Untitled

JOKHROF AHMED DOHA

2026-02-27

#1

#2

```
rm(list = ls())
graphics.off()
```

#————————————-# # 3. LOAD PACKAGES #————————————-#

```
library(haven)       # read .sav / .dta
library(dplyr)       # data manipulation
library(tidyr)       # data reshaping
library(cluster)     # clustering algorithms
library(factoextra) # clustering visuals + gap statistic
library(e1071)       # fuzzy C-means
library(aricode)     # ARI + NMI
library(pheatmap)    # heatmaps
library(fmsb)        # radar charts
library(ggplot2)     # plots
```

==========

4.DISTRICT CODE → DISTRICT NAME LOOKUP

==========

```
district_lookup <- data.frame(
  HH7A = c(
    1,3,4,6,9,10,12,13,15,18,19,22,26,27,29,30,
    32,33,35,36,38,39,41,42,44,46,47,48,49,50,
    51,52,54,55,56,57,58,59,61,64,65,67,68,69,
    70,72,73,75,76,77,78,79,81,82,84,85,86,87,
    88,89,90,91,93,94
  ),
  district_name = c(
    "Bagerhat","Bandarban","Barguna","Barishal","Bhola","Bogura",
    "Brahmanbaria","Chandpur","Chattogram","Chuadanga","Cumilla",
    "Cox's Bazar","Dhaka","Dinajpur","Faridpur","Feni",
    "Gaibandha","Gazipur","Gopalganj","Habiganj","Joypurhat",
    "Jamalpur","Jashore","Jhalokati","Jhenaidah","Khagrachari",
    "Khulna","Kishoreganj","Kurigram","Kushtia","Lakshmipur",
    "Lalmonirhat","Madaripur","Magura","Manikganj","Meherpur",
```

```
    "Moulvibazar","Munshiganj","Mymensingh","Naogaon","Narail",
    "Narayanganj","Narsingdi","Natore","Chapainawabganj",
    "Netrokona","Nilphamari","Noakhali","Pabna","Panchagarh",
    "Patuakhali","Pirojpur","Rajshahi","Rajbari","Rangamati",
    "Rangpur","Shariatpur","Satkhira","Sirajganj","Sherpur",
    "Sunamganj","Sylhet","Tangail","Thakurgaon"
  )
)
```

#5

```
hh <- read_sav("hh.sav")
hl <- read_sav("hl.sav")
wm <- read_sav("wm.sav")
ch <- read_sav("ch.sav")
fs <- read_sav("fs.sav")  # if available
```

