Figure 2b, 2c — Clustering AIC and Sankey plot

MAXOMOD_CSF

2026-02-16

Contents

Overview

This document reproduces **Figure 2b** (AIC value across clustering methods and number of clusters) and **Figure 2c** (Sankey plot: k=2 to k=3 cluster flow) from the clustering step, and exports **cluster_assignments_2.csv**.

Equivalent command line:

```
Rscript 08_Clustering_subclusters.R -i
↪  Discovery/02_Missing_Inspection_subclusters/norm_imp_MinProb.rds -o
↪  Discovery/08_Clustering_als -e 9
```

Input is read from **demo/Discovery/02_Missing_Inspection_subclusters/** (ALS-only normalized/imputed data). Ensure the demo folder has been populated before running. Paths are relative to the **project root**. **Run the "Project root and parameters" chunk first** when running chunks interactively.

Setup

Load packages

```
library(dplyr)
library(ggplot2)
library(reshape2)
library(ggthemes)
library(ggalluvial)
library(tidyverse)
library(RColorBrewer)
library(ggpubr)
library(mclust)
library(cluster)
library(SummarizedExperiment)
```

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
```

```
input_path  <- file.path(project_root, params$input)
output_dir  <- file.path(project_root, params$output)
set.seed(params$seed)
reverse_labels <- isTRUE(params$reverse)
```

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

Clustering helper functions

Same logic as `08_Clustering_subclusters.R`: TWSS, AIC/BIC, and `perform_clustering` for hclust, mclust, kmeans, pam (k=2..10).

```
calc_SS <- function(df) sum(as.matrix(dist(df)^2)) / (2 * nrow(df))
calc_TWSS <- function(df, clusters) {
  k <- length(unique(clusters))
  sum(sapply(1:k, function(i) calc_SS(df[clusters == i, , drop = FALSE])))
}
BIC2 <- function(df, clusters) {
  m <- ncol(df)
  n <- nrow(df)
  k <- length(unique(clusters))
  D <- calc_TWSS(df, clusters)
  data.frame(AIC = D + 2 * m * k, BIC = D + log(n) * m * k)
}

perform_clustering <- function(assay_data, seed, clin_labels) {
  set.seed(seed)
  AIC_scores <- BIC_scores <- as.data.frame(matrix(NA, nrow = 4, ncol = 9))
  rownames(AIC_scores) <- rownames(BIC_scores) <- c("hclust", "mclust", "kmeans", "pam")
  colnames(AIC_scores) <- colnames(BIC_scores) <- 2:10
  cluster_assignments <- data.frame(patid = clin_labels)

  for (i in 2:10) {
    # hclust
    title <- paste0("hclust_k=", i)
    dist_mat <- dist(assay_data, method = "euclidean")
    cl <- hclust(dist_mat, method = "ward.D")
    cluster_assignments[, title] <- cutree(cl, k = i)
    AIC_scores["hclust", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 1]
    BIC_scores["hclust", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 2]

    # mclust
    title <- paste0("mclust_k=", i)
    cl <- Mclust(assay_data, G = i)
    cluster_assignments[, title] <- cl$classification
    AIC_scores["mclust", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 1]
    BIC_scores["mclust", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 2]

    # kmeans
    title <- paste0("kmeans_k=", i)
    cl <- kmeans(assay_data, centers = i, nstart = 25, iter.max = 50)
    cluster_assignments[, title] <- cl$cluster
    AIC_scores["kmeans", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 1]
    BIC_scores["kmeans", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 2]
```

```
    # pam
    title <- paste0("pam_k=", i)
    cl <- pam(assay_data, k = i)
    cluster_assignments[, title] <- cl$clustering
    AIC_scores["pam", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 1]
    BIC_scores["pam", as.character(i)] <- BIC2(assay_data, cluster_assignments[,
↪  title])[1, 2]
  }

  cluster_assignments[, 2:ncol(cluster_assignments)] <- cluster_assignments[,
↪  2:ncol(cluster_assignments)] - 1
  list(
    cluster_assignments = cluster_assignments,
    AIC_scores = AIC_scores,
    BIC_scores = BIC_scores
  )
}
```

Load data and run clustering

