Figure 3 — WGCNA (subclusters alpha vs beta)

MAXOMOD_CSF

2026-02-16

Contents

Overview

This document reproduces **Figure 3** from the WGCNA subclusters pipeline by running the same data preparation and plotting code as `Script/10_WGCNA_subclusters.R` and `Script/11_WGCNA_comparison_subclusters.R`.

- **Fig 3a:** Dendrogram and module–trait correlations
- **Fig 3b:** Eigengene networks
- **Fig 3c:** Top GO terms per module (IC heatmap)
- **Fig 3d:** Eigenprotein boxplots (turquoise & blue) in Discovery, Validation, External
- **Fig 3e:** Module–trait heatmap (Discovery)
- **Fig 3f:** Module–trait heatmap (Validation)
- **Fig 3g:** MEs vs clinical variables

Paths are relative to the **project root**. Run the **params** chunk first when running interactively. The first full run may take several minutes (WGCNA + GO enrichment).

Setup

Load packages

```
library(SummarizedExperiment)
library(tidyverse)
library(ggplot2)
library(ggpubr)
library(RColorBrewer)
library(gridExtra)
library(WGCNA)
library(org.Hs.eg.db)
library(clusterProfiler)
library(GOSemSim)
library(pheatmap)
library(tidyr)
library(ComplexHeatmap)
library(circlize)
library(grid)
library(readxl)
library(dplyr)
allowWGCNAThreads()
```

```
## Allowing multi-threading with up to 11 threads.
```

Project root and parameters

```
project_root <- if (basename(getwd()) == "vignettes") {
  normalizePath("..", winslash = "/")
```

```
} else {
  getwd()
}
setwd(project_root)
message("Project root: ", project_root)
```

```
## Project root: /Users/xliu2942/Documents/Projects/MAXOMOD/MAXOMOD_CSF
```

```
discovery_input  <- file.path(project_root, params$discovery_input)
discovery_cluster <- file.path(project_root, params$discovery_cluster)
validation_input <- file.path(project_root, params$validation_input)
validation_cluster <- file.path(project_root, params$validation_cluster)
external_input   <- file.path(project_root, params$external_input)
external_cluster <- file.path(project_root, params$external_cluster)
output_dir       <- file.path(project_root, params$output)
set.seed(params$seed)
top_n_GO         <- params$top_n_GO
```

```
if (!dir.exists(output_dir)) dir.create(output_dir, recursive = TRUE)
```

Helper functions

```
# prepare_toplot: align MEs with cluster assignments and metadata (Script 11 version with
↪  ModuleTrait)
prepare_toplot <- function(MEs, cluster_file, metaData, cluster_col, dat_matrix,
↪  ModuleTrait = TRUE) {
  cluster_assignments <- read.csv(cluster_file)
  cluster_assignments <- cluster_assignments[match(colnames(dat_matrix),
↪  cluster_assignments$patid), ]
  metaData <- metaData %>% dplyr::select(-starts_with("kmeans_k"))
  factorMeta <- left_join(cluster_assignments, metaData, by = c("patid" = "label"))
  cluster_num <- as.numeric(gsub("\\D", "", cluster_col))
  colnames(factorMeta)[which(colnames(factorMeta) == paste0("kmeans_k.", cluster_num))]
  ↪  <- cluster_col
  factorMeta[[cluster_col]] <- as.factor(factorMeta[[cluster_col]])
  toplot <- t(MEs)
  toplot <- toplot[, factorMeta$patid]
  if (ModuleTrait) {
    want <- c(cluster_col, "sex", "condition", "genetics")
    factorMeta <- factorMeta[, intersect(want, colnames(factorMeta)), drop = FALSE]
  } else {
    factorMeta <- factorMeta[, cluster_col, drop = FALSE]
  }
  list(toplot = toplot, factorMeta = factorMeta)
}

calculate_pvalues <- function(MEs, toplot, factorMeta, cluster_col) {
```

```
  group <- factorMeta[[cluster_col]]
  num_groups <- length(unique(group))
  pvec <- numeric(ncol(MEs))
  for (i in seq_len(ncol(MEs))) {
    y <- MEs[, i]
    if (num_groups == 2) {
      pval <- tryCatch(wilcox.test(y ~ group)$p.value, error = function(e) NA)
    } else {
      pval <- tryCatch(kruskal.test(y ~ group)$p.value, error = function(e) NA)
    }
    pvec[i] <- pval
  }
  names(pvec) <- colnames(MEs)
  pvec <- pvec[match(names(pvec), rownames(toplot))]
  rownames(toplot) <- paste(rownames(toplot), "\np = ", signif(pvec, 2), sep = "")
  toplot
}

# MEs_ref: MEs matrix used for module names. modules_to_show: only plot these, in this
↪  order (left to right)
plot_boxplots <- function(toplot, factorMeta, cluster_col, MEs_ref, main_prefix = "",
↪  modules_to_show = c("turquoise", "blue")) {
  idx_all <- setdiff(seq_len(nrow(toplot)), which(colnames(MEs_ref) == "grey"))
  # Order indices so panels appear left-to-right as in modules_to_show (e.g. turquoise
  ↪  then blue)
  idx <- match(modules_to_show, colnames(MEs_ref)[idx_all])
  idx <- idx_all[idx[!is.na(idx)]]
  n_plots <- length(idx)
  if (n_plots == 0) return(invisible(NULL))
  par(mfrow = c(1, n_plots), mar = c(3, 3, 2, 1))
  for (i in idx) {
    boxplot(toplot[i, ] ~ factorMeta[[cluster_col]], col = colnames(MEs_ref)[i],
            ylab = "Eigenprotein", main = paste0(main_prefix, rownames(toplot)[i]), xlab
            ↪  = NULL, cex.main = 0.9)
  }
  par(mfrow = c(1, 1))
}

# Top GO IC heatmap (from Script 10)
plot_top_ic_heatmap <- function(net, enrichment_list_with_IC, top_n = 3, fill_na = TRUE)
↪  {
  desired_order <- setdiff(gsub("^ME", "", colnames(net$MEs)), "grey")
  go_ic_top_df <- lapply(names(enrichment_list_with_IC), function(module) {
    df <- enrichment_list_with_IC[[module]]
    if (nrow(df) == 0) return(NULL)
    df %>% arrange(p.adjust, IC) %>% slice_head(n = top_n) %>% mutate(Module = module)
  }) %>% bind_rows()
  if (nrow(go_ic_top_df) == 0) return(invisible(NULL))
  heatmap_mat <- go_ic_top_df %>%
    mutate(logpadj = -log10(p.adjust)) %>%
    dplyr::select(Module, Description, logpadj) %>%
    pivot_wider(names_from = Description, values_from = logpadj) %>%
    column_to_rownames("Module")
  star_matrix <- go_ic_top_df %>%
```

```
    mutate(stars = case_when(
      p.adjust < 0.001 ~ "***", p.adjust < 0.01 ~ "**", p.adjust < 0.05 ~ "*", TRUE ~ ""
    )) %>%
    dplyr::select(Module, Description, stars) %>%
    pivot_wider(names_from = Description, values_from = stars) %>%
    column_to_rownames("Module")
  heatmap_mat <- heatmap_mat[intersect(desired_order, rownames(heatmap_mat)), , drop =
↪  FALSE]
  star_matrix <- star_matrix[rownames(heatmap_mat), , drop = FALSE]
  if (fill_na) {
    heatmap_mat[is.na(heatmap_mat)] <- 0
    star_matrix[is.na(star_matrix)] <- ""
  }
  ph <- pheatmap::pheatmap(
    mat = heatmap_mat,
    cluster_rows = FALSE, cluster_cols = TRUE,
    color = colorRampPalette(c("white", "red"))(100),
    border_color = "black",
    display_numbers = as.matrix(star_matrix),
    number_color = "white", fontsize_number = 12, fontsize = 10,
    main = paste0("Top ", top_n, " GO Terms per Module (Min IC)"), angle_col = 90
  )
  invisible(ph)
}
```