6.Creating Unique Household ID

```
hh <- hh %>% mutate(hh_id = paste(HH1, HH2, sep = "_"))
hl <- hl %>% mutate(hh_id = paste(HH1, HH2, sep = "_"))
wm <- wm %>% mutate(hh_id = paste(HH1, HH2, sep = "_"))
ch <- ch %>% mutate(hh_id = paste(HH1, HH2, sep = "_"))
fs <- fs %>% mutate(hh_id = paste(HH1, HH2, sep = "_"))
```

```
# ============================================================
# 7) HOUSEHOLD SIZE FROM HL
# ============================================================
hh_size_df <- hl %>%
  dplyr::group_by(hh_id) %>%
  dplyr::summarise(hh_size = dplyr::n(), .groups = "drop")

# ============================================================
# 8) MERGE HH WITH HOUSEHOLD SIZE
# ============================================================
hh <- hh %>%
  dplyr::left_join(hh_size_df, by = "hh_id")

# ============================================================
# 9) QUICK CHECKS (WASH ONLY)
# ============================================================
table(hh$WS1,  useNA = "ifany")
```

```
##
##    11    12    13    14    21    31    32    41    42    51    71    72    81
##  2121  2273   205   739 53685   171   499     8   132   333    50    84   764
##    91    92    96  <NA>
##   163     5    10  3158
```

```
table(hh$WS3,  useNA = "ifany")
```

```
##
##     1     2     3     9  <NA>
##  2517 41518 12604     2  7759
```

```r
table(hh$WS7,  useNA = "ifany")
```

```
##
##     1     2     8     9  <NA>
##  2023 59174    33    12  3158
```

```r
table(hh$WS11, useNA = "ifany")
```

```
##
##    11    12    13    14    18    21    22    23    31    51    95    96    99
##  2580 12642 10222  1470    64   652 24890  6083    32  1402  1186    16     3
##  <NA>
##  3158
```

```r
table(hh$WS15, useNA = "ifany")
```

```
##
##     1     2     9  <NA>
## 15692 44347    17  4344
```

```r
table(hh$HW1,  useNA = "ifany")
```

```
##
##     1     2     3     4     5     6  <NA>
##  8893 36263  6415  9585    23    63  3158
```

```r
table(hh$HW2,  useNA = "ifany")
```

```
##
##     1     2     9  <NA>
## 49271  2286    14 12829
```

```r
table(hh$HW3,  useNA = "ifany")
```

```
##
##     1     2     9  <NA>
## 34865 16699     7 12829
```

```r
# ============================================================
# 10) RECODE WASH INDICATORS (HH LEVEL)
#     + COMBINED WASH (water & sanitation & hygiene)
# ============================================================
hh <- hh %>%
  dplyr::mutate(
    # ----------------------
    # WATER (your definition: reliable on-premises improved water)
    # ----------------------
    improved_water_source = dplyr::case_when(
      WS1 %in% c(11,12,13,14,21,31,41,51,91,92) ~ 1,
      WS1 %in% c(32,42,61,71,72,81,96) ~ 0,
      WS1 %in% c(98,99) ~ NA_real_,   # if 98 exists, treat as missing/DK
      TRUE ~ NA_real_
    ),
    water_on_prem = dplyr::case_when(
      WS3 %in% c(1,2) ~ 1,
      WS3 == 3 ~ 0,
      WS3 %in% c(8,9) ~ NA_real_,
      TRUE ~ NA_real_
```

```
  ),
  water_sufficient = dplyr::case_when(
    WS7 == 2 ~ 1,           # always sufficient
    WS7 == 1 ~ 0,           # at least once insufficient
    WS7 %in% c(8,9) ~ NA_real_,
    TRUE ~ NA_real_
  ),
  improved_water = dplyr::case_when(
    improved_water_source == 1 & water_on_prem == 1 & water_sufficient == 1 ~ 1,
    improved_water_source == 0 ~ 0,
    improved_water_source == 1 & (water_on_prem == 0 | water_sufficient == 0) ~ 0,
    TRUE ~ NA_real_
  ),

  # ----------------------
  # SANITATION (basic: improved AND not shared)
  # ----------------------
  improved_san_facility = dplyr::case_when(
    WS11 %in% c(11,12,13,21,22,31) ~ 1,                 # improved
    WS11 %in% c(14,18,23,41,51,95,96) ~ 0,              # unimproved
    WS11 %in% c(98,99) ~ NA_real_,
    TRUE ~ NA_real_
  ),
  san_not_shared = dplyr::case_when(
    WS15 == 2 ~ 1,   # NOT shared
    WS15 == 1 ~ 0,   # shared
    WS15 %in% c(8,9) ~ NA_real_,
    TRUE ~ NA_real_
  ),
  improved_san = dplyr::case_when(
    improved_san_facility == 1 & san_not_shared == 1 ~ 1,
    improved_san_facility == 0 ~ 0,
    improved_san_facility == 1 & san_not_shared == 0 ~ 0,
    TRUE ~ NA_real_
  ),

  # ----------------------
  # HYGIENE (basic: place observed + water + soap)
  # ----------------------
  hw_place_observed = dplyr::case_when(
    HW1 %in% c(1,2,3) ~ 1,
    HW1 %in% c(4,5,6) ~ 0,
    HW1 %in% c(8,9) ~ NA_real_,
    TRUE ~ NA_real_
  ),
  hw_water = dplyr::case_when(
    HW2 == 1 ~ 1,
    HW2 == 2 ~ 0,
    HW2 %in% c(8,9) ~ NA_real_,
    TRUE ~ NA_real_
  ),
  hw_soap = dplyr::case_when(
    HW3 == 1 ~ 1,
```

```
      HW3 == 2 ~ 0,
      HW3 %in% c(8,9) ~ NA_real_,
      TRUE ~ NA_real_
    ),
    handwashing_basic = dplyr::case_when(
      hw_place_observed == 1 & hw_water == 1 & hw_soap == 1 ~ 1,
      hw_place_observed == 0 | hw_water == 0 | hw_soap == 0 ~ 0,
      TRUE ~ NA_real_
    ),

    # ----------------------
    # COMBINED WASH (PLOS concept):
    # 1 if household has ALL THREE: water + sanitation + hygiene
    # ----------------------
    wash_combined = dplyr::case_when(
      improved_water == 1 & improved_san == 1 & handwashing_basic == 1 ~ 1,
      improved_water == 0 | improved_san == 0 | handwashing_basic == 0 ~ 0,
      TRUE ~ NA_real_
    ),

    # weight
    w_hh = as.numeric(hhweight)
  )

# ============================================================
# 11) WEIGHTED MEAN FUNCTION (safe)
# ============================================================
wmean <- function(x, w) {
  ok <- is.finite(x) & is.finite(w)
  if (!any(ok)) return(NA_real_)
  sum(x[ok] * w[ok]) / sum(w[ok])
}

# ============================================================
# 12) DISTRICT AGGREGATION (WASH ONLY + COMBINED)
# ============================================================
district_wash <- hh %>%
  dplyr::filter(is.finite(HH7A)) %>%
  dplyr::group_by(HH7A) %>%
  dplyr::summarise(
    improved_water_pct    = 100 * wmean(improved_water, w_hh),
    improved_san_pct      = 100 * wmean(improved_san, w_hh),
    handwashing_basic_pct = 100 * wmean(handwashing_basic, w_hh),
    wash_combined_pct     = 100 * wmean(wash_combined, w_hh),
    n_hh_unw              = dplyr::n(),
    n_hh_wt               = sum(w_hh[is.finite(w_hh)], na.rm = TRUE),
    .groups = "drop"
  )

# ============================================================
# 13) MERGE LOOKUP + CLUSTER MATRIX (include combined if you want)
# ============================================================
district <- district_wash %>%
```

```
  dplyr::left_join(district_lookup, by = "HH7A") %>%
  dplyr::rename(district_code = HH7A)

wash_mat <- district %>%
  dplyr::select(
    improved_water_pct,
    improved_san_pct,
    handwashing_basic_pct,
    wash_combined_pct
  )
```

#————————————-# # 16. PREPARE MATRIX FOR CLUSTERING #————————————-#

```
wash_mat <- district %>%
  dplyr::select(
    improved_water_pct,
    improved_san_pct,
    handwashing_basic_pct,
    wash_combined_pct
 # included only if present
  )

# Replace NA using median (robust) and scale
wash_scaled <- wash_mat %>%
  mutate(across(everything(), ~ replace(., is.na(.), median(., na.rm = TRUE)))) %>%
  scale()
```

