1是一种糖酵解途径中的酶，参与葡萄糖代谢和能量产生。除了在糖酵解中起着催化反应的作用外，ENO1还与多种基因调控和细胞信号传导通路有关。此外，ENO1也被发现参与了细胞增殖、凋亡和肿瘤的发生与发展。

GAPDH是糖酵解途径中的一种酶。它在糖酵解过程中催化糖分子的分解以产生能量。除了在能量代谢中的作用外，GAPDH也在其他重要的细胞过程中发挥作用，如细胞增殖、RNA转录和调控、凋亡等。

关于这三种蛋白质与非小细胞肺癌的关系，研究表明它们在非小细胞肺癌的发展和进展中可能发挥着重要的角色。ALB的降低与非小细胞肺癌的恶化和预后不良有关。低血浆ALB水平与肿瘤的恶性程度、淋巴结转移、预后不良等因素相关。ENO1在非小细胞肺癌的发展中起着促进肿瘤细胞增殖、侵袭和转移的作用。高表达ENO1与非小细胞肺癌的恶性程度、预后不良等因素相关。GAPDH在非小细胞肺癌的发展中也发挥重要的作用。它参与调节细胞凋亡、增殖和转化等过程。

2.3.2 Protein蛋白质互作网络

根据筛选的得到的相关性系数矩阵（gzallt.cccn），我们先对列层面的相关性系数合并。

A  B 

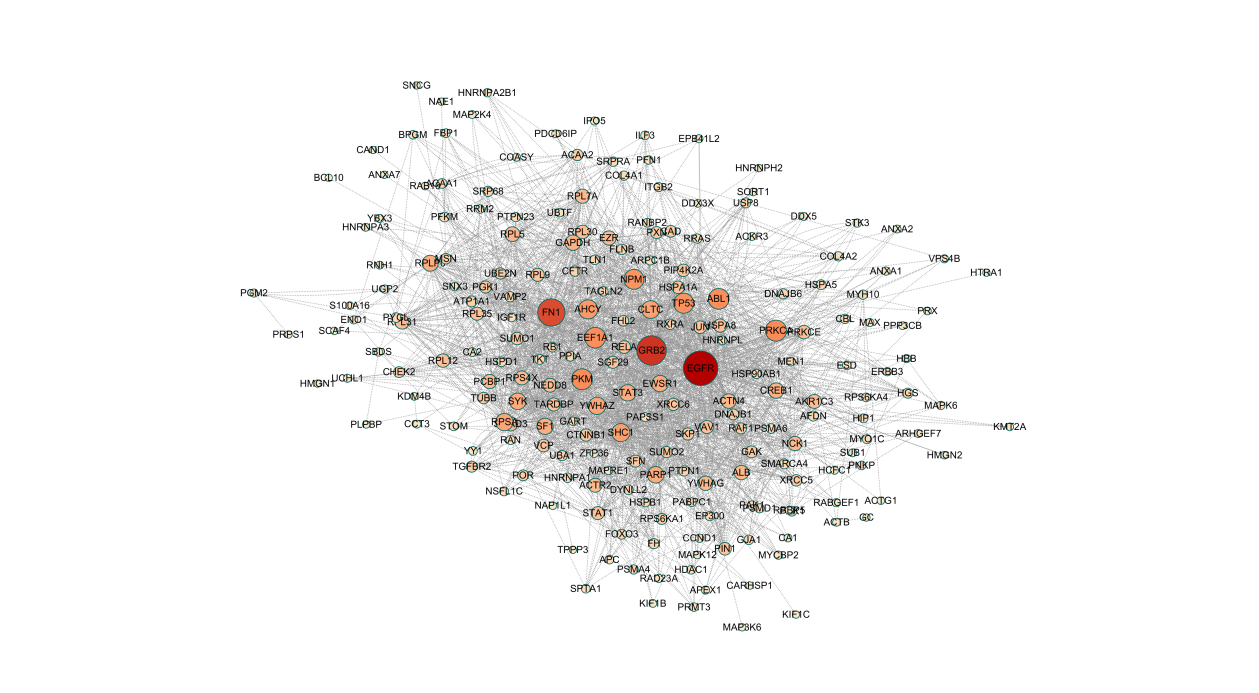
以第一列数据（ABL1 p S16）为例，该PTM修饰位点与很多PTM修饰位点都有相关性。我们对列名按照基因名前缀进行分组（如ABL1 p S16、ABL1 p S199、ABL1 p S588等都有前缀名ABL1，所以归为ABL1组）。将ABL1 p S16与ABL1下的所有PTM site的相关性加起来，作为ABL1 p S16与ABL1的相关性系数，存储到新的矩阵数据（图B）（数据结果经过转置操作）。

接下来就是行层面的相关性系数合并。一样的方法，得到新的矩阵数据，作为Protein CCCN的构建来源，后续也将用于CFN的构建。

2.4 构建CFN网络

根据Stringdb、GeneMANIA、PathwayCommons11、BioPlex、Iptmnet等数据库构建PPI信息，并以此为根据对CCCN数据进一步筛选，找出已被证实二者之间有相互作用的蛋白质互作信息，构建CFN网络。

目前只对Stringdb、BioPlex以及PathwayCommons11数据进行处理分析。



EGFR与肿瘤细胞的增殖、血管生成、肿瘤侵袭、转移及细胞凋亡的抑制有关。

GRB2在细胞表面生长因子受体和Ras信号通路之间提供关键联系的适配器蛋白

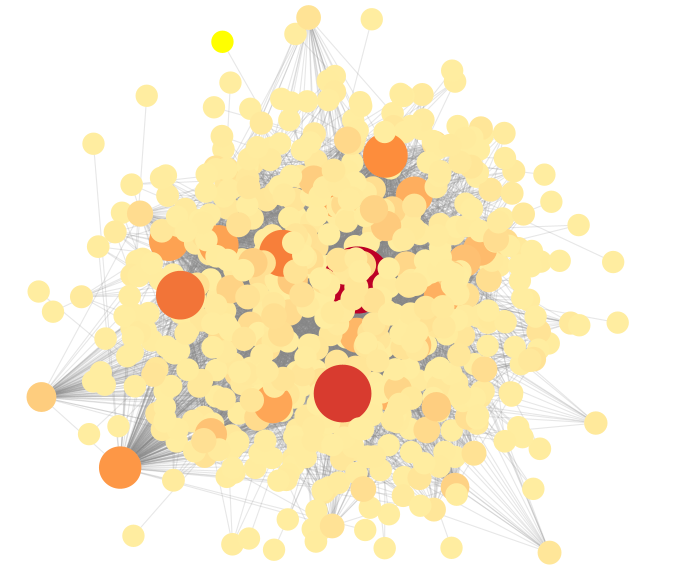
FN1在纤连蛋白结合细胞表面和各种化合物，包括胶原蛋白，纤维蛋白，肝素，DNA和肌动蛋白。纤连蛋白参与细胞粘附、细胞运动、调理、伤口愈合和维持细胞形状。

