# =====================================================

# CfWRKY WGCNA 分析流程（加入 TPM 过滤）

# =====================================================

suppressMessages({

library(WGCNA)

library(tidyverse)

library(readxl)

library(matrixStats)

library(gplots)

library(igraph)

})

# --- 多线程设置 ---

enableWGCNAThreads(nThreads = 8)

allowWGCNAThreads(nThreads = 8)

options(stringsAsFactors = FALSE)

# --- 文件路径 ---

exp\_file <- "/home/data/t210549/liuhuacheng/wgcna\_wrky/tpm\_matrix.txt"

trait\_file <- "/home/data/t210549/liuhuacheng/wgcna\_wrky/pheno\_data.csv"

wrky\_file <- "/home/data/t210549/liuhuacheng/wgcna\_wrky/gene\_info.xlsx"

output\_dir <- "/home/data/t210549/liuhuacheng/wgcna\_wrky/output"

if (!dir.exists(output\_dir)) dir.create(output\_dir, recursive = TRUE)

# =====================================================

# TPM 读取 + 数据过滤处理

# =====================================================

exp <- read.table(exp\_file, row.names = 1, check.names = FALSE)

message("✅ 原始表达矩阵：", nrow(exp), " 基因 × ", ncol(exp), " 样本")

# --- 定义过滤函数（同你上面的）---

filter\_low\_expression <- function(expr\_matrix, min\_tpm = 0.5, min\_samples\_ratio = 0.1) {

nsamples <- ncol(expr\_matrix)

keep <- rowSums(expr\_matrix >= min\_tpm) >= (min\_samples\_ratio \* nsamples)

return(expr\_matrix[keep, ])

}

# --- Step 1: 去除低表达 ---

filtered\_tpm <- filter\_low\_expression(exp, min\_tpm = 0.5, min\_samples\_ratio = 0.1)

message("✅ 低表达过滤后剩余基因数：", nrow(filtered\_tpm))

# --- Step 2: MAD 筛选 ---

mad\_vals <- rowMads(as.matrix(filtered\_tpm))

mad\_rank\_cutoff <- quantile(mad\_vals, probs = 0.25)

keep\_mad <- mad\_vals > mad\_rank\_cutoff

TPM\_filt2 <- filtered\_tpm[keep\_mad, ]

message("✅ MAD 过滤后剩余基因数：", nrow(TPM\_filt2))

# --- Step 3: log2 转换 ---

TPM\_log2 <- log2(TPM\_filt2 + 1)

# --- Step 4: 转置供 WGCNA 使用 ---

datExpr <- as.data.frame(t(TPM\_log2))

message("✅ 过滤后矩阵：", nrow(datExpr), " 样本 × ", ncol(datExpr), " 基因")

# =====================================================

# 样本表型匹配

# =====================================================

trait <- read.csv(trait\_file, fileEncoding = "UTF-8-BOM")

colnames(trait)[1] <- "sample"

rownames(trait) <- trait$sample

common\_samples <- intersect(rownames(datExpr), rownames(trait))

datExpr <- datExpr[common\_samples, , drop = FALSE]

trait <- trait[common\_samples, , drop = FALSE]

message("✅ 匹配样本数：", length(common\_samples))

# =====================================================

# 样本聚类检测异常

# =====================================================

sampleTree <- hclust(dist(datExpr), method = "average")

pdf(file.path(output\_dir, "SampleTree.pdf"), width = 8, height = 6)

plot(sampleTree, main = "Sample clustering to detect outliers", xlab = "", sub = "")

dev.off()

# =====================================================

# 表型热图

# =====================================================

datTraitsNumeric <- as.data.frame(lapply(trait, function(x) as.numeric(as.factor(x))))

traitColors <- numbers2colors(datTraitsNumeric, signed = FALSE)

pdf(file.path(output\_dir, "SampleTraitHeatmap.pdf"), width = 10, height = 6)

plotDendroAndColors(sampleTree, traitColors,

groupLabels = names(datTraitsNumeric),

main = "Sample dendrogram and trait heatmap")

dev.off()

# =====================================================

# 软阈值选择

# =====================================================

powers <- c(1:10, seq(12, 20, 2))

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

softPower <- 8

message("✅ 使用软阈值 Power =", softPower)

pdf(file.path(output\_dir, "SoftThreshold.pdf"), width = 10, height = 5)

par(mfrow = c(1, 2))

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

xlab="Soft Threshold (power)", ylab="Scale Free Topology Model Fit, signed R²",

main="Scale independence", type="n")

text(sft$fitIndices[,1], -sign(sft$fitIndices[,3])\*sft$fitIndices[,2], labels=powers, col="red")

abline(h=0.85, col="red")