Load Discovery data and run WGCNA

```
if (!file.exists(discovery_input)) stop("Discovery input not found: ", discovery_input)
cleanDat_file <- readRDS(discovery_input)
metaData_disc <- as.data.frame(colData(cleanDat_file))
cleanDat_disc <- assay(cleanDat_file)
colnames(cleanDat_disc) <- metaData_disc$label
rownames(metaData_disc) <- metaData_disc$label

powers <- seq(2, 15, by = 1)
sft <- pickSoftThreshold(t(cleanDat_disc), powerVector = powers, networkType = "signed",
↪  RsquaredCut = 0.85)
```

```
##    Power SFT.R.sq  slope truncated.R.sq mean.k. median.k. max.k.
## 1      2   0.1890  6.700          0.986  144.00   143.000  164.0
## 2      3   0.0161  0.746          0.728   83.10    81.600  108.0
## 3      4   0.3080 -1.450          0.747   50.80    48.500   78.8
## 4      5   0.7440 -1.820          0.882   32.60    29.800   61.6
## 5      6   0.9270 -1.810          0.978   21.80    18.900   50.7
## 6      7   0.9460 -1.720          0.958   15.20    12.200   42.8
## 7      8   0.9420 -1.630          0.951   10.90     8.090   36.9
## 8      9   0.9490 -1.550          0.946    8.08     5.360   32.3
## 9     10   0.9170 -1.520          0.904    6.15     3.560   28.5
```

```
## 10    11   0.9170 -1.460          0.906    4.79     2.470   25.5
## 11    12   0.9380 -1.400          0.930    3.81     1.800   22.9
## 12    13   0.9550 -1.360          0.945    3.08     1.280   20.7
## 13    14   0.9390 -1.330          0.922    2.54     0.913   18.8
## 14    15   0.9400 -1.290          0.923    2.11     0.652   17.2
```

```
cleanDat <- cleanDat_disc
net <- blockwiseModules(t(cleanDat), power = sft$powerEstimate, deepSplit = 4,
↪  minModuleSize = 6,
                        mergeCutHeight = 0.07, corType = "pearson", networkType =
                        ↪  "signed",
                        pamStage = TRUE, pamRespectsDendro = TRUE, reassignThreshold =
                        ↪  0.05,
                        verbose = 0, saveTOMs = FALSE, maxBlockSize = 10000)

MEList <- moduleEigengenes(t(cleanDat), colors = net$colors)
MEs_disc <- MEList$eigengenes
MEs_disc <- MEs_disc[, colnames(MEs_disc) != "MEgrey"]
colnames(MEs_disc) <- substr(colnames(MEs_disc), 3, 100)
rownames(MEs_disc) <- colnames(cleanDat)

if (sum(!is.na(metaData_disc$pNFh)) > 20) {
  numericMeta_disc <- metaData_disc[, c("age", "Nfl", "pNFh", "progression_rate",
↪  "slow_vital_capacity", "age_at_onset", "disease_duration")]
} else {
  numericMeta_disc <- metaData_disc[, c("age", "Nfl", "progression_rate",
↪  "slow_vital_capacity", "age_at_onset", "disease_duration")]
}
numericMeta_disc <- numericMeta_disc[match(colnames(cleanDat),
↪  rownames(numericMeta_disc)), ]
```

Figure 3a — WGCNA dendrogram and module–trait correlations

```
geneSignificance <- cor(numericMeta_disc, t(cleanDat), use = "pairwise.complete.obs")
geneSigColors <- t(WGCNA::numbers2colors(t(geneSignificance), signed = TRUE, lim = c(-1,
↪  1), naColor = "black"))
rownames(geneSigColors) <- colnames(numericMeta_disc)

par(mar = c(2, 4, 2, 2))
plotDendroAndColors(
  dendro = net$dendrograms[[1]],
  colors = t(rbind(net$colors, geneSigColors)),
  cex.dendroLabels = 1.2, addGuide = TRUE, dendroLabels = FALSE,
  groupLabels = c("Module Colors", colnames(numericMeta_disc))
)
```

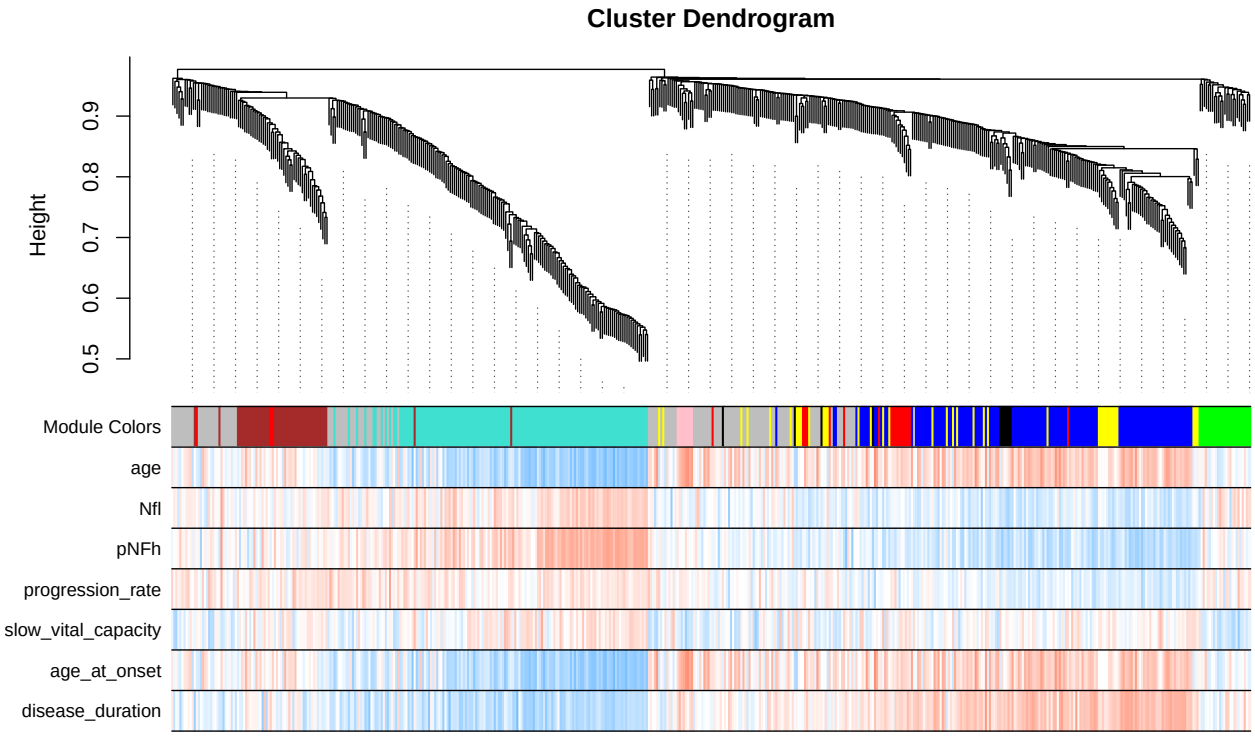

Figure 1: Figure 3a — WGCNA dendrogram and module–trait correlations.

```
pdf(file.path(output_dir, "Fig3a.pdf"), width = 10, height = 6)
par(mar = c(2, 4, 2, 2))
plotDendroAndColors(
  dendro = net$dendrograms[[1]],
  colors = t(rbind(net$colors, geneSigColors)),
  cex.dendroLabels = 1.2, addGuide = TRUE, dendroLabels = FALSE,
  groupLabels = c("Module Colors", colnames(numericMeta_disc))
)
dev.off()
```

```
## pdf
##   2
```