==========

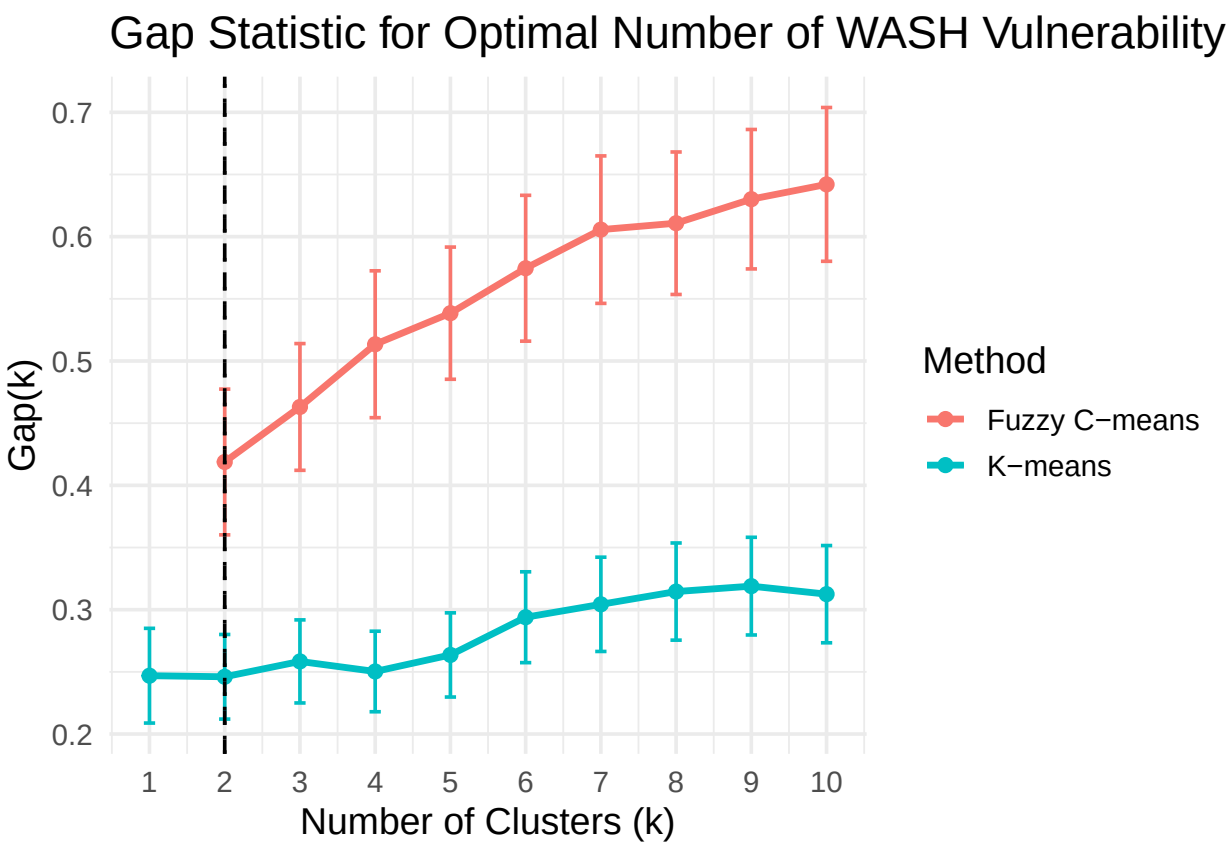
18. GAP STATISTIC CODE (K-MEANS + FUZZY C-MEANS)