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

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

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

dev.off()

# =====================================================

# 构建模块

# =====================================================

net <- blockwiseModules(datExpr, power = softPower,

TOMType = "signed", minModuleSize = 50,

mergeCutHeight = 0, numericLabels = TRUE,

saveTOMs = FALSE, verbose = 3)

moduleColors <- labels2colors(net$colors)

pdf(file.path(output\_dir, "ModuleDendrogram.pdf"), width = 9, height = 6)

plotDendroAndColors(net$dendrograms[[1]],

moduleColors[net$blockGenes[[1]]],

"Module colors", dendroLabels = FALSE, hang = 0.03)

dev.off()

# =====================================================

# 模块–表型相关性热图

# =====================================================

trait$treatment\_group <- as.factor(trait$treatment\_group)

design <- model.matrix(~0 + trait$treatment\_group)

colnames(design) <- levels(trait$treatment\_group)

MES0 <- moduleEigengenes(datExpr, moduleColors)$eigengenes

MEs <- orderMEs(MES0)

moduleTraitCor <- cor(MEs, design, use = "p")

moduleTraitPvalue <- corPvalueStudent(moduleTraitCor, nrow(datExpr))

textMatrix <- paste0(signif(moduleTraitCor, 2), "\n(",

signif(moduleTraitPvalue, 1), ")")

dim(textMatrix) <- dim(moduleTraitCor)

pdf(file.path(output\_dir, "ModuleTrait\_Correlation\_Heatmap.pdf"),

width = 2 \* ncol(design),

height = 0.35 \* ncol(MEs))

labeledHeatmap(Matrix = moduleTraitCor,

xLabels = colnames(moduleTraitCor),

yLabels = names(MEs),

ySymbols = names(MEs),

colorLabels = FALSE,

colors = blueWhiteRed(50),

textMatrix = textMatrix,

cex.text = 0.6,

cex.lab.y = 0.8,

zlim = c(-1, 1),

main = "Module-Trait Relationships")

dev.off()

message("✅ 模块–表型相关性热图已保存。")

# =====================================================

# 只保留“干旱性状显著相关”的模块

# =====================================================

# --- BH 调整 ---

p\_adj <- apply(moduleTraitPvalue, 2, p.adjust, method = "BH")

# --- 找到 drought 列 ---

d\_col <- grep("^drought$", colnames(moduleTraitCor), ignore.case = TRUE, value = TRUE)

if (length(d\_col) == 0) stop("❌ 未找到名为 'drought' 的列，请检查 trait 文件或设计矩阵列名！")

# --- 显著性筛选条件（仅针对干旱列）---

sig\_mask <- (abs(moduleTraitCor[, d\_col]) >= 0.5) & (p\_adj[, d\_col] < 0.05)

keep\_rows <- sig\_mask # 每行一个 TRUE/FALSE

# --- 只保留显著模块 ---

cor\_sub <- moduleTraitCor[keep\_rows, , drop = FALSE]

p\_sub <- moduleTraitPvalue[keep\_rows, , drop = FALSE]

p\_adj\_sub <- p\_adj[keep\_rows, , drop = FALSE]

if (nrow(cor\_sub) == 0) {

warning("⚠️ 没有模块在干旱条件下达到显著性阈值 (|r|≥0.5 且 FDR<0.05)。")

}

# --- 合并文字矩阵（显示 FDR）---

text\_sub <- paste0(signif(cor\_sub, 2), "\n", signif(p\_adj\_sub, 2))

dim(text\_sub) <- dim(cor\_sub)

# --- 可选排序：按干旱相关性降序 ---

ord <- order(abs(cor\_sub[, d\_col]), decreasing = TRUE)

if (length(ord) > 0) {

cor\_sub <- cor\_sub[ord, , drop = FALSE]

p\_sub <- p\_sub[ord, , drop = FALSE]

text\_sub <- text\_sub[ord, , drop = FALSE]

}

# --- 根据模块和性状数量自动调整图像尺寸 ---

n\_modules <- nrow(cor\_sub)

n\_traits <- ncol(cor\_sub)

plot\_width <- max(6, min(2 + 1.2 \* n\_traits, 18))

plot\_height <- max(6, min(4 + 0.25 \* n\_modules, 20))

# --- 自动调整文字大小 ---

cex\_text\_val <- max(0.4, min(1.0, 0.8 - 0.015 \* max(n\_modules, n\_traits)))

cex\_lab\_val <- max(0.6, min(1.0, 0.9 - 0.01 \* n\_modules))

cat(sprintf("自动调整文字大小: cex.text=%.2f, cex.lab.y=%.2f\n", cex\_text\_val, cex\_lab\_val))