Figure 3b — Eigengene networks

```
MEs_plot <- net$MEs[, colnames(net$MEs) != "MEgrey"]
nMods <- ncol(MEs_plot)
par(mar = c(2, 2, 2, 2))
plotEigengeneNetworks(
  MEs_plot,
  setLabels = "Eigengene Network",
  marHeatmap = c(3, 4, 2, 2), marDendro = c(3, 4, 2, 2),
```

```
  plotDendrograms = TRUE, plotHeatmaps = TRUE, xLabelsAngle = 90,
  heatmapColors = WGCNA::blueWhiteRed(50),
  colorLabels = TRUE, coloredBarplot = TRUE, barplotMeans = TRUE
)
```

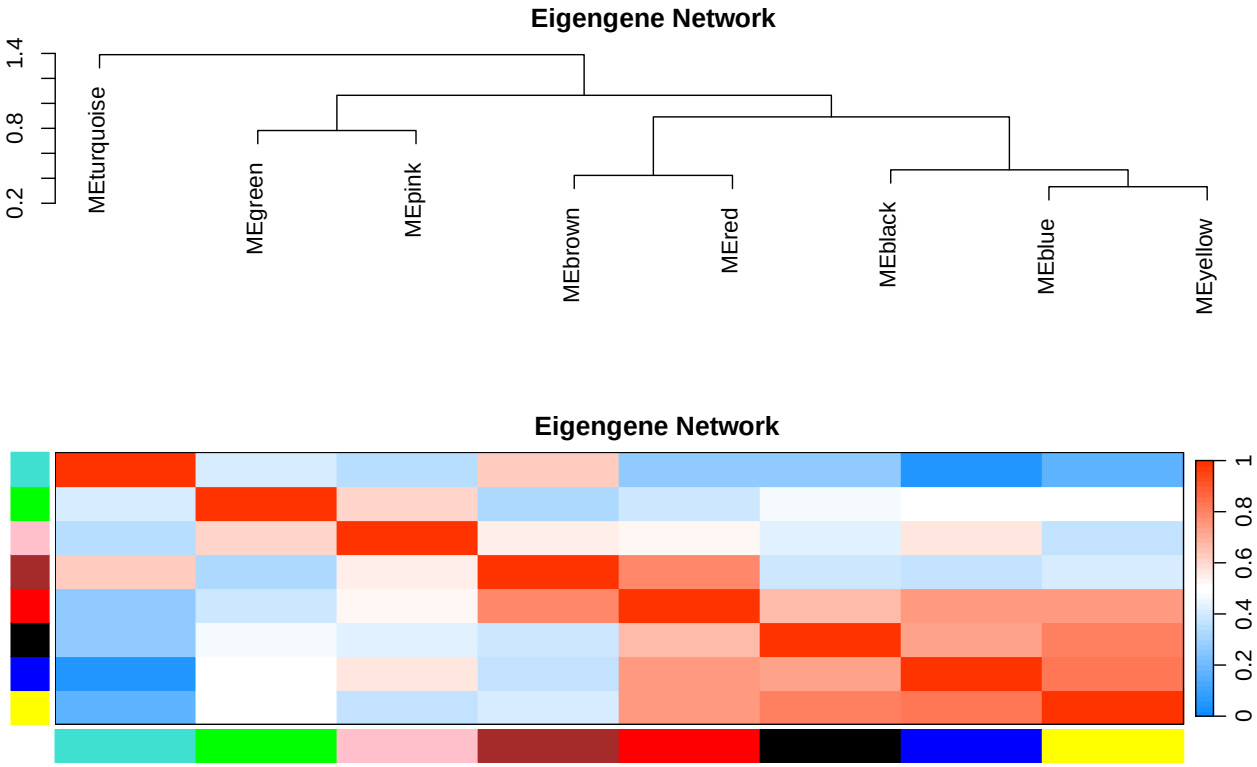

Figure 2: Figure 3b — WGCNA eigengene networks.

```
pdf(file.path(output_dir, "Fig3b.pdf"), width = 10, height = max(6, min(nMods * 0.4 + 4,
↪  15)))
par(mar = c(2, 2, 2, 2))
plotEigengeneNetworks(
  MEs_plot,
  setLabels = "Eigengene Network",
  marHeatmap = c(3, 4, 2, 2), marDendro = c(3, 4, 2, 2),
  plotDendrograms = TRUE, plotHeatmaps = TRUE, xLabelsAngle = 90,
  heatmapColors = WGCNA::blueWhiteRed(50),
  colorLabels = TRUE, coloredBarplot = TRUE, barplotMeans = TRUE
)
dev.off()
```

```
## pdf
##   2
```

GO enrichment and Figure 3c — Top GO terms per module

```
moduleLabels <- as.data.frame(net$colors)
colnames(moduleLabels) <- "color"
moduleLabels$gene <- rownames(moduleLabels)
moduleLabels_list <- split(moduleLabels, moduleLabels$color)
enrichment_list <- list()

for (color in setdiff(names(moduleLabels_list), "grey")) {
  genes <- moduleLabels[moduleLabels$color == color, ]$gene
  go_enrich <- enrichGO(gene = genes, OrgDb = org.Hs.eg.db, keyType = "SYMBOL", ont =
↪  "BP",
                        qvalueCutoff = 0.05, pAdjustMethod = "BH", minGSSize = 10)
  if (nrow(go_enrich) != 0) {
    go_enrich <- mutate(go_enrich, logpadj = -log10(p.adjust)) %>% filter(Count >= 3)
    enrichment_list[[color]] <- go_enrich@result
  }
}

enrichment_list_filtered <- lapply(enrichment_list, function(df) df %>% filter(p.adjust <
↪  0.1))
hsGO <- GOSemSim::godata("org.Hs.eg.db", ont = "BP")
```

```
## Warning in GOSemSim::godata("org.Hs.eg.db", ont = "BP"): use 'annoDb' instead
## of 'OrgDb'

## preparing gene to GO mapping data...

## preparing IC data...
```

```
enrichment_list_with_IC <- list()
for (module in names(enrichment_list_filtered)) {
  df <- enrichment_list_filtered[[module]]
  go_ids <- df$ID
  ic_values <- hsGO@IC[go_ids]
  df$IC <- ic_values[match(df$ID, names(ic_values))]
  enrichment_list_with_IC[[module]] <- df
}
```

```
p_3c <- plot_top_ic_heatmap(net, enrichment_list_with_IC, top_n = top_n_GO)
print(p_3c)
```

```
pdf(file.path(output_dir, "Fig3c.pdf"), width = 10, height = 7)
grid::grid.newpage()
grid::grid.draw(p_3c$gtable)
dev.off()
```

```
## pdf
##   2
```