==========

```
library(cluster)
library(e1071)
library(ggplot2)
library(dplyr)

set.seed(42)

# ---- Safety checks ----
wash_scaled <- as.matrix(wash_scaled)
stopifnot(all(is.finite(wash_scaled)))

K.max <- 10
B <- 100
m_val <- 2

# ------------------------------------------------------------
# Helper: 1-SE rule (FORCE k >= 2)
# ------------------------------------------------------------
gap_1se_k_from2 <- function(k, gap, se) {
```

```
  ord <- order(k)
  k <- k[ord]; gap <- gap[ord]; se <- se[ord]

  idx <- which(k >= 2)
  k <- k[idx]; gap <- gap[idx]; se <- se[idx]

  for (i in seq_along(k)[-length(k)]) {
    if (gap[i] >= gap[i + 1] - se[i + 1]) return(k[i])
  }
  k[which.max(gap)]
}

# ============================================================
# GAP STATISTIC - K-MEANS
# ============================================================
gap_km <- clusGap(
  wash_scaled,
  FUN = kmeans,
  nstart = 25,
  K.max = K.max,
  B = B
)

gap_km_df <- as.data.frame(gap_km$Tab) %>%
  mutate(
    k = seq_len(n()),
    method = "K-means"
  ) %>%
  transmute(k, gap = gap, SE.sim = SE.sim, method)

k_km_opt <- gap_1se_k_from2(
  gap_km_df$k, gap_km_df$gap, gap_km_df$SE.sim
)

# ============================================================
# GAP STATISTIC - FUZZY C-MEANS (MANUAL)
# ============================================================
mins <- apply(wash_scaled, 2, min)
maxs <- apply(wash_scaled, 2, max)
n <- nrow(wash_scaled)
p <- ncol(wash_scaled)

fuzzy_gap <- rep(NA_real_, K.max)
fuzzy_se  <- rep(NA_real_, K.max)

for (k in 2:K.max) {

  fcm_real <- try(cmeans(wash_scaled, centers = k, m = m_val), silent = TRUE)
  if (inherits(fcm_real, "try-error")) next
  Wk_real <- fcm_real$withinerror

  Wk_ref <- numeric(B)
```

```
  for (b in 1:B) {
    ref_data <- matrix(NA_real_, n, p)
    for (j in 1:p) ref_data[, j] <- runif(n, mins[j], maxs[j])

    fcm_ref <- try(cmeans(ref_data, centers = k, m = m_val), silent = TRUE)
    if (!inherits(fcm_ref, "try-error")) Wk_ref[b] <- fcm_ref$withinerror
  }

  Wk_ref <- Wk_ref[is.finite(Wk_ref)]

  if (length(Wk_ref) >= 10) {
    fuzzy_gap[k] <- mean(log(Wk_ref)) - log(Wk_real)
    fuzzy_se[k]  <- sd(log(Wk_ref)) * sqrt(1 + 1 / length(Wk_ref))
  }
}

gap_fcm_df <- data.frame(
  k = 1:K.max,
  gap = fuzzy_gap,
  SE.sim = fuzzy_se,
  method = "Fuzzy C-means"
) %>%
  filter(!is.na(gap))

k_fcm_opt <- gap_1se_k_from2(
  gap_fcm_df$k, gap_fcm_df$gap, gap_fcm_df$SE.sim
)

# ============================================================
# FINAL GAP PLOT
# ============================================================
gap_all <- bind_rows(gap_km_df, gap_fcm_df)

p_gap <- ggplot(gap_all, aes(x = k, y = gap, color = method)) +
  geom_line(linewidth = 1.2) +
  geom_point(size = 2) +
  geom_errorbar(aes(ymin = gap - SE.sim, ymax = gap + SE.sim), width = 0.15) +
  geom_vline(xintercept = k_km_opt, linetype = "dashed") +
  geom_vline(xintercept = k_fcm_opt, linetype = "dotted") +
  scale_x_continuous(breaks = 1:K.max) +
  labs(
    title = "Gap Statistic for Optimal Number of WASH Vulnerability Clusters",
    x = "Number of Clusters (k)",
    y = "Gap(k)",
    color = "Method"
  ) +
  theme_minimal(base_size = 14)

print(p_gap)
```




```
ggsave("Gap_stat_ch_4.2.pdf",
       p_gap, width = 10, height = 6, dpi = 600)

cat("Optimal k (K-means, 1-SE rule):", k_km_opt, "\n")
```

```
## Optimal k (K-means, 1-SE rule): 2
```

```
cat("Optimal k (Fuzzy C-means, 1-SE rule):", k_fcm_opt, "\n")
```

```
## Optimal k (Fuzzy C-means, 1-SE rule): 2
```

19. Table X: Gap Statistics and Standard Errors for WASH Indicators

```
gap_table <- gap_all %>%
  dplyr::filter(!is.na(gap)) %>%
  dplyr::arrange(method, k) %>%
  dplyr::transmute(
    Method = method,
    k = k,
    Gap_k = gap,
    SE_k = SE.sim
  )

# Show in RMD
knitr::kable(
  gap_table,
  digits = 4,
  caption = "Gap statistics and associated standard errors for WASH indicators using K-means and Fuzzy C
```

```
)
```

Table 1: Gap statistics and associated standard errors for WASH indicators using K-means and Fuzzy C-means

Method	k	Gap_k	SE_k
Fuzzy C-means	2	0.4188	0.0586
Fuzzy C-means	3	0.4630	0.0510
Fuzzy C-means	4	0.5135	0.0591
Fuzzy C-means	5	0.5385	0.0531
Fuzzy C-means	6	0.5746	0.0586
Fuzzy C-means	7	0.6057	0.0593
Fuzzy C-means	8	0.6108	0.0573
Fuzzy C-means	9	0.6301	0.0561
Fuzzy C-means	10	0.6420	0.0618
K-means	1	0.2469	0.0381
K-means	2	0.2460	0.0340
K-means	3	0.2584	0.0334
K-means	4	0.2503	0.0324
K-means	5	0.2636	0.0339
K-means	6	0.2939	0.0365
K-means	7	0.3043	0.0379
K-means	8	0.3146	0.0391
K-means	9	0.3189	0.0393
K-means	10	0.3125	0.0391

```
# Save as CSV
write.csv(
  gap_table,
  "Table_Gap_Statistic_WASH.csv",
  row.names = FALSE
)
```