# --- 绘制显著模块-表型关系热图 ---

pdf(file.path(output\_dir, "ModuleTrait\_Correlation\_DROUGHT\_SIG\_only.pdf"),

width = plot\_width, height = plot\_height)

labeledHeatmap(Matrix = cor\_sub,

xLabels = colnames(cor\_sub),

yLabels = rownames(cor\_sub),

ySymbols = rownames(cor\_sub),

colorLabels = FALSE,

colors = blueWhiteRed(50),

textMatrix = text\_sub,

cex.text = cex\_text\_val,

cex.lab.y = cex\_lab\_val,

zlim = c(-1, 1),

main = "Significant Modules Sig Drought")

dev.off()

message("✅ 干旱显著模块热图已保存：ModuleTrait\_Correlation\_DROUGHT\_SIG\_only.pdf")

message("👉 保留下来的模块数：", n\_modules)

# =====================================================

# 计算 TOM 相似度矩阵

# =====================================================

message("🧮 计算 TOM 相似度矩阵...")

TOM <- TOMsimilarityFromExpr(datExpr, power = softPower, corType = "bicor")

rownames(TOM) <- colnames(datExpr)

colnames(TOM) <- colnames(datExpr)

# =====================================================

# 🚀 Step 8: 模块 TF-Hub 共表达网络分析（含 WRKY 命名替换）

# =====================================================

library(igraph)

# 参数设置 ----------------------------------------------------

target\_module <- "plum2" # 指定模块

kME\_cutoff <- 0.9 # Hub 判定阈值

num\_top\_connections <- NULL # 每个 Hub 的 topN 伙伴

tf\_file <- "/home/data/t210549/liuhuacheng/wgcna\_wrky/output/TF\_gene\_family.xlsx"

wrky\_file <- "/home/data/t210549/liuhuacheng/wgcna\_wrky/gene\_info.xlsx"

# =====================================================

# 1️⃣ 提取模块内基因

# =====================================================

module\_genes <- names(datExpr)[moduleColors == target\_module]

message("✅ 模块 ", target\_module, " 基因数：", length(module\_genes))

# =====================================================

# 2️⃣ 计算 kME

# =====================================================

MEs <- moduleEigengenes(datExpr, moduleColors)$eigengenes

MEs <- orderMEs(MEs)

kME <- signedKME(datExpr, MEs, outputColumnName = "kME")

module\_of\_interest <- paste0("kME", target\_module)

sub\_kME <- kME[module\_genes, module\_of\_interest, drop = FALSE]

# =====================================================

# 3️⃣ 读取 TF 与 WRKY 信息表

# =====================================================

TF\_info <- read\_xlsx(tf\_file) %>%

rename(GENEID = GENE, FAMILY = FAMILY)

WRKY\_info <- read\_xlsx(wrky\_file) %>%

rename(GENEID = GENEID, GENENAME = GENENAME)

tf\_ids <- unique(TF\_info$GENEID)

wrky\_ids <- unique(WRKY\_info$GENEID)

# =====================================================

# 4️⃣ 模块内 TF 与 Hub 基因识别

# =====================================================

tf\_in\_module <- intersect(module\_genes, tf\_ids)

hub\_genes <- rownames(sub\_kME)[sub\_kME[, 1] > kME\_cutoff]

hub\_TFs <- intersect(hub\_genes, tf\_in\_module)

message("🏆 模块 ", target\_module, " 内 Hub 基因数：", length(hub\_genes))

message("💡 其中 Hub TF 数：", length(hub\_TFs))

# =====================================================

# 5️⃣ 提取每个 Hub 基因的 topN 共表达伙伴

# =====================================================

message("🔍 提取每个 hub 基因的 top ", num\_top\_connections, " 个共表达伙伴...")

# 假定 num\_top\_connections <- NULL 或是一个正整数

all\_edges <- map\_dfr(hub\_genes, function(gene) {

cor\_vec <- TOM[gene, module\_genes]

cor\_vec <- cor\_vec[!is.na(cor\_vec)]

cor\_vec <- cor\_vec[names(cor\_vec) != gene] # 去掉自连

partners <- sort(cor\_vec, decreasing = TRUE)

# 当 num\_top\_connections 为 NULL 时，取所有 partners；否则取 top N

if (!is.null(num\_top\_connections)) {

partners <- head(partners, num\_top\_connections)

}

# 如果没有 partners，则返回空 tibble（被 map\_dfr 自动合并）

if (length(partners) == 0) {

return(tibble(source = character(0), target = character(0), weight = numeric(0)))