将蛋白互作数目大于20的蛋白进行基因富集分析（GO、KEGG），具体如下。

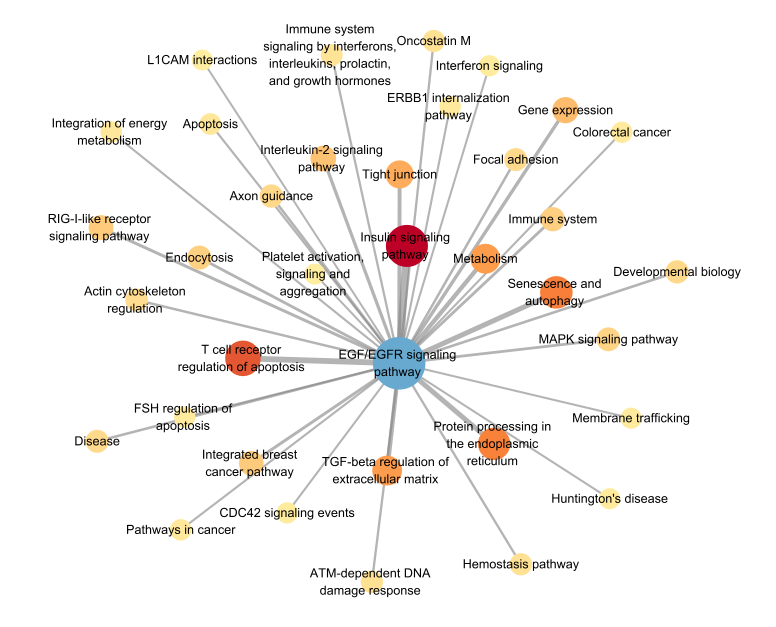
2.5构建PCN（Pathway crosstalk network）

根据NCATS BioPlanet数据库中1657条的通路，构建了一个通路串扰网络（PCN），其中通路-通路关系（网络的边）由PTM共簇的通路基因定义。通路之间相互作用的权重（PTM cluster weight）由每个通路的蛋白质的数量定义。

最终构建的PCN网络中有998个节点，371602条边。设置筛选阈值：PTM cluster weight > 0.065，得到相互作用较强的子网络（node: 649, edge: 5579）。



EGF/EGFR signaling pathway相关信号通路：

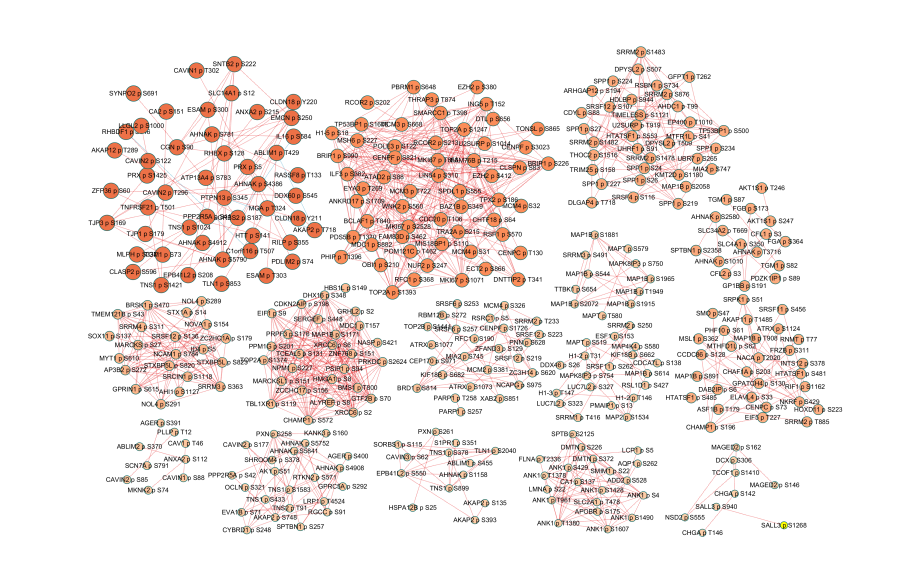
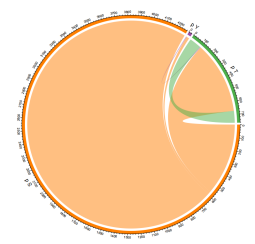


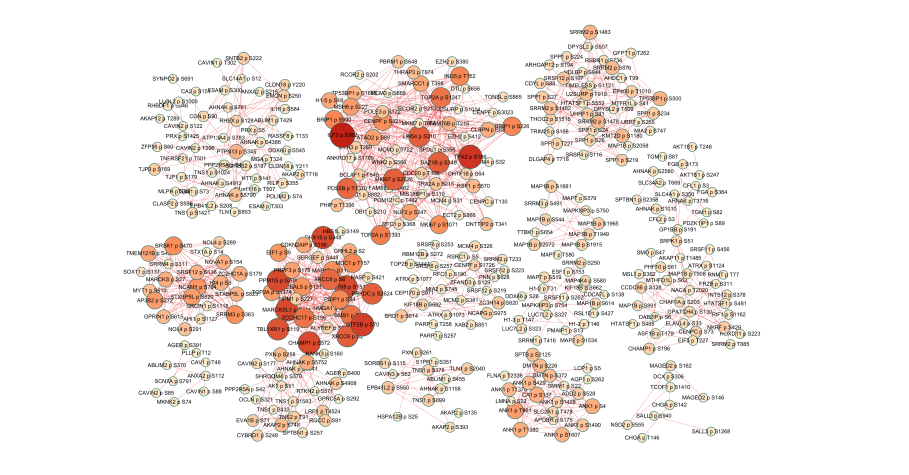
从PCN中提取出EGF/EGFR signaling pathway相关信号通路，共有953条通路与该通路有相互作用。筛选PTM cluster weight > 0.065的相互作用得到上图（节点的大小颜色代表变的权重）。其中，Insulin signaling pathway、T cell receptor regulation of apoptosis、Senescence and autophagy、Protein processing in the endoplasmic reticulum、Metabolism与EGF/EGFR signaling pathway相互作用较强。

1. sclc数据处理及分析

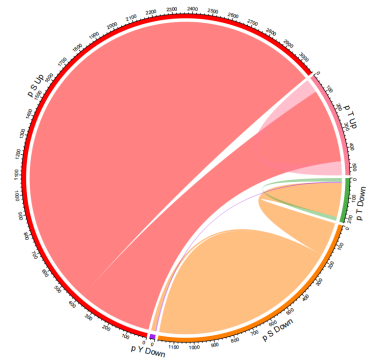
|  |  |  |  |
| --- | --- | --- | --- |
|  | p S | p Y | p T |
| p S | 918 | 11 | 285 |
| p Y |  | 0 | 3 |
| p T |  |  | 47 |

使用edgeR包对数据进行差异基因分析，筛选条件为（pvalue<0.05,logFC绝对值>2）。将筛选出来的差异基因的癌症样本的数据（1390条数据）另存为dataframe，作为后续的分析。此处数据log2转化之后，分析之前需将其还原。