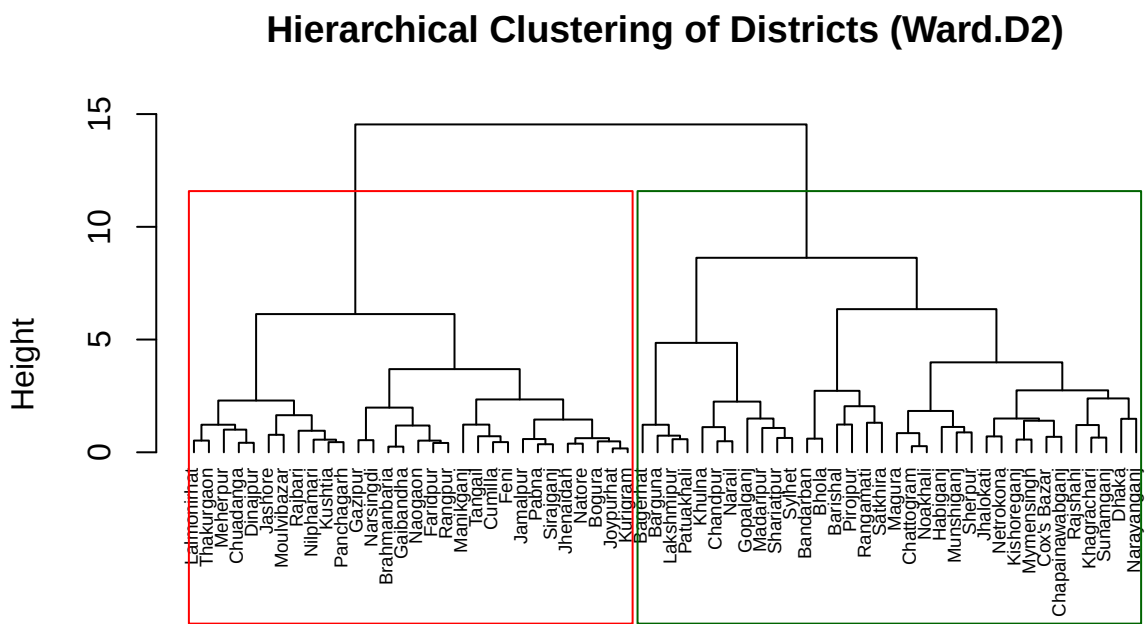
#————————————————- # 21. HIERARCHICAL CLUSTERING #————————————————-

```
# Distance matrix
dist_mat <- dist(wash_scaled)

# Ward's hierarchical clustering
hc <- hclust(dist_mat, method = "ward.D2")
best_k <- 2

# Cut using optimal k from NbClust
hc_cut <- cutree(hc, k = best_k)
table(hc_cut)
```

```
## hc_cut
##  1  2
## 34 30
```

```
# Attach cluster labels
district$cluster_hc <- factor(hc_cut)

# ---- CLEAN DENDROGRAM PLOT ----
```

```
par(mar = c(5, 4, 4, 2))   # better margins

#pdf("Figure_4.3.1_Dendrogram_WardD2.pdf", width = 12, height = 7)
plot_hier <- plot(
  hc,
  labels = district$district_name,  # USE NAMES instead of numeric IDs
  cex = 0.6,                         # shrink label size
  hang = -1,                        # align labels at same level
  main = "Hierarchical Clustering of Districts (Ward.D2)",
  xlab = "",
  sub  = "",
  ylab = "Height"
)
rect.hclust(hc, k = best_k, border = c("red", "darkgreen"))
```




```
#dev.off()


district_cluster_tbl <- district %>%
  dplyr::select(district_code, district_name, cluster_hc) %>%
  dplyr::arrange(cluster_hc, district_name)

knitr::kable(
  district_cluster_tbl,
  digits = 4,
  caption = "District-to-cluster assignment (Ward's hierarchical clustering)"
)
```

Table 2: District-to-cluster assignment (Ward's hierarchical clustering)

district_code	district_name	cluster_hc
1	Bagerhat	1

district_code	district_name	cluster_hc
3	Bandarban	1
4	Barguna	1
6	Barishal	1
9	Bhola	1
13	Chandpur	1
70	Chapainawabganj	1
15	Chattogram	1
22	Cox's Bazar	1
26	Dhaka	1
35	Gopalganj	1
36	Habiganj	1
42	Jhalokati	1
46	Khagrachari	1
47	Khulna	1
48	Kishoreganj	1
51	Lakshmipur	1
54	Madaripur	1
55	Magura	1
59	Munshiganj	1
61	Mymensingh	1
65	Narail	1
67	Narayanganj	1
72	Netrokona	1
75	Noakhali	1
78	Patuakhali	1
79	Pirojpur	1
81	Rajshahi	1
84	Rangamati	1
87	Satkhira	1
86	Shariatpur	1
89	Sherpur	1
90	Sunamganj	1
91	Sylhet	1
10	Bogura	2
12	Brahmanbaria	2
18	Chuadanga	2
19	Cumilla	2
27	Dinajpur	2
29	Faridpur	2
30	Feni	2
32	Gaibandha	2
33	Gazipur	2
39	Jamalpur	2
41	Jashore	2
44	Jhenaidah	2
38	Joypurhat	2
49	Kurigram	2
50	Kushtia	2
52	Lalmonirhat	2
56	Manikganj	2
57	Meherpur	2
58	Moulvibazar	2

district_code	district_name	cluster_hc
64	Naogaon	2
68	Narsingdi	2
69	Natore	2
73	Nilphamari	2
76	Pabna	2
77	Panchagarh	2
82	Rajbari	2
85	Rangpur	2
88	Sirajganj	2
93	Tangail	2
94	Thakurgaon	2