}

tibble(source = gene,

target = names(partners),

weight = as.numeric(partners))

})

# =====================================================

# 6️⃣ 添加 Hub–Hub 边（不受 topN 限制）

# =====================================================

hub\_pairs <- expand.grid(source = hub\_genes, target = hub\_genes, stringsAsFactors = FALSE) %>%

filter(source != target)

hub\_edges <- hub\_pairs %>%

mutate(weight = map2\_dbl(source, target, ~ TOM[.x, .y])) %>%

filter(!is.na(weight))

all\_edges <- bind\_rows(all\_edges, hub\_edges)

# =====================================================

# 7️⃣ 去重 & 去自连（无向网络）

# =====================================================

network\_edges <- all\_edges %>%

filter(source != target) %>%

mutate(node1 = pmin(source, target),

node2 = pmax(source, target)) %>%

distinct(node1, node2, .keep\_all = TRUE) %>%

select(source = node1, target = node2, weight) %>%

arrange(desc(weight))

message("✅ 最终共表达边数：", nrow(network\_edges))

# =====================================================

# 8️⃣ 构建节点注释表（含 WRKY 替换 + kME 值 + 分组标记）

# =====================================================

all\_nodes <- unique(c(network\_edges$source, network\_edges$target))

# 从 kME 表中提取每个节点的 kME 值

kME\_values <- tibble(GENEID = rownames(kME),

kME\_value = kME[, module\_of\_interest])

node\_info <- tibble(GENEID = all\_nodes) %>%

left\_join(TF\_info, by = "GENEID") %>%

left\_join(WRKY\_info, by = "GENEID") %>%

left\_join(kME\_values, by = "GENEID") %>%

mutate(

DisplayName = ifelse(!is.na(GENENAME), GENENAME, GENEID), # WRKY 命名替换

isTF = !is.na(FAMILY),

isHub = GENEID %in% hub\_genes,

isHubTF = GENEID %in% hub\_TFs,

Group = ifelse(GENEID %in% hub\_TFs, "HubTF", "Normal") # 🌟 新增分组列

)

# =====================================================

# 9️⃣ 边表也替换 WRKY 名称 + 增加组别标记列

# =====================================================

replace\_wrky <- function(id) {

ifelse(id %in% WRKY\_info$GENEID,

WRKY\_info$GENENAME[match(id, WRKY\_info$GENEID)],

id)

}

# 替换 WRKY 名称

network\_edges <- network\_edges %>%

mutate(source = replace\_wrky(source),

target = replace\_wrky(target))

# 🔹 创建分组标签：根据节点类型标记边的来源与目标类型

node\_groups <- node\_info %>% select(GENEID, Group)

network\_edges <- network\_edges %>%

left\_join(node\_groups, by = c("source" = "GENEID")) %>%

rename(SourceGroup = Group) %>%

left\_join(node\_groups, by = c("target" = "GENEID")) %>%

rename(TargetGroup = Group) %>%

mutate(

EdgeGroup = case\_when(

SourceGroup == "HubTF" & TargetGroup == "HubTF" ~ "HubTF–HubTF",

SourceGroup == "HubTF" | TargetGroup == "HubTF" ~ "HubTF–Normal",

TRUE ~ "Normal–Normal"

)

)

# =====================================================

# 🔟 输出结果文件

# =====================================================

node\_output <- file.path(output\_dir, paste0("Module\_", target\_module, "\_TF\_WRKY\_Hub\_Nodes.csv"))

edge\_output <- file.path(output\_dir, paste0("Module\_", target\_module, "\_TF\_WRKY\_Hub\_top", num\_top\_connections, "\_Network\_All.csv"))

write.csv(node\_info, node\_output, row.names = FALSE)

write.csv(network\_edges, edge\_output, row.names = FALSE)

message("📁 导出节点文件：", node\_output)

message("📁 导出网络文件：", edge\_output)

message("📊 网络包含 ", nrow(network\_edges), " 条边，涉及 ", length(unique(c(network\_edges$source, network\_edges$target))), " 个节点。")

message("🔥 模块 ", target\_module, " 中 Hub TF 数量：", length(hub\_TFs))  
  
#####

备注：cyan模块weight值大于0.2，kME值大于0.9（边和hub基因过多）  
hubWRKY:节点：#B9DFB6，边：3B3DFC;Size=80,With=12

HubTF边：#E6DBE5,With=8

其他：With=0.5  
#####

#####

#####  
#####

备注：plum2模块weight值大于或等于0.1，kME值大于0.9（边和hub基因过多）

#####