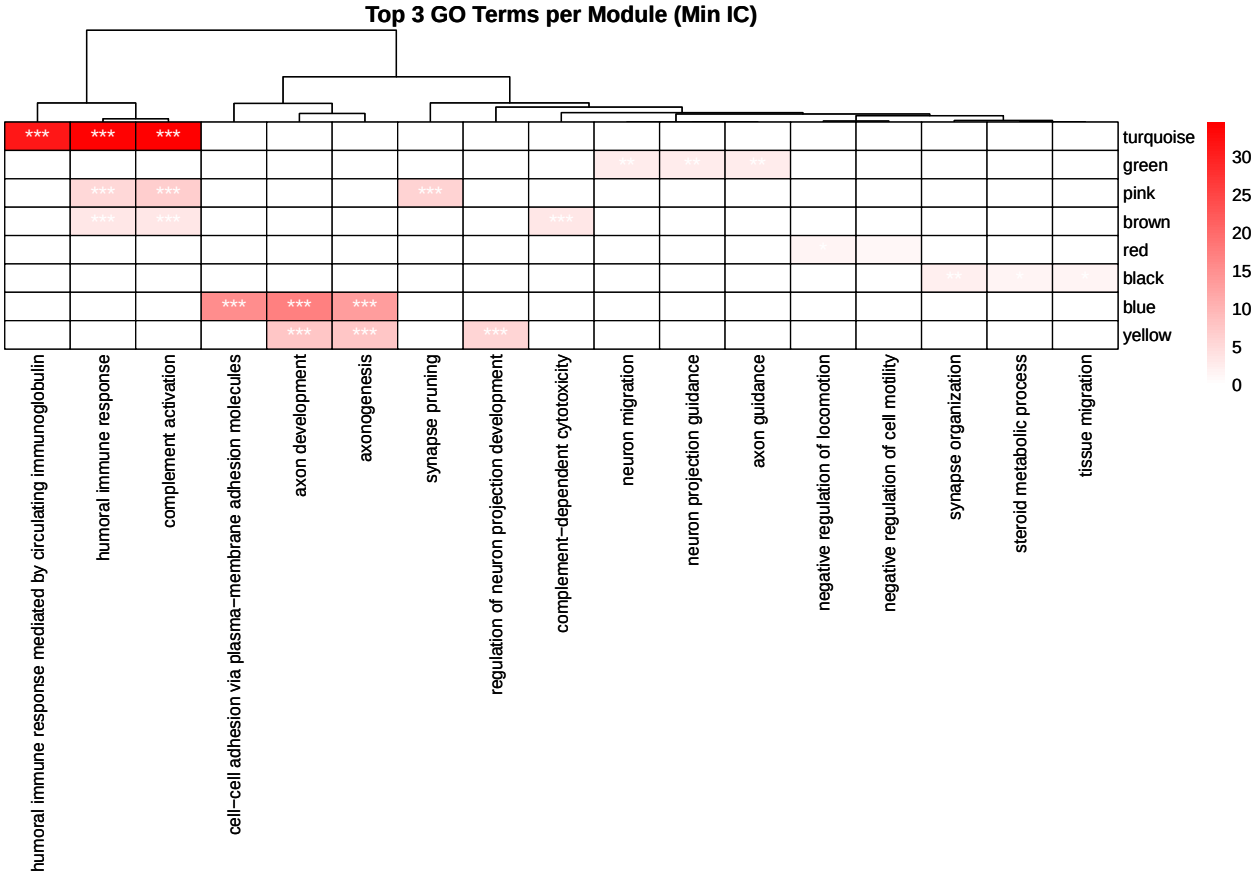

Figure 3: Figure 3c — Top GO terms per module (IC heatmap).

Figure 3d (Discovery) — Eigenprotein boxplots

```
data_k2_disc <- prepare_toplot(MEs_disc, discovery_cluster, metaData_disc, "k2",
↪  cleanDat_disc, ModuleTrait = TRUE)
toplot_k2_disc <- calculate_pvalues(MEs_disc, data_k2_disc$toplot,
↪  data_k2_disc$factorMeta, "k2")
plot_boxplots(toplot_k2_disc, data_k2_disc$factorMeta, "k2", MEs_disc, "Discovery: ")
```

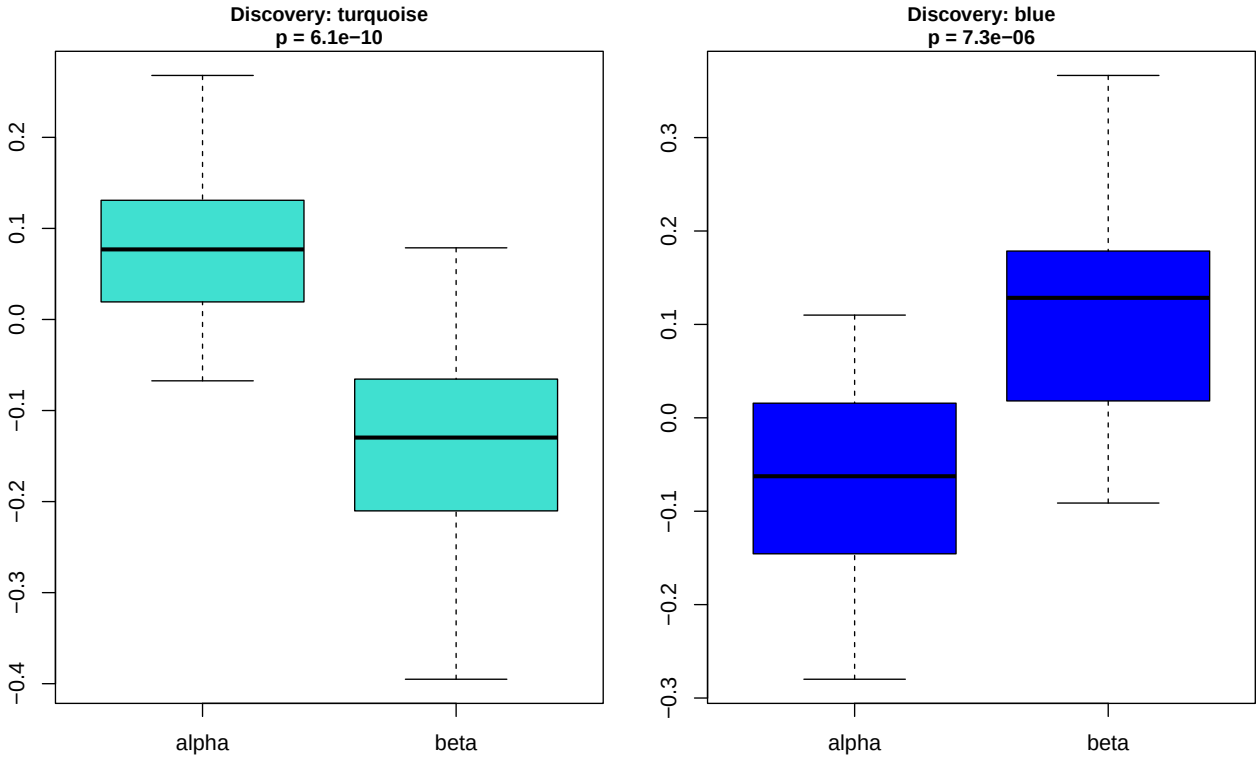

Figure 4: Figure 3d (Discovery) — k2 eigenprotein boxplots.

```
pdf(file.path(output_dir, "Fig3d_Discovery.pdf"), width = 10, height = 6)
plot_boxplots(toplot_k2_disc, data_k2_disc$factorMeta, "k2", MEs_disc, "Discovery: ")
dev.off()
```

```
## pdf
##   2
```