```
if (!file.exists(input_path)) {
  stop("Input not found: ", input_path,
       ". Populate demo/Discovery/02_Missing_Inspection_subclusters/ or run
       ↪  01_Pre_Processing.R (--subset als) and 02_Missing_Inspection.R first.")
}

se <- readRDS(input_path)
clin <- as.data.frame(se@colData)
clin$onset <- clin$condition
clin$onset <- as.factor(clin$onset)
clin$sex <- as.factor(clin$sex)

assay <- as.data.frame(assay(se))
assay <- assay[, clin$ID]
colnames(assay) <- clin$label
assay <- as.data.frame(t(assay))

cluster_result <- perform_clustering(assay, seed = params$seed, clin_labels = clin$label)
cluster_assignments <- cluster_result$cluster_assignments
AIC_scores <- cluster_result$AIC_scores
```

Build cluster_assignments_2 (kmeans k=2 and k=3 with alpha/beta/theta labels)

```
cluster_assignments_2 <- cluster_assignments[, c("patid", "kmeans_k=2", "kmeans_k=3")]

# k=2: larger cluster -> alpha, smaller -> beta
cluster_counts <- table(cluster_assignments_2$`kmeans_k=2`)
larger_cluster <- names(cluster_counts)[which.max(cluster_counts)]
smaller_cluster <- names(cluster_counts)[which.min(cluster_counts)]
cluster_assignments_2$`kmeans_k=2` <- as.factor(cluster_assignments_2$`kmeans_k=2`)
if (reverse_labels) {
  levels(cluster_assignments_2$`kmeans_k=2`) <-
  ↪  ifelse(levels(cluster_assignments_2$`kmeans_k=2`) == larger_cluster, "beta",
  ↪  "alpha")
} else {
  levels(cluster_assignments_2$`kmeans_k=2`) <-
  ↪  ifelse(levels(cluster_assignments_2$`kmeans_k=2`) == larger_cluster, "alpha",
  ↪  "beta")
}
cluster_assignments_2$`kmeans_k=2` <- ordered(cluster_assignments_2$`kmeans_k=2`, levels
↪  = c("alpha", "beta", "theta"))

# k=3: assign alpha/beta/theta by majority from k=2
adjusted_clusters <- cluster_assignments_2 %>%
  group_by(`kmeans_k=3`) %>%
  summarise(
    majority_alpha = sum(`kmeans_k=2` == "alpha"),
    majority_beta = sum(`kmeans_k=2` == "beta"),
    total = n()
  ) %>%
  mutate(
    alpha_ratio = majority_alpha / total,
    beta_ratio = majority_beta / total,
    alpha_beta = alpha_ratio - beta_ratio
  ) %>%
  mutate(
    kmeans_k3_adjusted = case_when(
      alpha_beta == max(alpha_beta) ~ "alpha",
      alpha_beta == min(alpha_beta) ~ "beta",
      TRUE ~ "theta"
    )
  )
adjusted_clusters <- adjusted_clusters[, c("kmeans_k=3", "kmeans_k3_adjusted")]
cluster_assignments_2 <- left_join(cluster_assignments_2, adjusted_clusters, by =
↪  "kmeans_k=3") %>% select(-`kmeans_k=3`)
colnames(cluster_assignments_2)[3] <- "kmeans_k=3"
cluster_assignments_2$`kmeans_k=3` <- as.factor(cluster_assignments_2$`kmeans_k=3`)
cluster_assignments_2$`kmeans_k=3` <- ordered(cluster_assignments_2$`kmeans_k=3`, levels
↪  = c("alpha", "beta", "theta"))
```

Figure 2b — AIC value by method and number of clusters

```
AIC_scores$method <- rownames(AIC_scores)
melt_aic <- reshape2::melt(AIC_scores, id.vars = "method")
melt_aic$variable <- as.numeric(as.character(melt_aic$variable))
```