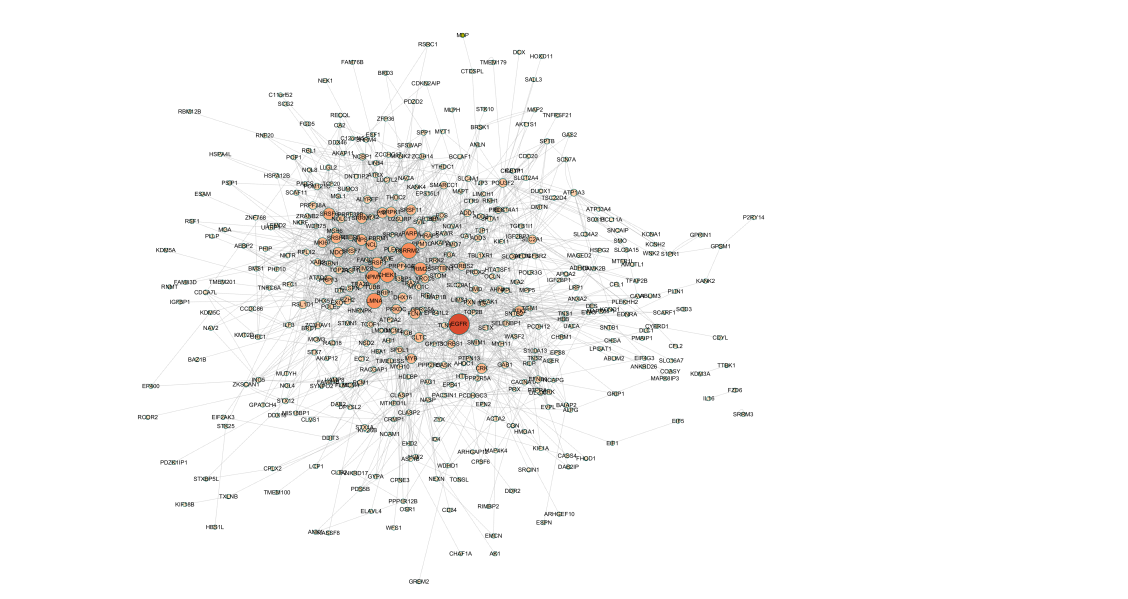
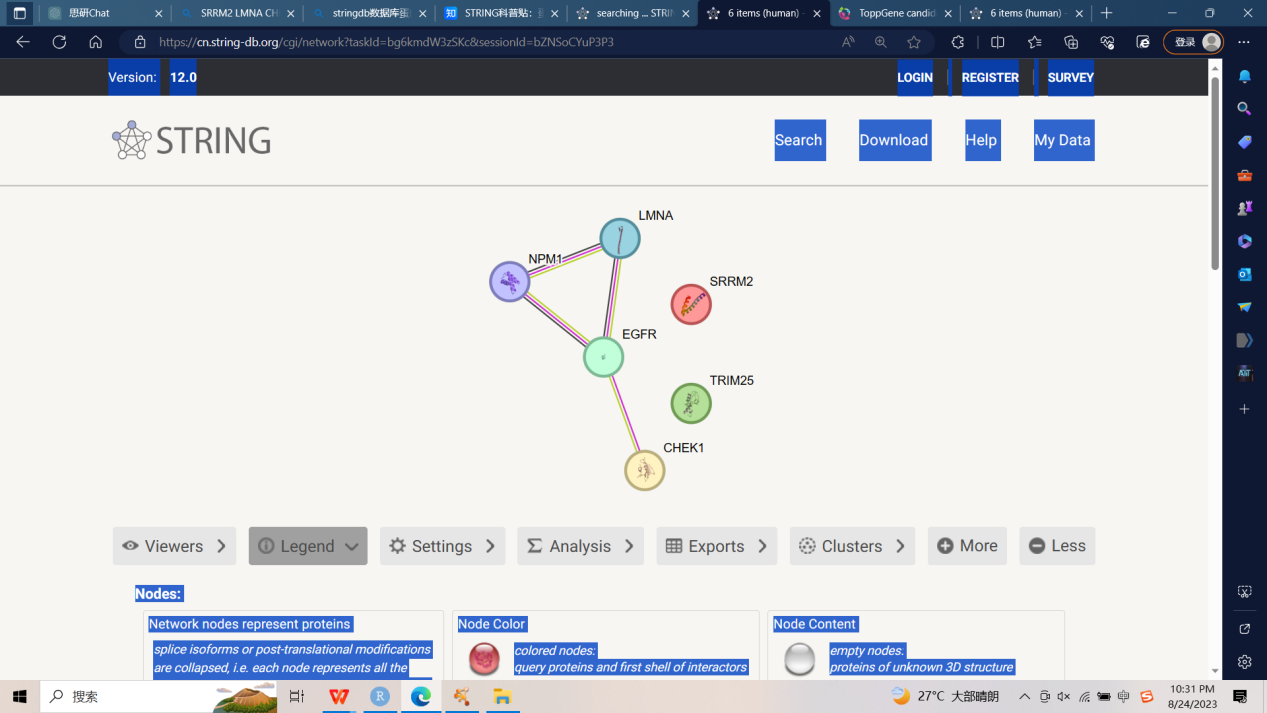
共563个PTM修饰位点，共1264个相互作用，涉及330个蛋白质。从表格及图中，我们可以看到，大部分的作用关系都是丝氨酸磷酸化之间，以及丝氨酸磷酸化与苏氨酸磷酸化的相互作用。



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | p S Down | p Y Down | p T Down | p S Up | p Y Up | p T Up |
| p S Down | 245 | 11 | 90 |  |  |  |
| p Y Down |  | 0 | 3 |  |  |  |
| p T Down |  |  | 10 |  |  |  |
| p S Up |  |  |  | 673 | 0 | 195 |
| p Y Up |  |  |  |  | 0 | 0 |
| p T Up |  |  |  |  |  | 37 |

统计每种蛋白出现的次数，我们发现ANK1、TOP2A、AHNAK等蛋白在CCCN网络中的占比较大。

|  |  |  |  |
| --- | --- | --- | --- |
| Gene | Count | Gene | Count |
| ANK1 | 95 | XRCC6 | 41 |
| TOP2A | 94 | MKI67 | 40 |
| AHNAK | 83 | SPP1 | 38 |
| SRRM2 | 66 | BRIP1 | 34 |
| MAP18 | 64 | SRRM3 | 30 |

|  |  |
| --- | --- |
| Gene | SCLC.count |
| EGFR | 103 |
| SRRM2 | 71 |
| LMNA | 69 |
| CHEK1 | 64 |
| NPM1 | 61 |
| TRIM25 | 56 |