Figure 3e — Module–trait heatmap (Discovery)

```
moduleTraitCor_disc <- cor(MEs_disc, numericMeta_disc, use = "pairwise.complete.obs",
↪  method = "pearson")
```

```r
moduleTraitPvalue_disc <- WGCNA::corPvalueStudent(moduleTraitCor_disc, nrow(MEs_disc))
textMatrix_disc <- paste(signif(moduleTraitCor_disc, 2), "\n(",
↪  signif(moduleTraitPvalue_disc, 1), ")", sep = "")
dim(textMatrix_disc) <- dim(moduleTraitCor_disc)

white_threshold <- -log10(0.1)
signed_logP_disc <- sign(moduleTraitCor_disc) * -log10(moduleTraitPvalue_disc)
max_abs_logP <- 2
col_fun <- colorRamp2(
  c(-max_abs_logP, -white_threshold, 0, white_threshold, max_abs_logP),
  c("#4575B4", "white", "white", "white", "#D73027")
)
legend_breaks <- seq(-max_abs_logP, max_abs_logP, by = 1)

ht_3e <- Heatmap(
  matrix = signed_logP_disc,
  col = col_fun,
  cell_fun = function(j, i, x, y, w, h, col) {
    grid.text(textMatrix_disc[i, j], x, y, gp = gpar(fontsize = 8))
  },
  row_names_side = "left", cluster_rows = FALSE, cluster_columns = FALSE,
  show_row_names = TRUE, show_column_names = TRUE,
  row_names_gp = gpar(fontsize = 12), column_names_gp = gpar(fontsize = 12),
  column_names_rot = 45,
  column_title = "Module-trait relationships\npearson r-value\n(p-value)",
  border_gp = gpar(col = "black"),
  heatmap_legend_param = list(title = "signed -log10(p-value)", at = legend_breaks,
  ↪  labels = as.character(legend_breaks))
)
draw(ht_3e)
```

```r
# Save to PDF in a separate step so the document plot and file content match
pdf(file.path(output_dir, "Fig3e.pdf"), width = 7, height = 5.5)
draw(ht_3e)
dev.off()
```

```
## pdf
##   2
```

Figure 3g — MEs vs clinical variables

```r
plot_df <- cbind(net$MEs, numericMeta_disc)
module_names <- intersect(colnames(net$MEs), c("MEturquoise", "MEblue"))
clinical_vars <- colnames(numericMeta_disc)
plot_list <- list()
for (mod in module_names) {
  mod_color <- gsub("ME", "", mod)
```

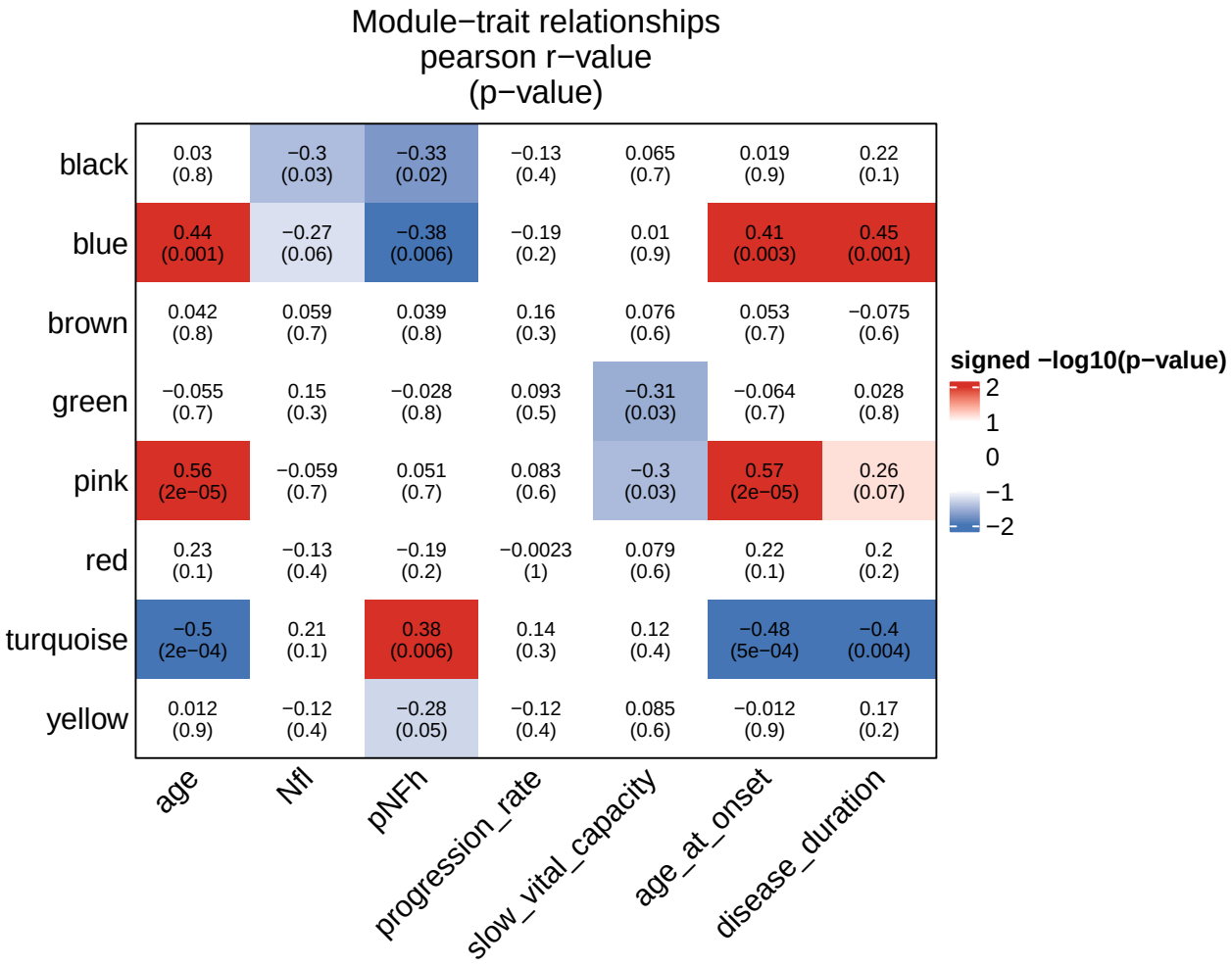

Figure 5: Figure 3e — Module–trait relationships (Discovery).

```r
  for (trait in clinical_vars) {
    df <- plot_df[, c(mod, trait)]
    df <- df[complete.cases(df), ]
    cor_test <- cor.test(df[[1]], df[[2]], method = "pearson")
    if (!is.na(cor_test$p.value) && cor_test$p.value < 0.1) {
      p <- ggplot(df, aes_string(x = trait, y = mod)) +
        geom_point(color = mod_color, size = 2.5, alpha = 1) +
        geom_smooth(method = "lm", se = TRUE, color = "black") +
        theme_classic(base_size = 12) +
        labs(x = trait, y = paste0("MEs (", mod_color, ")"), title = paste(mod_color,
          ↪  "vs", trait))
      plot_list[[paste(mod, trait, sep = "_")]] <- p
    }
  }
}
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
n_show <- min(max(12, length(plot_list)), length(plot_list))
if (n_show > 0) {
  do.call(gridExtra::grid.arrange, c(plot_list[seq_len(n_show)], ncol = 4))
  pdf(file.path(output_dir, "Fig3g.pdf"), width = 12, height = 10)
  do.call(gridExtra::grid.arrange, c(plot_list[seq_len(n_show)], ncol = 4))
  dev.off()
}
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```