```
p_b <- ggplot(melt_aic, aes(x = variable, y = value, group = method, colour = method)) +
  geom_line() +
  geom_point() +
  labs(title = "AIC", x = "number of clusters (k)", y = "value") +
  theme_few()
print(p_b)
```

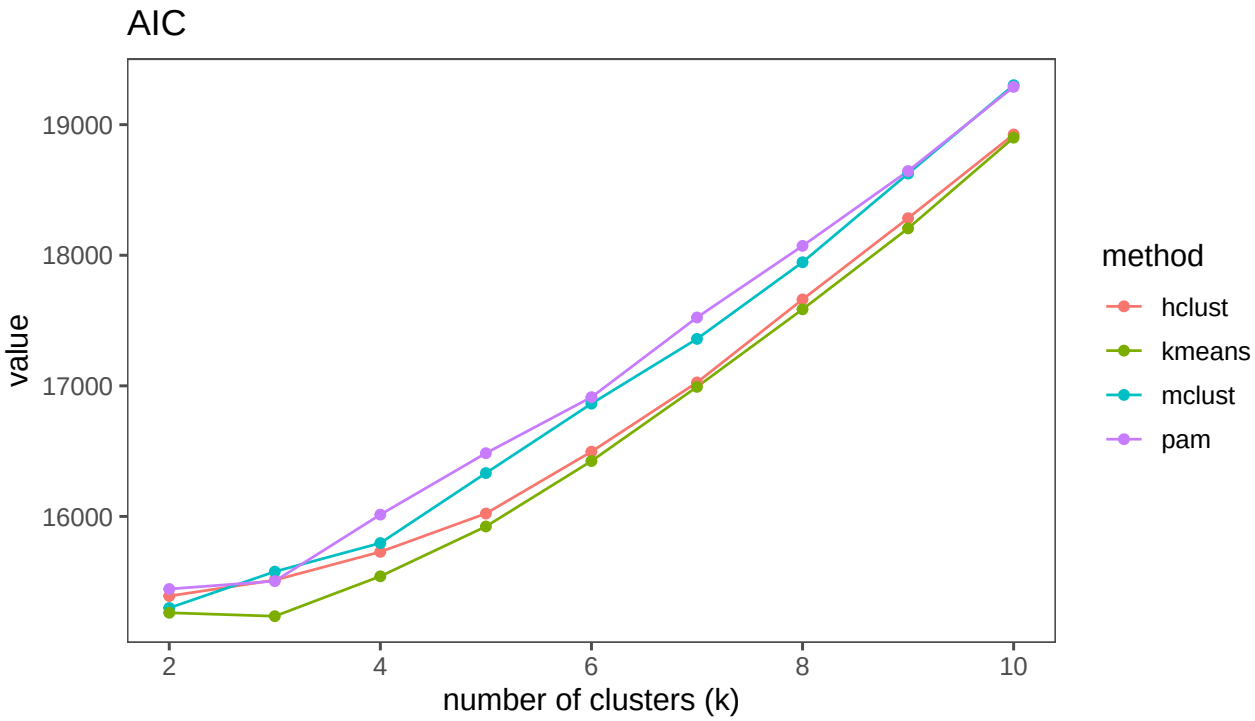

Figure 1: Figure 2b — AIC value for different clustering methods (hclust, kmeans, mclust, pam) and number of clusters (k=2..10).

```
ggsave(file.path(output_dir, "Fig2b_AIC.pdf"), p_b, width = 7, height = 4, dpi = 300)
ggsave(file.path(output_dir, "Fig2b_AIC.png"), p_b, width = 7, height = 4, dpi = 300)
```

Figure 2c — Sankey plot (k=2 to k=3)

```r
final_colours_clustering <- c(
  alpha = brewer.pal(8, "Set2")[2],
  beta  = brewer.pal(8, "Set2")[3],
  theta = brewer.pal(8, "Set2")[5]
)
data23 <- cluster_assignments_2 %>%
  pivot_longer(cols = c("kmeans_k=2", "kmeans_k=3"), names_to = "k", values_to =
  ↪  "cluster")
```