共涉及到209个蛋白

1. lscc与sclc分析

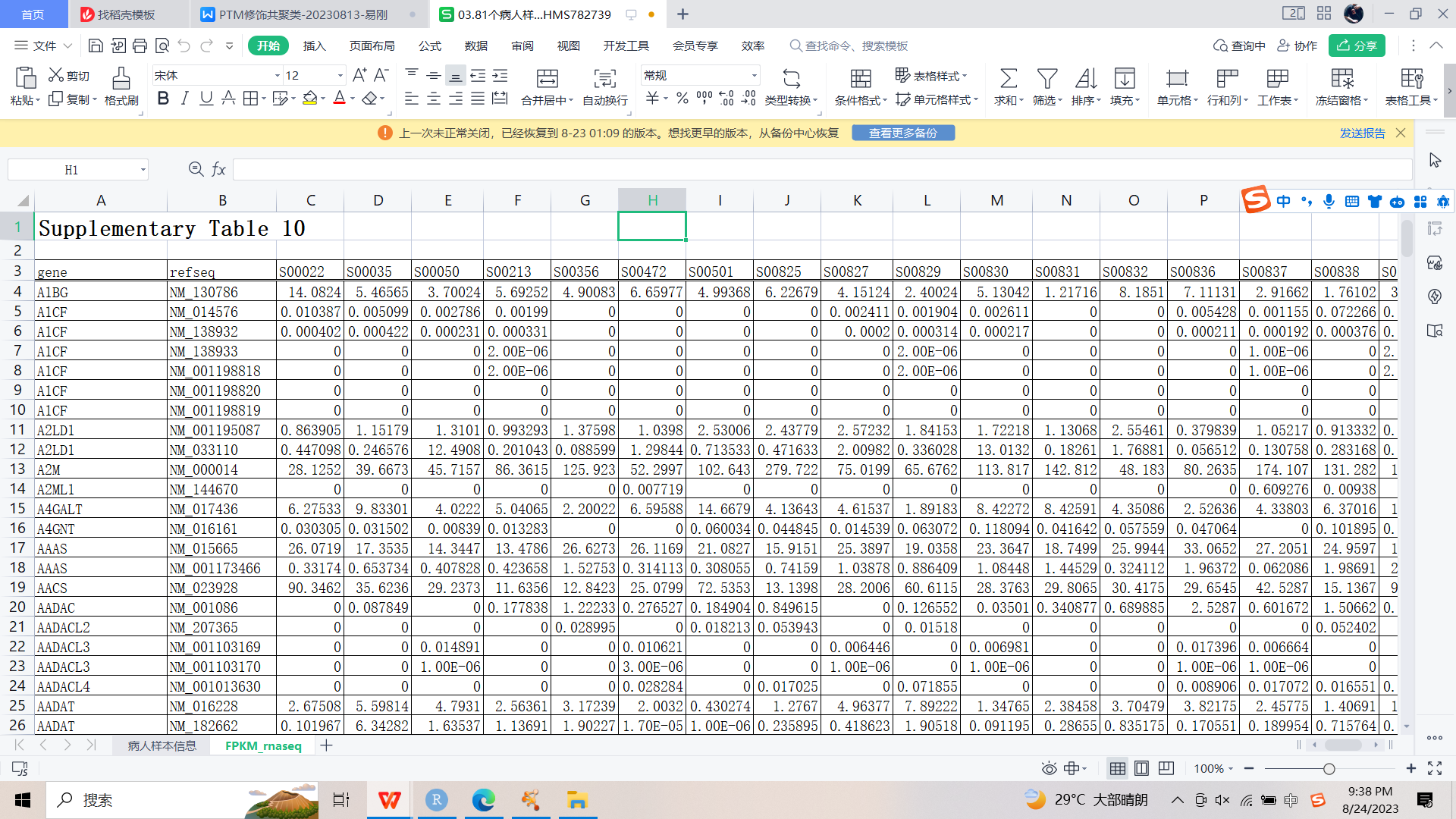
共同基因

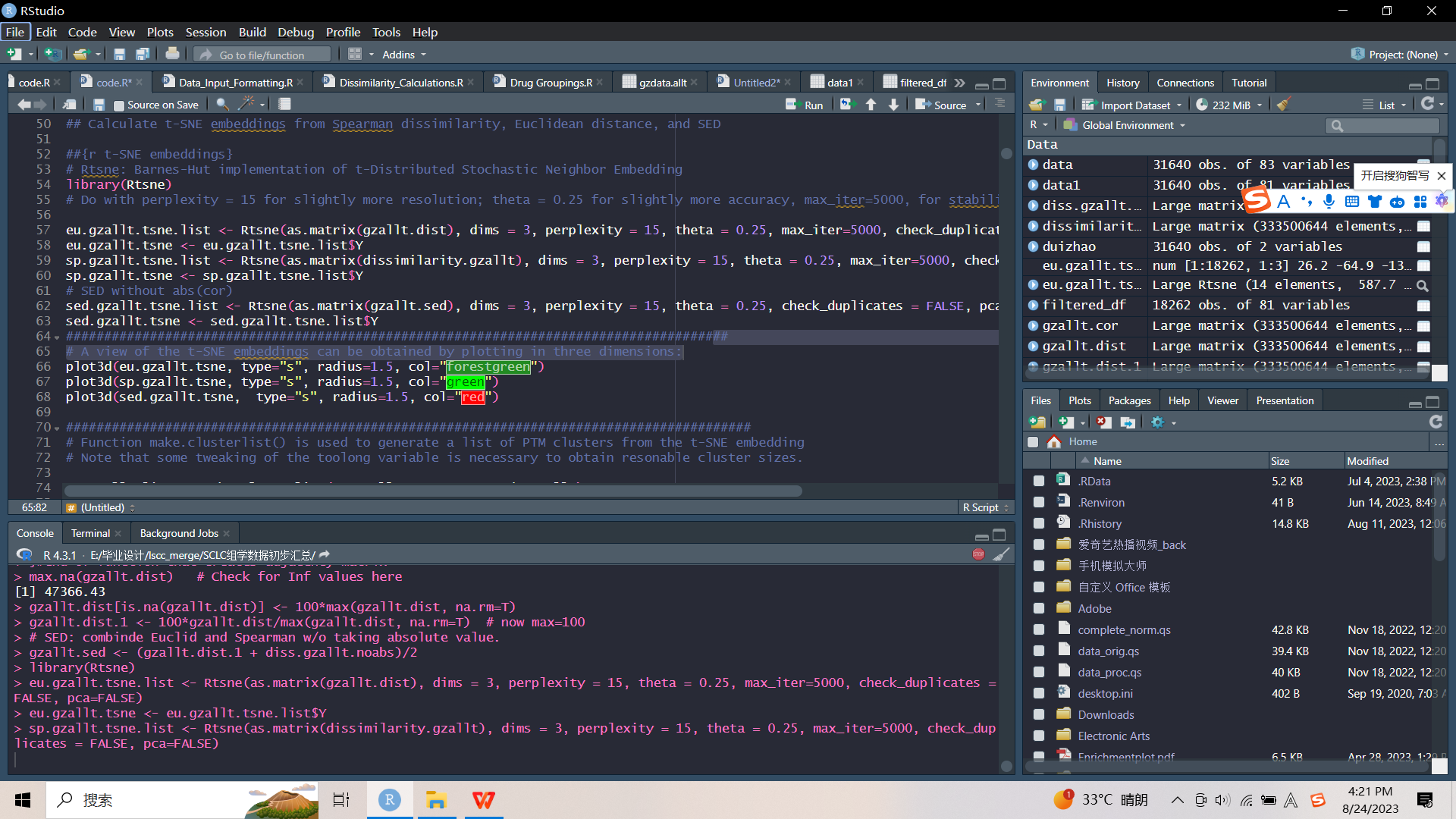
|  |  |  |  |
| --- | --- | --- | --- |
| Gene | SCLC.count | LSCC.count | Function |
| EGFR | 103 | 140 | 参与细胞增殖、分化和生存的信号传导过程 |
| NPM1 | 61 | 65 | 参与细胞核内的多种生物过程，包括RNA处理、核仁组装和细胞周期调控 |
| PARP1 | 54 | 48 | 负责DNA损伤修复和基因表达调控 |
| CLTC | 37 | 54 | 参与内质网与高尔基体之间的物质转运 |
| XRCC6 | 29 | 26 | 与DNA双链断裂的修复和维持基因组稳定性密切相关 |
| PXN | 20 | 25 | 参与细胞迁移、细胞黏附、细胞增殖和细胞凋亡等。 |