#————————————-# # 22. K-MEANS CLUSTERING #————————————-#

```
library(factoextra)
library(ggplot2)

set.seed(42)

# Ensure wash_scaled is numeric matrix for kmeans
wash_scaled_mat <- as.matrix(wash_scaled)

# Run K-means
km <- kmeans(wash_scaled_mat, centers = best_k, nstart = 25)
km
```

```
## K-means clustering with 2 clusters of sizes 29, 35
##
## Cluster means:
##   improved_water_pct improved_san_pct handwashing_basic_pct wash_combined_pct
## 1         -0.7997884       -0.1356114            -0.8350625        -0.8693599
## 2          0.6626818        0.1123637             0.6919089         0.7203268
##
## Clustering vector:
##  [1] 1 1 1 1 1 2 2 1 1 2 2 1 1 2 2 2 2 2 1 1 2 2 2 1 2 1 1 1 2 2 1 2 2 2 2 2 2 1
## [39] 1 2 1 2 2 2 1 1 2 1 2 2 1 1 2 2 1 2 1 1 2 1 1 2 2 2
##
## Within cluster sum of squares by cluster:
## [1] 82.03647 58.01132
##  (between_SS / total_SS =  44.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# Attach labels
district$cluster_km <- factor(km$cluster)

# For fviz_cluster, use a numeric data.frame (NOT matrix)
wash_scaled_df <- as.data.frame(wash_scaled_mat)

pdf("Figure_4.3.2_pca_kmeans.pdf", width = 12, height = 7)
```

```r
# Plot
fviz_cluster(
  object = km,
  data   = wash_scaled_df,
  geom   = "point",
  repel  = TRUE,
  main   = "K-means Clusters of Districts (WASH Indicators)"
) + theme_minimal(base_size = 14)

dev.off()
```

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

#——————————————-# # 23. FUZZY C-MEANS CLUSTERING #——————————————-#

```r
suppressPackageStartupMessages({
  library(e1071)   # cmeans()
  library(dplyr)
  library(knitr)
})

set.seed(42)

# ---- 0) SAFETY CHECKS ----
if (!exists("best_k")) best_k <- 2
stopifnot(is.numeric(best_k) && best_k >= 2)

wash_scaled <- as.matrix(wash_scaled)
stopifnot(all(is.finite(wash_scaled)))
stopifnot(exists("district"))
stopifnot(nrow(wash_scaled) == nrow(district))

cat("\n==============================\n")
```

```
##
## ==============================
```

```r
cat("FUZZY C-MEANS (FCM) RUN SUMMARY\n")
```

```
## FUZZY C-MEANS (FCM) RUN SUMMARY
```

```r
cat("==============================\n")
```

```
## ==============================
```

```r
cat("k (best_k) =", best_k, "\n")
```

```
## k (best_k) = 2
```

```r
# ---- 1) RUN / REUSE FCM SAFELY ----
m_val <- 2

# NEVER trust object named 'fcm' (it may be overwritten)
need_refit <- TRUE

if (exists("fcm_model") && is.list(fcm_model) && !is.null(fcm_model$membership)) {
  need_refit <- FALSE
```

```
}

# If no fcm_model, try to reuse 'fcm' ONLY if it looks like a model object
if (need_refit && exists("fcm") && is.list(fcm) && !is.null(fcm$membership)) {
  fcm_model <- fcm
  need_refit <- FALSE
}

# Otherwise refit
if (need_refit) {
  fcm_model <- e1071::cmeans(wash_scaled, centers = best_k, m = m_val)
}

U <- as.matrix(fcm_model$membership)  # n x k membership matrix
stopifnot(ncol(U) == best_k)

# ---- 2) SANITY CHECK: MEMBERSHIP ROWS SUM TO 1 ----
rs <- rowSums(U)
cat("\nMembership row-sum check (should be ~1):\n")
```

```
##
## Membership row-sum check (should be ~1):
```

```
print(summary(rs))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1       1       1       1       1       1
```

```
stopifnot(max(abs(rs - 1)) < 1e-6)

# ---- 3) CRISP LABELS FOR COMPARISON / MAPS ----
district$cluster_fcm <- factor(apply(U, 1, which.max))

cat("\nFCM crisp cluster sizes:\n")
```

```
##
## FCM crisp cluster sizes:
```

```
print(table(district$cluster_fcm))
```

```
##
##  1  2
## 28 36
```

```
# ---- 4) CERTAINTY + AMBIGUITY MEASURES ----
district$max_membership <- apply(U, 1, max)

# gap between top-1 and top-2 membership (works for any k)
district$top2_gap <- apply(U, 1, function(u) {
  s <- sort(u, decreasing = TRUE)
  s[1] - s[2]
})

# Special for k=2 only: |u1 - u2|
district$membership_gap_k2 <- if (best_k == 2) abs(U[, 1] - U[, 2]) else NA_real_
```

```r
cat("\nMax-membership (certainty) summary:\n")
```

```
##
## Max-membership (certainty) summary:
```

```r
print(summary(district$max_membership))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.5122  0.7166  0.8235  0.7979  0.8978  0.9954
```

```r
cat("\nTop-2 membership gap summary (smaller = more ambiguous):\n")
```

```
##
## Top-2 membership gap summary (smaller = more ambiguous):
```

```r