```
## pdf
##   2
```

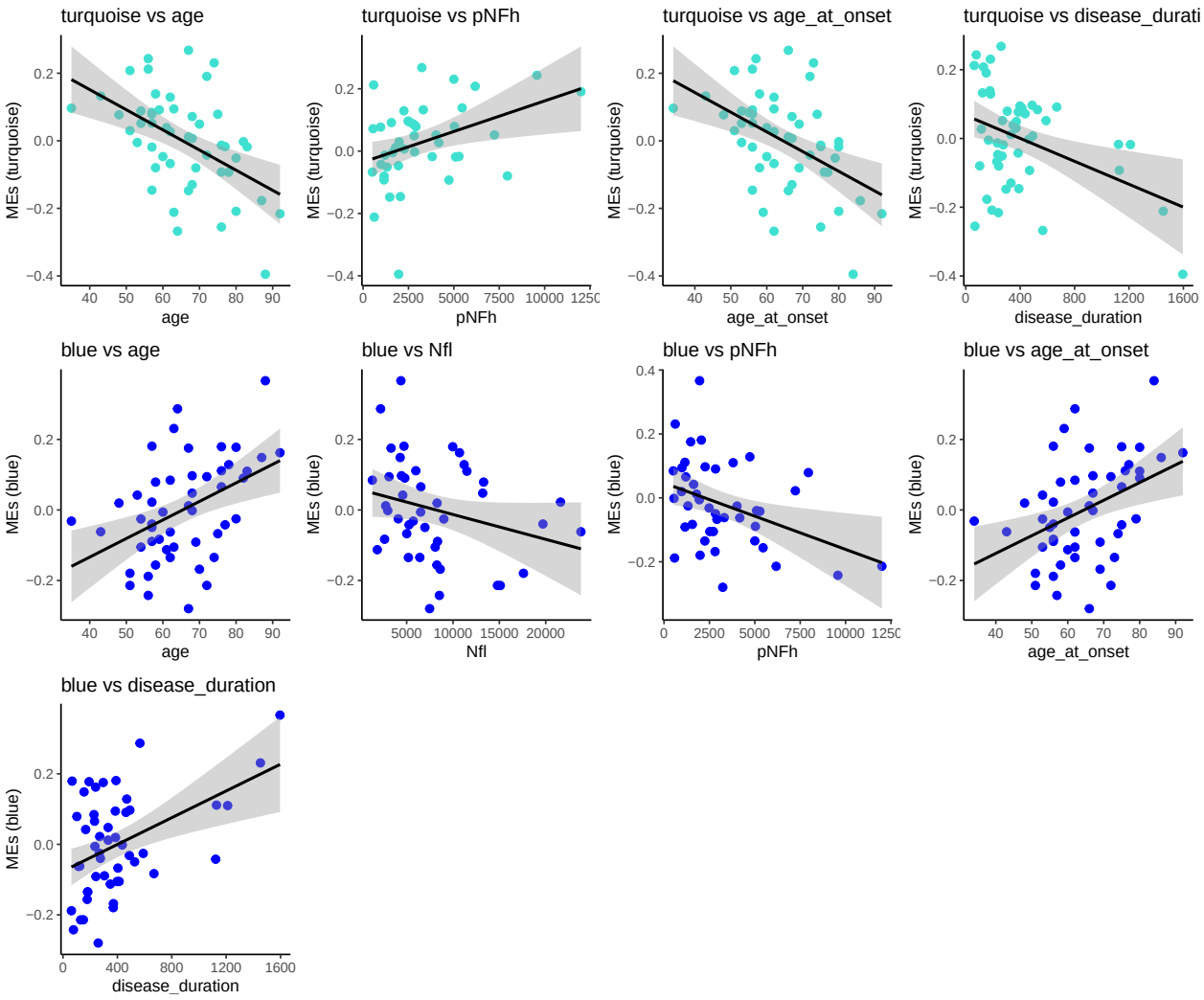

Figure 6: Figure 3g — MEs vs clinical variables.

Validation cohort: project Discovery modules and plot Fig 3d & 3f

```
if (!file.exists(validation_input)) {
  message("Validation input not found; skipping Fig 3d (Validation) and 3f.")
} else {
  cleanDat_file_val <- readRDS(validation_input)
  metaData_val <- as.data.frame(colData(cleanDat_file_val))
  cleanDat_val <- assay(cleanDat_file_val)
  colnames(cleanDat_val) <- metaData_val$label
  rownames(metaData_val) <- metaData_val$label

  common_proteins <- names(net$colors)[names(net$colors) %in% rownames(cleanDat_val)]
  net_inDC <- net$colors[common_proteins]
  cleanDat_val <- cleanDat_val[common_proteins, ]

  MEList_val <- moduleEigengenes(t(cleanDat_val), colors = net_inDC)
  MEs_val <- MEList_val$eigengenes
  MEs_val <- MEs_val[, colnames(MEs_val) != "MEgrey"]
  colnames(MEs_val) <- substr(colnames(MEs_val), 3, 100)
  MEs_val <- MEs_val[, setdiff(substr(colnames(net$MEs), 3, 100), "grey")]

  data_k2_val <- prepare_toplot(MEs_val, validation_cluster, metaData_val, "k2",
↪   cleanDat_val, ModuleTrait = TRUE)
  toplot_k2_val <- calculate_pvalues(MEs_val, data_k2_val$toplot, data_k2_val$factorMeta,
↪   "k2")

  if (sum(!is.na(metaData_val$pNFh)) > 20) {
    numericMeta_val <- metaData_val[, c("age", "Nfl", "pNFh", "progression_rate",
↪   "slow_vital_capacity", "age_at_onset", "disease_duration")]
  } else {
    numericMeta_val <- metaData_val[, c("age", "Nfl", "progression_rate",
↪   "slow_vital_capacity", "age_at_onset", "disease_duration")]
  }
  numericMeta_val <- numericMeta_val[match(colnames(cleanDat_val),
↪   rownames(numericMeta_val)), ]
  moduleTraitCor_val <- cor(MEs_val, numericMeta_val, use = "pairwise.complete.obs",
↪   method = "pearson")
  moduleTraitPvalue_val <- WGCNA::corPvalueStudent(moduleTraitCor_val, nrow(MEs_val))
  textMatrix_val <- paste(signif(moduleTraitCor_val, 2), "\n(",
↪   signif(moduleTraitPvalue_val, 1), ")", sep = "")
  dim(textMatrix_val) <- dim(moduleTraitCor_val)
  signed_logP_val <- sign(moduleTraitCor_val) * -log10(moduleTraitPvalue_val)
}
```