不同基因

|  |  |  |  |
| --- | --- | --- | --- |
| SCLC.gene | Count | LSCC.gene | Count |
| SRRM2 | 71 | GRB2 | 114 |
| LMNA | 69 | FN1 | 103 |
| CHEK1 | 64 | PRKCA | 71 |
| TRIM25 | 56 | PKM | 70 |
| SRSF6 | 48 | ABL1 | 69 |
| NCL | 43 | EEF1A1 | 69 |
| RNPS1 | 43 | TP53 | 66 |
| SRSF3 | 43 | AHCY | 61 |
| SRSF4 | 42 | SHC1 | 56 |
| FLNA | 41 | RPSA | 52 |
| SRPK1 | 40 | YWHAZ | 52 |

转录组学数据研究





附录

library(rgl)

##read data

setwd("E:/毕业设计/lscc\_merge/cluster\_code")

data <- read.csv("union\_site\_protein.csv")

result <- data

result <- as.data.frame(result)

rownames(result) <- result[,"X"]

result <- result[,-1]

# Getting the raw data

result1 <- 2^result

gzdata.allt <- result1

# correlation calculation

gzallt.cor <- cor(t(gzdata.allt), use = "pairwise.complete.obs", method = "spearman")

# Pearson Works, though there were many zero std. deviations

diag(gzallt.cor) <- NA

dissimilarity.gzallt <- 1 - abs(gzallt.cor)

diss.gzallt.noabs <- 1 - gzallt.cor

# set NA to two orders of magnitude higher than max distance

dissimilarity.gzallt[is.na(dissimilarity.gzallt)] <- 100\*max(dissimilarity.gzallt, na.rm=T)

diss.gzallt.noabs[is.na(diss.gzallt.noabs)] <- 50\*max(diss.gzallt.noabs, na.rm=T)

# Euclid

gzallt.dist = as.matrix (dist (gzdata.allt), method = "euclidean")

# check

max.na(gzallt.dist) # Check for Inf values here

gzallt.dist[is.na(gzallt.dist)] <- 100\*max(gzallt.dist, na.rm=T)

gzallt.dist.1 <- 100\*gzallt.dist/max(gzallt.dist, na.rm=T) # now max=100

# SED: combinde Euclid and Spearman w/o taking absolute value.

gzallt.sed <- (gzallt.dist.1 + diss.gzallt.noabs)/2

#

## Calculate t-SNE embeddings from Spearman dissimilarity, Euclidean distance, and SED

##{r t-SNE embeddings}

# Rtsne: Barnes-Hut implementation of t-Distributed Stochastic Neighbor Embedding

library(Rtsne)

# Do with perplexity = 15 for slightly more resolution; theta = 0.25 for slightly more accuracy, max\_iter=5000, for stabiliztion of groups

eu.gzallt.tsne.list <- Rtsne(as.matrix(gzallt.dist), dims = 3, perplexity = 15, theta = 0.25, max\_iter=5000, check\_duplicates = FALSE, pca=FALSE)

eu.gzallt.tsne <- eu.gzallt.tsne.list$Y

sp.gzallt.tsne.list <- Rtsne(as.matrix(dissimilarity.gzallt), dims = 3, perplexity = 15, theta = 0.25, max\_iter=5000, check\_duplicates = FALSE, pca=FALSE)

sp.gzallt.tsne <- sp.gzallt.tsne.list$Y

# SED without abs(cor)

sed.gzallt.tsne.list <- Rtsne(as.matrix(gzallt.sed), dims = 3, perplexity = 15, theta = 0.25, check\_duplicates = FALSE, pca=FALSE)

sed.gzallt.tsne <- sed.gzallt.tsne.list$Y

#######################################################################################

# A view of the t-SNE embeddings can be obtained by plotting in three dimensions:

plot3d(eu.gzallt.tsne, type="s", radius=1.5, col="forestgreen")

plot3d(sp.gzallt.tsne, type="s", radius=1.5, col="green")

plot3d(sed.gzallt.tsne, type="s", radius=1.5, col="red")

##########################################################################################

# Function make.clusterlist() is used to generate a list of PTM clusters from the t-SNE embedding

# Note that some tweaking of the toolong variable is necessary to obtain resonable cluster sizes.

eu.gzallt.list <- make.clusterlist(eu.gzallt.tsne, 3.8, gzdata.allt)

esizes.gzallt <- sapply(eu.gzallt.list, function(x) dim(x)[1])

sp.gzallt.list <- make.clusterlist(sp.gzallt.tsne, 3.8, gzdata.allt)

spsizes.gzallt <- sapply(sp.gzallt.list, function(x) dim(x)[1])

sed.gzallt.list <- make.clusterlist(sed.gzallt.tsne, 3.0, gzdata.allt)

sedsizes.gzallt <- sapply(sed.gzallt.list, function(x) dim(x)[1])

# note several large clusters that should be broken up

hist(esizes.gzallt, breaks=100, col="red")

hist(spsizes.gzallt, breaks=100, col="blue")

hist(sedsizes.gzallt, breaks=100, col="green")

# Focus on intersect of all clusters from Euclid, Spearman, and SED using list.common()

# Concatenate group list files

# make group names unique

eu.gzallt.df <- ldply(eu.gzallt.list)[,2:3]

sp.gzallt.df <- ldply(sp.gzallt.list)[,2:3]

sed.gzallt.df <- ldply(sed.gzallt.list)[,2:3] # Further partition large groups

eu.gzallt.df $group <- paste(noquote(eu.gzallt.df $group), noquote("e"), sep="", collapse=NULL)

sp.gzallt.df $group <- paste(noquote(sp.gzallt.df $group), noquote("s"), sep="", collapse=NULL)

sed.gzallt.df $group <- paste(noquote(sed.gzallt.df $group), noquote("sed"), sep="", collapse=NULL)

gzalltgroups.df <- rbind(eu.gzallt.df, sed.gzallt.df, sp.gzallt.df)

# For checking:

gzalltgroups.tab <- table(gzalltgroups.df)