```r
p_c <- ggplot(data23, aes(x = k, stratum = cluster, alluvium = patid, fill = cluster,
↪  label = cluster)) +
  scale_fill_manual(values = final_colours_clustering) +
  geom_flow(stat = "alluvium", lode.guidance = "frontback", color = "darkgray") +
  geom_stratum() +
  theme(legend.position = "bottom") +
  ggtitle("K-means: 2 clusters to 3 clusters") +
  theme_few()
print(p_c)
```

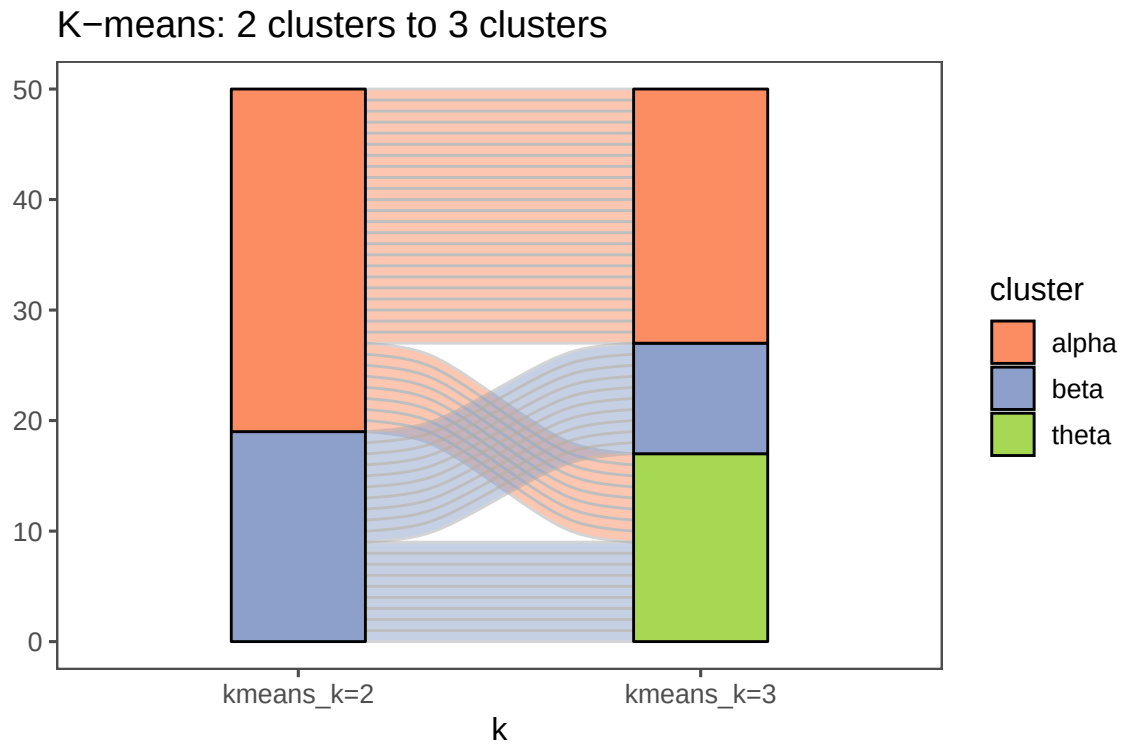

Figure 2: Figure 2c — Sankey plot: cluster assignment from k=2 to k=3 (k-means).

```r
ggsave(file.path(output_dir, "Fig2c_Sankey_k2_k3.pdf"), p_c, width = 6, height = 4, dpi =
↪  300)
ggsave(file.path(output_dir, "Fig2c_Sankey_k2_k3.png"), p_c, width = 6, height = 4, dpi =
↪  300)
```

Export cluster_assignments_2.csv

```
out_csv <- file.path(output_dir, "cluster_assignments_2.csv")
write.csv(cluster_assignments_2, out_csv, row.names = FALSE)
message("Exported: ", out_csv)
```

```
## Exported: /Users/xliu2942/Documents/Projects/MAXOMOD/MAXOMOD_CSF/Plots/Fig2bc_Clustering/cluster_ass
```

Outputs

- **Figure 2b:** `Fig2b_AIC.pdf` / `.png` in **/Users/xliu2942/Documents/Projects/MAXOMOD/MAXOMOD_CS**
- **Figure 2c:** `Fig2c_Sankey_k2_k3.pdf` / `.png` in **/Users/xliu2942/Documents/Projects/MAXOMOD/MAXO**
- **Table:** `cluster_assignments_2.csv` in **/Users/xliu2942/Documents/Projects/MAXOMOD/MAXOMOD_** (columns: patid, kmeans_k=2, kmeans_k=3 with levels alpha, beta, theta)

Ensure **demo/Discovery/02_Missing_Inspection_subclusters/norm_imp_MinProb.rds** exists (copy from pipeline output if needed) before knitting.