print(summary(district$top2_gap))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.02446 0.43325 0.64696 0.59572 0.79550 0.99089
```

```r
if (best_k == 2) {
  cat("\nMembership gap (k=2) summary (|u1-u2|):\n")
  print(summary(district$membership_gap_k2))
}
```

```
##
## Membership gap (k=2) summary (|u1-u2|):
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.02446 0.43325 0.64696 0.59572 0.79550 0.99089
```

```r
# ---- 5) TRANSITIONAL DISTRICTS: SENSITIVITY TO THRESHOLDS ----
thresholds <- c(0.05, 0.10, 0.15)

cat("\nTransitional district counts (sensitivity thresholds):\n")
```

```
##
## Transitional district counts (sensitivity thresholds):
```

```r
if (best_k == 2) {
  for (thr in thresholds) {
    nm <- paste0("transition_fcm_k2_thr", gsub("\\.", "", sprintf("%.2f", thr)))
    district[[nm]] <- as.integer(district$membership_gap_k2 < thr)
    cat("  threshold <", thr, ": ", sum(district[[nm]], na.rm = TRUE), " districts\n", sep = "")
  }
} else {
  for (thr in thresholds) {
    nm <- paste0("transition_fcm_top2_thr", gsub("\\.", "", sprintf("%.2f", thr)))
    district[[nm]] <- as.integer(district$top2_gap < thr)
    cat("  threshold <", thr, ": ", sum(district[[nm]], na.rm = TRUE), " districts\n", sep = "")
  }
}
```

```
##   threshold <0.05: 3 districts
##   threshold <0.1: 3 districts
##   threshold <0.15: 4 districts
```

```r
# ---- 6) LIST MOST AMBIGUOUS DISTRICTS (TOP 15) ----
ambiguity_var <- if (best_k == 2) "membership_gap_k2" else "top2_gap"
```

```r
cat("\nMost ambiguous districts (TOP 15; smallest gap = most ambiguous):\n")
```

```
##
## Most ambiguous districts (TOP 15; smallest gap = most ambiguous):
```

```r
amb_tbl <- district %>%
  mutate(.gap = .data[[ambiguity_var]]) %>%
  arrange(.gap) %>%
  select(district_name, cluster_fcm, max_membership, .gap) %>%
  head(15)

print(amb_tbl)
```

```
## # A tibble: 15 x 4
##    district_name cluster_fcm max_membership   .gap
##    <chr>         <fct>                <dbl>  <dbl>
##  1 Shariatpur    2                    0.512 0.0245
##  2 Narayanganj   2                    0.521 0.0412
##  3 Rajshahi      2                    0.524 0.0478
##  4 Gopalganj     1                    0.569 0.139
##  5 Gazipur       2                    0.594 0.187
##  6 Chattogram    1                    0.600 0.200
##  7 Mymensingh    1                    0.606 0.213
##  8 Munshiganj    1                    0.629 0.258
##  9 Narail        1                    0.632 0.264
## 10 Sherpur       1                    0.647 0.294
## 11 Manikganj     2                    0.673 0.346
## 12 Madaripur     2                    0.676 0.353
## 13 Narsingdi     2                    0.688 0.377
## 14 Noakhali      1                    0.700 0.400
## 15 Chandpur      1                    0.703 0.407
```

```r
# ---- 7) MEMBERSHIP TABLE PREVIEW ----
cat("\nMembership matrix preview (first 10 districts):\n")
```

```
##
## Membership matrix preview (first 10 districts):
```

```r
U_preview <- as.data.frame(round(U, 4))
colnames(U_preview) <- paste0("u", seq_len(ncol(U_preview)))
U_preview$district_name <- district$district_name
U_preview <- U_preview %>% select(district_name, everything()) %>% head(10)
print(U_preview)
```

```
##    district_name     u1     u2
## 1       Bagerhat 0.7394 0.2606
## 2      Bandarban 0.7944 0.2056
## 3        Barguna 0.7588 0.2412
## 4       Barishal 0.8864 0.1136
## 5          Bhola 0.8225 0.1775
## 6         Bogura 0.0499 0.9501
## 7   Brahmanbaria 0.2431 0.7569
## 8       Chandpur 0.7033 0.2967
## 9     Chattogram 0.6000 0.4000
## 10     Chuadanga 0.1340 0.8660
```

```r
# ---- 8) EXPORT FULL MEMBERSHIP TABLE ----
U_full <- as.data.frame(round(U, 4))
colnames(U_full) <- paste0("u", seq_len(ncol(U_full)))
U_full$district_name <- district$district_name
U_full$cluster_fcm <- district$cluster_fcm
U_full$max_membership <- round(district$max_membership, 4)
U_full$top2_gap <- round(district$top2_gap, 4)
if (best_k == 2) U_full$membership_gap_k2 <- round(district$membership_gap_k2, 4)

write.csv(U_full, "FCM_membership_full.csv", row.names = FALSE)
cat("\nSaved: FCM_membership_full.csv\n")
```

```
##
## Saved: FCM_membership_full.csv
```

```r
# Optional nice table:
knitr::kable(amb_tbl, digits = 4,
             caption = "Top 15 most ambiguous districts (smallest membership gap)")
```