# Get the gene nemase from PTM clusters

eu.gzallt.genes <- lapply(eu.gzallt.list, extract.genes.from.clist)

sp.gzallt.genes <- lapply(sp.gzallt.list, extract.genes.from.clist)

sed.gzallt.genes <- lapply(sed.gzallt.list, extract.genes.from.clist)

# Extract the PTM names

eu.gzallt.peps <- lapply(eu.gzallt.list, extract.peps.from.clist)

sp.gzallt.peps <- lapply(sp.gzallt.list, extract.peps.from.clist)

sed.gzallt.peps <- lapply(sed.gzallt.list, extract.peps.from.clist)

eu.sp.gene <- list.common(eu.gzallt.genes, sp.gzallt.genes, keeplength=2)

eu.sp.gene.sizes <- sapply(eu.sp.gene, length)

eu.sp.sed.gene <- list.common(eu.sp.gene, sed.gzallt.genes, keeplength=2)

eu.sp.sed.gene.sizes <- sapply(eu.sp.sed.gene, length)

eu.sp.gzallt <- list.common(eu.gzallt.peps, sp.gzallt.peps, keeplength=2)

eu.sp.gzallt.sizes <- sapply(eu.sp.gzallt, length)

eu.sp.sed.gzallt <- list.common(eu.sp.gzallt, sed.gzallt.peps, keeplength=2)

eu.sp.sed.gzallt.sizes <- sapply(eu.sp.sed.gzallt, length)

hist(eu.sp.sed.gzallt.sizes, breaks=100, col="gold")

# max = 111; length = 612; round 2 max 88, length 878; 111 and 839 round 3

#

# Examine all clusters

eu.sp.sed.gzallt.data <- list()

for (i in 1:length(eu.sp.sed.gzallt)) {

if (length(intersect(eu.sp.sed.gzallt[[i]], rownames(gzdata.allt)))==0) next

at <- gzdata.allt[unlist(eu.sp.sed.gzallt[[i]]),]

if(dim(at)[1]<2 | dim(at)[2]<2) next

eu.sp.sed.gzallt.data[[i]] <- clust.data.from.vec(eu.sp.sed.gzallt[[i]], tbl=gzdata.allt)

print(i)

}

# Check:

gzdata.allt[rownames(gzdata.allt) %in% eu.sp.sed.gzallt[[i]],]

# Some have data in only one column - -

# Note that first data set (gzdata above) was not trimmed for PTMs that were detected in more than two experiments

alltsamples <- apply(gzdata.allt, 1, filled)

#hist(alltsamples, breaks=50, col="magenta")

gzdata.allt.t <- gzdata.allt[which(alltsamples>56),]

dim(gzdata.allt.t) ## 3308

gzdata.allt1 <- gzdata.allt.t

# Repair here because t-SNE steps took a very long time

# Print out and delete bad ones

bad.clusterlist <- list()

for (i in 1:length(eu.sp.sed.gzallt)) {

if (length(intersect(eu.sp.sed.gzallt[[i]], rownames(gzdata.allt)))==0) {

print(i)

bad.clusterlist[[i]] }

}

# Find any that may be included in clusters

badptms <- unique(outersect(rownames(gzdata.allt), rownames(gzdata.allt1))) # 1141

testnames <- unique(unlist(eu.sp.sed.gzallt))

length(intersect (badptms, testnames)) # 569

essgzallt.1 <- lapply(eu.sp.sed.gzallt, function(x) x %w/o% badptms)

essgzallt.1.sizes <- sapply(essgzallt.1, length)

removed <- eu.sp.sed.gzallt.sizes - essgzallt.1.sizes

removed <- removed[removed>0]

# note: some entire clusters removed

essgzallt.2 <- essgzallt.1[essgzallt.1.sizes>0] # now 83 clusters from 839

essgzallt <- essgzallt.2

# Create data list for evaluation of clusters

essgzallt.data <- lapply(essgzallt, clust.data.from.vec, tbl= gzdata.allt1)

length(eu.sp.sed.gzallt)

# 838

length(essgzallt)

# Pruned length: 818

# This list was prepared with all data including ratios

# Adjacency matrix:

require(plyr)

library(plyr)

gzallt.adj <- rbind.fill.matrix(llply(essgzallt, make.adj.mat))

rownames(gzallt.adj) <- colnames(gzallt.adj)

# dim 1013 1013

gzallt.adj.o <- gzallt.adj[order(rownames(gzallt.adj)), order(colnames(gzallt.adj))]

gzallt.cccn.1 <- gzallt.cor[rownames(gzallt.cor) %in% rownames(gzallt.adj.o), colnames(gzallt.cor) %in% colnames(gzallt.adj.o)]

# Check:

setdiff(rownames(gzallt.adj), rownames(gzallt.cccn.1)) # 0

length(intersect(rownames(gzallt.adj), rownames(gzallt.cor))) # 1013 all good

# Add in correlation as edge values in adjacency matrix:

gzallt.cccn <- gzallt.cor[intersect(rownames(gzallt.adj.o), rownames(gzallt.cor)), intersect(colnames(gzallt.adj.o), colnames(gzallt.cor))]

identical(rownames(gzallt.cccn), colnames(gzallt.cccn))

# [1] TRUE

dim(gzallt.cccn)

# 7398 7398

gzallt.NA <- which(is.na(gzallt.adj.o), arr.ind = TRUE)

gzallt.cccn <- replace (gzallt.cccn, gzallt.NA, NA)

# remove self loops

if(any(!is.na(diag(gzallt.cccn)))) {diag(gzallt.cccn) <- NA}

## \*\*\*\*

# Option: Limit to a particlar correlation value like this.

# gzallt.cccn.halflim <- replace (gzallt.cccn, abs(gzallt.cccn)<0.5, NA)

#

# Make igraph object, which cannot have NA values, so replace NA with 0

gzallt.cccn0 <- gzallt.cccn

gzallt.cccn0[is.na(gzallt.cccn0)] <- 0

library(igraph)

gzallt.cccn.g <- graph.adjacency(as.matrix(gzallt.cccn0), mode="lower", diag=FALSE, weighted="Weight")

any(duplicated(rownames(gzallt.cccn))) # FALSE

any(duplicated(colnames(gzallt.cccn))) # FALSE

# Make Gene CCCN

# Note: Did this on server for large data set, much faster at ddply step

gzallt.gene.cccn <- data.frame(gzallt.cccn, row.names = rownames(gzallt.cccn), check.rows=TRUE, check.names=FALSE, fix.empty.names = FALSE)

identical(rownames(gzallt.gene.cccn), colnames(gzallt.gene.cccn)) # TRUE

# Extract the gene names from the PTM names

gzallt.gene.cccn$Gene.Name <- sapply(rownames(gzallt.gene.cccn), function (x) unlist(strsplit(x, " ", fixed=TRUE))[1])