```
if (exists("toplot_k2_val")) {
  plot_boxplots(toplot_k2_val, data_k2_val$factorMeta, "k2", MEs_val, "Validation: ")
  pdf(file.path(output_dir, "Fig3d_Validation.pdf"), width = 10, height = 6)
  plot_boxplots(toplot_k2_val, data_k2_val$factorMeta, "k2", MEs_val, "Validation: ")
  dev.off()
}
```

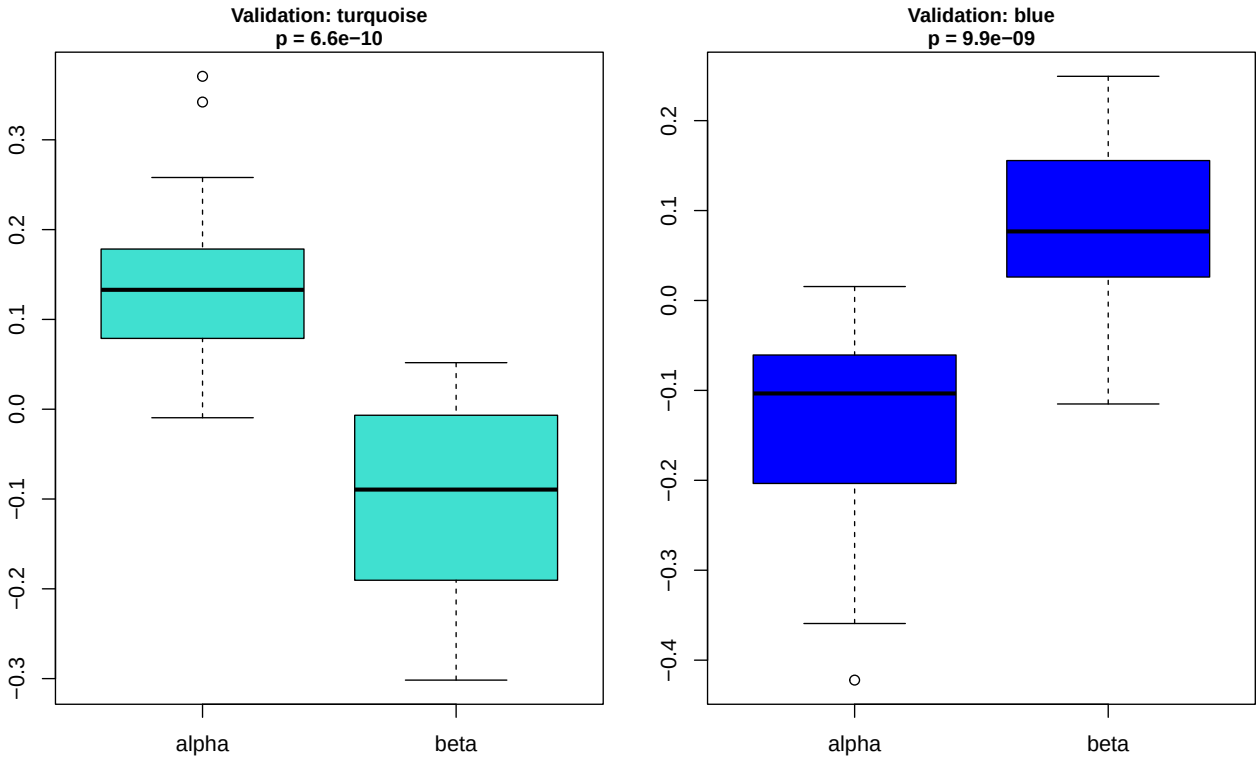

Figure 7: Figure 3d (Validation) — k2 eigenprotein boxplots.

```
## pdf
##   2
```

```
if (exists("signed_logP_val")) {
  ht_3f <- Heatmap(
    matrix = signed_logP_val,
    col = col_fun,
    cell_fun = function(j, i, x, y, w, h, col) {
      grid.text(textMatrix_val[i, j], x, y, gp = gpar(fontsize = 8))
    },
    row_names_side = "left", cluster_rows = FALSE, cluster_columns = FALSE,
    show_row_names = TRUE, show_column_names = TRUE,
    row_names_gp = gpar(fontsize = 12), column_names_gp = gpar(fontsize = 12),
    column_names_rot = 45,
    column_title = "Module-trait relationships (Validation)",
    border_gp = gpar(col = "black"),
    heatmap_legend_param = list(title = "signed -log10(p-value)", at = legend_breaks,
    ↪  labels = as.character(legend_breaks))
  )
  draw(ht_3f)
  pdf(file.path(output_dir, "Fig3f.pdf"), width = 7, height = 5.5)
  draw(ht_3f)
  dev.off()
}
```

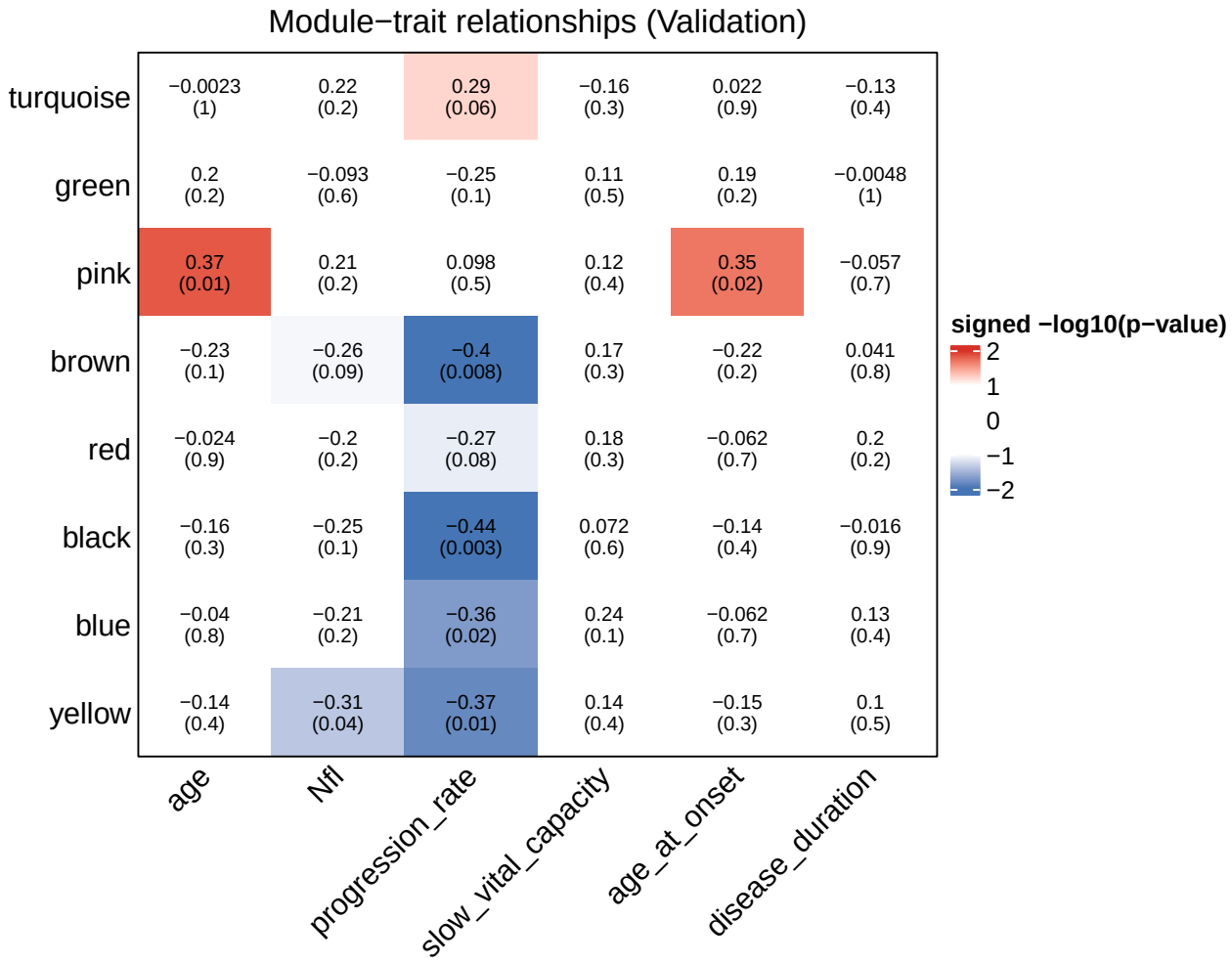

Figure 8: Figure 3f — Module–trait relationships (Validation).