length(unique(gzallt.gene.cccn$Gene.Name)) # 2913

any(is.na(gzallt.gene.cccn$Gene.Name)) # F

# gzallt.gene.cccn$Gene.Name[grep("NA", gzallt.gene.cccn$Gene.Name, fixed=TRUE)]

# All genes with NA in the name, e.g. "CTNNA1"

# Use only upper triangle so correlations are not duplicated during the next step

gzallt.gene.cccn[lower.tri(gzallt.gene.cccn)] <- NA

library(reshape2)

result <- melt(as.matrix(gzallt.gene.cccn[,1:1013]))

colnames(result) <- c("Site1", "Site2", "weigh")

result.1 <- result %>% drop\_na(weigh)

result.1$interaction <- "correlation"

result.1$interaction[result.1$weigh<=-0.5] <- "negative correlation"

result.1$interaction[result.1$weigh>=0.5] <- "positive correlation"

write.table(result.1,"site\_corr.txt",quote = FALSE,row.names = FALSE,sep = "\t")

result.2 <- result.1[result.1$interaction %in% c("negative correlation","positive correlation"),]

write.table(result.2,"select\_site\_corr.txt",quote = FALSE,row.names = FALSE,sep = "\t")

box\_data <- read.table("select\_site\_corr.txt",header=TRUE, sep = "\t", comment.char = "", na.strings='', stringsAsFactors=FALSE, fill=TRUE)

box\_data <- box\_data[-203, ]

box\_data1 <- box\_data[,c(2,1,3,4)]

box\_data2 <- rbind(box\_data,box\_data1)

PTMSITE <- box\_data2$Site1

PTMSITE1 <- sub("(.\*?)\\s.\*", "\\1", PTMSITE)

PTMSITE1 <- table(PTMSITE1)

PTMSITE1 <- as.data.frame(PTMSITE1)

PTMSITE1 <- PTMSITE1[order(PTMSITE1$Freq,decreasing = TRUE),]

zhonglei <- c("p S","p Y","p T","ack K","ubi K")

df = as.data.frame(matrix(nrow=0,ncol=3))

for (i in zhonglei) {

for (j in zhonglei) {

select\_data <- box\_data2[grepl(i, box\_data2$Site1) & grepl(j, box\_data2$Site2), ]

if(i==j){

b <- dim(select\_data)[1]

row <- c(i, j, b)

df <- rbind(df, row)

}else{

select\_data1 <- box\_data2[grepl(i, box\_data2$Site2) & grepl(j, box\_data2$Site1), ]

b <- dim(select\_data)[1] + dim(select\_data1)[1]

row <- c(i, j, b)

df <- rbind(df, row)

}

}

}

df1 = as.data.frame(matrix(nrow=0,ncol=3))

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

for(j in i:length(zhonglei)){

name1 <- df[df$X.p.S.==zhonglei[i] & df$X.p.S..1== zhonglei[j],]

df1 <- rbind(df1, name1)

}

}

df1$X.46. <- as.numeric(df1$X.46.)

colnames(df1) <- c("from","to","value")

library(circlize)

pdf("PTM-COR-CIR.pdf")

grid.col <- c(

"p S" = "#ff7f00", "p Y"= "#984ea3", "p T" = "#4daf4a","ack K" = "red","ubi K" = "blue"

)

chordDiagram(df1, grid.col = grid.col)

dev.off()

png("PTM-COR-CIR.png")

grid.col <- c(

"p S" = "#ff7f00", "p Y"= "#984ea3", "p T" = "#4daf4a","ack K" = "red","ubi K" = "blue"

)

chordDiagram(df1, grid.col = grid.col)

dev.off()

boxdata <- as.data.frame(table(box\_data$Site1))

boxdata1 <- as.data.frame(table(box\_data$Site2))

colnames(boxdata1) <- c("Var1","Freq1")

boxdata2 <- merge(boxdata,boxdata1,all=TRUE)

boxdata2 [is.na(boxdata2 )]=0

boxdata2$Count <- boxdata2$Freq + boxdata2$Freq1

boxdata2$new\_column <- ifelse(grepl("p S", boxdata2$Var1), "p\_S",

ifelse(grepl("p T", boxdata2$Var1), "p\_T",

ifelse(grepl("p Y", boxdata2$Var1), "p\_Y",

ifelse(grepl("ack K", boxdata2$Var1), "ack\_K",

ifelse(grepl("ubi K", boxdata2$Var1), "ubi\_K", "Other")))))

boxdata2 <- boxdata2[boxdata2$Count>0,]

library(ggplot2)

ggplot(boxdata2, aes(x = new\_column, y = Count, fill = new\_column)) +

geom\_bar(stat = "identity") +

labs(x = "Category", y = "Count") +

theme\_minimal()

p <- ggplot(boxdata2, aes(new\_column, Count))

p + geom\_boxplot()

p + geom\_boxplot(outlier.shape = NA) + geom\_jitter(width = 0.2)

df <- data.frame(key = character(), value = character(), stringsAsFactors = FALSE)

for (key in names(eu.sp.sed.gzallt)) {

values <- eu.sp.sed.gzallt[[key]]

numRows <- length(values)

newRows <- data.frame(key = rep(key, numRows), value = values, stringsAsFactors = FALSE)

df <- rbind(df, newRows)

}

write.table(df,"cluster\_result.txt",quote = FALSE,row.names = FALSE,sep = "\t")

df1 <- data.frame(key = character(), value = character(), stringsAsFactors = FALSE)

for (key in names(eu.sp.sed.gene)) {

values <- eu.sp.sed.gene[[key]]

numRows <- length(values)

newRows <- data.frame(key = rep(key, numRows), value = values, stringsAsFactors = FALSE)

df1 <- rbind(df1, newRows)

}

write.table(df1,"cluster\_gene\_result.txt",quote = FALSE,row.names = FALSE,sep = "\t")

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Sum correlations in one dimension, then the other dimension

gzallt.gene.cccn2 <- ddply(gzallt.gene.cccn, .(Gene.Name), numcolwise(function(x) sum(x, na.rm=T)), .progress = "tk")

dim(gzallt.gene.cccn2) # 2913 7399

any(is.na(gzallt.gene.cccn2$Gene.Name)) # F

# make row names the Gene.Name

rownames(gzallt.gene.cccn2) <- gzallt.gene.cccn2$Gene.Name

# remove Gene.Name column

gzallt.gene.cccn2 <- gzallt.gene.cccn2[, 2:ncol(gzallt.gene.cccn2)]

# Now transform to sum in the other dimention

gzallt.gene.cccn2 <- data.frame(t(gzallt.gene.cccn2))

gzallt.gene.cccn2$Gene <- sapply(rownames(gzallt.gene.cccn2), function (x) unlist(strsplit(x, " ", fixed=TRUE))[1])

any(is.na(gzallt.gene.cccn2$Gene)) # F

# Sum other dimension

gzallt.gene.cccn3 <- ddply(gzallt.gene.cccn2, .(Gene), numcolwise(function(x) sum(x, na.rm=T)), .progress = "tk")

dim(gzallt.gene.cccn3) # 2913 2914

any(is.na(gzallt.gene.cccn3$Gene)) # F

# Check row and column names alignment

print(gzallt.gene.cccn3[1:20, 1:8])

tail(data.frame(gzallt.gene.cccn3$Gene, names(gzallt.gene.cccn3[2:ncol(gzallt.gene.cccn3)])))

# R likes to put dots in column names, which is a problem for ambiguous gene names and gene names with hyphens

# E.g., Note punctuation missing for "HLA" "NKX2" "NME1 in column names

gzallt.gene.cccn3$Gene[grep("HLA", gzallt.gene.cccn3$Gene)]

names(gzallt.gene.cccn3)[grep("HLA", names(gzallt.gene.cccn3))]

#

# Work around this problem (once satisfied that the gene names match).

names(gzallt.gene.cccn3)[2:ncol(gzallt.gene.cccn3)] <- gzallt.gene.cccn3$Gene

rownames(gzallt.gene.cccn3) <- gzallt.gene.cccn3$Gene

identical (rownames(gzallt.gene.cccn3), names(gzallt.gene.cccn3[,2:ncol(gzallt.gene.cccn3)])) # TRUE

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Make a adjacency matrix with NAs to index edges only for genes that co-cluster

gzallt.gene.cccn0 <- gzallt.gene.cccn3[,2:ncol(gzallt.gene.cccn3)]

# Replace 0 with NA

gzallt.gene.cccn.na <- zero.to.NA(gzallt.gene.cccn0)

# Check

gzallt.gene.cccn.na[1:20, 1:8]

hist(unlist(gzallt.gene.cccn.na), breaks=10, col="red", echo=FALSE)

hist(unlist(gzallt.gene.cccn0), breaks=1000, col="blue", ylim=c(0, 1200), echo=FALSE)

# Make igraph object

gzallt.gene.cccn.g <- graph.adjacency(as.matrix(gzallt.gene.cccn0), mode="lower", diag=FALSE, weighted="Weight")

# Check for identical edge attributes

hist(edge\_attr(gzallt.gene.cccn.g)[[1]], breaks=1000, col="deepskyblue", ylim=c(0,1200), echo=FALSE)

# to get the adjacency matrix back, but NOTE, edges are 0 or 1.

# gzallt.gene.cccn.mat <- as\_adjacency\_matrix(gzallt.gene.cccn.g, type="both")

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Here is how to make an edge list file

gzalltgenecccn.edges <- data.frame(as\_edgelist(gzallt.gene.cccn.g))

names(gzalltgenecccn.edges) <- c("Gene.1", "Gene.2")

gzalltgenecccn.edges$Weight <- edge\_attr(gzallt.gene.cccn.g)[[1]]

gzalltgenecccn.edges$interaction <- "correlation"

gzalltgenecccn.edges $interaction[gzalltgenecccn.edges$Weight<=-0.5] <- "negative correlation"

gzalltgenecccn.edges $interaction[gzalltgenecccn.edges$Weight>=0.5] <- "positive correlation"

write.table(gzalltgenecccn.edges,"CCCN.txt",quote = FALSE,row.names = FALSE,sep = "\t")

gzalltgenecccn.edges.1 <- gzalltgenecccn.edges[gzalltgenecccn.edges$interaction %in% c("negative correlation","positive correlation"),]

write.table(gzalltgenecccn.edges.1,"select\_CCCN\_corr.txt",quote = FALSE,row.names = FALSE,sep = "\t")

pcname <- "PathwayCommons11.All.hgnc.txt"

pcpath <- "PPI/"

pcfile <- paste(pcpath, pcname, sep="")

pcommons <- read.table(pcfile, header=TRUE, sep = "\t", comment.char = "#", na.strings='', stringsAsFactors=FALSE, fill=TRUE)

dim(pcommons)

pcnet <- pcommons[-grep("CHEBI:", pcommons$PARTICIPANT\_A),1:4]

pcnet <- pcnet[-grep("CHEBI:", pcnet$PARTICIPANT\_B),]

# Add an arbitrary Weight and rearrange to be consistent with other PPI networks

pcnet$Weight <- 0.2

pcnet <- pcnet[, c(1,3,5,2,4)]

names(pcnet) <- c("Gene.1", "Gene.2", "Weight", "edgeType", "Source")

save(pcnet, file=paste(pcpath,"pcnet.RData", sep=""))

pcnet1 <- pcnet[pcnet$edgeType == "interacts-with",]

bpfilename <- "PPI/BioPlex\_2019.tsv"

bioplex <- read.table(bpfilename, header=TRUE, sep = "\t", comment.char = "", na.strings='', stringsAsFactors=FALSE, fill=TRUE)

dim(bioplex)

# 118162 edges; pInt values range from 0.75 to 1.00

bioplex <- bioplex[,c("SymbolA", "SymbolB", "pInt")]

names(bioplex) <- c("Gene.1", "Gene.2", "Weight")

bioplex$Source <- "BioPlex"

save(pcommons, bioplex, pcnet, file=paste(pcpath,"pcommons\_2019.RData", sep=""))

library(STRINGdb)

string\_db <- STRINGdb$new( version="11.5", species=9606, score\_threshold=0, input\_directory="" )

string\_proteins <- string\_db$get\_proteins()

bioplex1 <- bioplex

bioplex1$edgeType <- c("interacts-with")

bioplex1 <- bioplex1[,c(1,2,3,5,4)]

PPImerge <- rbind(bioplex1,pcnet1)

PPImerge1 <- PPImerge[!duplicated(PPImerge[, c("Gene.1", "Gene.2")]), ]

ProteinCCCN <- read.table("CCCN.txt",header = TRUE,sep = "\t")

combinedgenes <- rownames(gzallt.gene.cccn0)

combined.pc.edges <- filter.edges.0(combinedgenes, pcnet) # 119255 edges

combined.bp.edges <- filter.edges.0(combinedgenes, bioplex) # 6328 edges

# combined.bp.edges <- combined.bp.edges[,c(1,2,4,3)]

combined.enzsub.edges <- filter.edges.0(combinedgenes, enzsub) # 4210 edges

#

combined.kinsub.edges <- filter.edges.0(combinedgenes, kinsub) # 4086 edges; 4428 with 2019 kinsub

# Make and "all ppi edge" file for comparisone

combined.all.ppi <- rbind(combined.pc.edges[,c(1,2,3,5)], combined.bp.edges)

# 1572925 edges! 1577353 with new kinsub

combined.all.ppi.g <- graph\_from\_data\_frame(combined.all.ppi)

# 927576 edges; 1009527 with kinsub

combined.all.ppi.g1 <- as.data.frame(combined.all.ppi.g)

# Make CFN with PPI edges

# Test whether combined.all.ppi from KGFunDataObjects.RData covers the genes

gzallt.genes.1 <- unique(c(gzalltgenecccn.edges$Gene.1, gzalltgenecccn.edges$Gene.2))

combined.ppi.genes <- unique(c(combined.all.ppi$Gene.1, combined.all.ppi$Gene.2))

leftover <- setdiff(gzallt.genes.1, combined.ppi.genes)

ppi.g <- as\_data\_frame(combined.all.ppi.g)

write.table(ppi.g,"CFN\_data.txt",quote = FALSE,row.names = FALSE,sep = "\t")