```
## pdf
##   2
```

External cohort: project Discovery modules and plot Fig 3d

```
if (!file.exists(external_input)) {
  message("External input not found; skipping Fig 3d (External).")
} else {
  cleanDat_file_ext <- readRDS(external_input)
  metaData_ext <- as.data.frame(colData(cleanDat_file_ext))
  cleanDat_ext <- assay(cleanDat_file_ext)
  colnames(cleanDat_ext) <- metaData_ext$label
  rownames(metaData_ext) <- metaData_ext$label

  common_proteins_ext <- names(net$colors)[names(net$colors) %in% rownames(cleanDat_ext)]
  net_inDC_ext <- net$colors[common_proteins_ext]
  cleanDat_ext <- cleanDat_ext[common_proteins_ext, ]

  MEList_ext <- moduleEigengenes(t(cleanDat_ext), colors = net_inDC_ext)
  MEs_ext <- MEList_ext$eigengenes
  MEs_ext <- MEs_ext[, colnames(MEs_ext) != "MEgrey"]
  colnames(MEs_ext) <- substr(colnames(MEs_ext), 3, 100)
  MEs_ext <- MEs_ext[, setdiff(substr(colnames(net$MEs), 3, 100), "grey")]

  data_k2_ext <- prepare_toplot(MEs_ext, external_cluster, metaData_ext, "k2",
↪   cleanDat_ext, ModuleTrait = TRUE)
  toplot_k2_ext <- calculate_pvalues(MEs_ext, data_k2_ext$toplot, data_k2_ext$factorMeta,
↪   "k2")
}
```

```
if (exists("toplot_k2_ext")) {
  plot_boxplots(toplot_k2_ext, data_k2_ext$factorMeta, "k2", MEs_ext, "External: ")
  pdf(file.path(output_dir, "Fig3d_External.pdf"), width = 10, height = 6)
  plot_boxplots(toplot_k2_ext, data_k2_ext$factorMeta, "k2", MEs_ext, "External: ")
  dev.off()
}
```

```
## pdf
##   2
```

Outputs

All figures are saved as PDFs in **/Users/xliu2942/Documents/Projects/MAXOMOD/MAXOMOD_CSF/Plots/F** (default: `Plots/Fig3_WGCNA`):

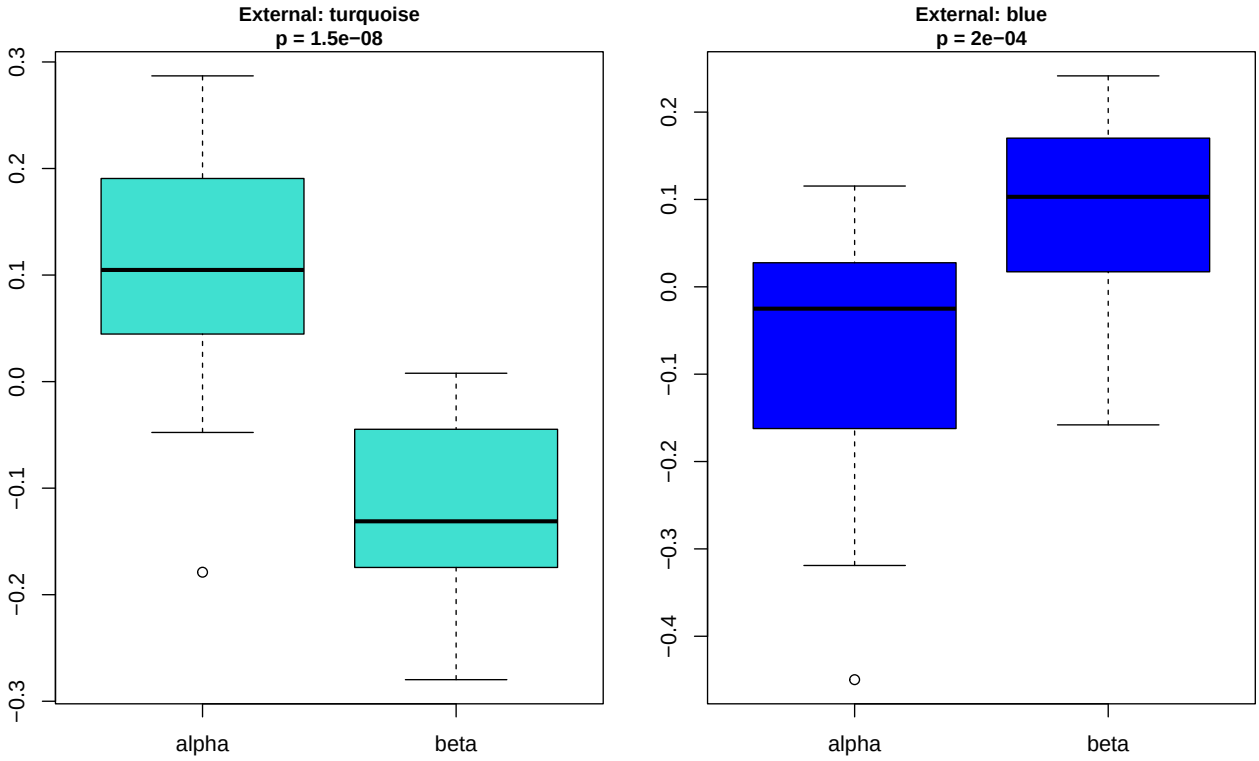

Figure 9: Figure 3d (External) — k2 eigenprotein boxplots.

- **Fig3a.pdf** — WGCNA dendrogram and module–trait correlations
- **Fig3b.pdf** — Eigengene networks
- **Fig3c.pdf** — Top GO terms per module (IC heatmap)
- **Fig3d_Discovery.pdf**, **Fig3d_Validation.pdf**, **Fig3d_External.pdf** — k2 eigenprotein boxplots (turquoise & blue)
- **Fig3e.pdf** — Module–trait heatmap (Discovery)
- **Fig3f.pdf** — Module–trait heatmap (Validation)
- **Fig3g.pdf** — MEs vs clinical variables (turquoise & blue)

Input paths are set via `params` (default: `demo/`). Ensure the demo input files exist (and optionally Validation/External for 3d and 3f).