Table 3: Top 15 most ambiguous districts (smallest membership gap)

district_name	cluster_fcm	max_membership	.gap
Shariatpur	2	0.5122	0.0245
Narayanganj	2	0.5206	0.0412
Rajshahi	2	0.5239	0.0478
Gopalganj	1	0.5693	0.1385
Gazipur	2	0.5937	0.1874
Chattogram	1	0.6000	0.2000
Mymensingh	1	0.6063	0.2127
Munshiganj	1	0.6288	0.2577
Narail	1	0.6321	0.2643
Sherpur	1	0.6469	0.2939
Manikganj	2	0.6731	0.3462
Madaripur	2	0.6764	0.3529
Narsingdi	2	0.6885	0.3770
Noakhali	1	0.7002	0.4004
Chandpur	1	0.7033	0.4066

#————————————-# # 28. COMPARISON OF METHODS (ARI & NMI) #————————————-#

```r
library(mclust)    # adjustedRandIndex()
library(aricode)   # NMI()

hc_labels  <- district$cluster_hc
km_labels  <- district$cluster_km
fcm_labels <- district$cluster_fcm

# keep only rows where all 3 exist
ok <- complete.cases(hc_labels, km_labels, fcm_labels)

hc <- as.integer(as.factor(hc_labels[ok]))
km <- as.integer(as.factor(km_labels[ok]))
```

```r
fcm <- as.integer(as.factor(fcm_labels[ok]))

# ---------- ARI ----------
mclust::adjustedRandIndex(hc, km)
```

```
## [1] 0.7073171
```

```r
mclust::adjustedRandIndex(hc, fcm)
```

```
## [1] 0.6547619
```

```r
mclust::adjustedRandIndex(km, fcm)
```

```
## [1] 0.9375
```

```r
# ---------- NMI ----------
aricode::NMI(hc, km)
```

```
## [1] 0.675514
```

```r
aricode::NMI(hc, fcm)
```

```
## [1] 0.6333285
```

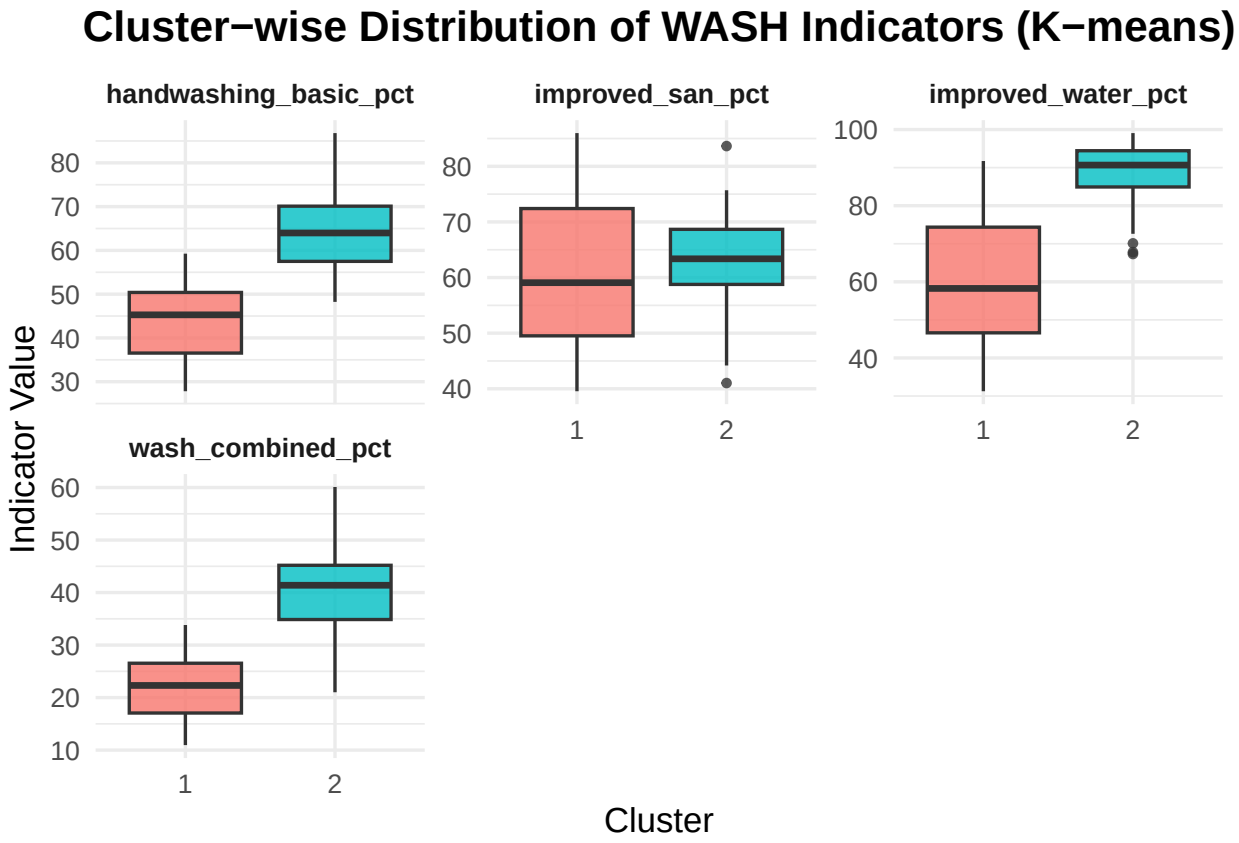
```r
aricode::NMI(km, fcm)
```

```
## [1] 0.8963356
```

32. Cluster-wise Distribution of WASH Indicators (K-means) — FULLY SELF-CONTAINED

```r
library(dplyr)
library(tidyr)
library(ggplot2)

# ----------------------------------------
#  1) RUN K-MEANS AGAIN SAFELY (k = 2)
# ----------------------------------------
set.seed(42)
kmeans_result <- kmeans(wash_scaled, centers = best_k, nstart = 25)

#  Create cluster vector explicitly
cluster_km <- kmeans_result$cluster

# ----------------------------------------
#  2) REBUILD WASH DATA FROM wash_mat
# ----------------------------------------
wash_box_df <- as.data.frame(wash_mat)

# ----------------------------------------
#  3) ATTACH CLUSTERS TO RAW WASH DATA
# ----------------------------------------
wash_box_df$Kmeans_Cluster <- factor(cluster_km)

# ----------------------------------------
#  4) CONVERT TO LONG FORMAT
# ----------------------------------------
```

```r
wash_box_long <- wash_box_df %>%
  dplyr::select(
    Kmeans_Cluster,
    improved_water_pct,
    improved_san_pct,
    handwashing_basic_pct,
    wash_combined_pct
  ) %>%
  pivot_longer(
    cols = -Kmeans_Cluster,
    names_to = "Indicator",
    values_to = "Value"
  )

# ----------------------------------------
#  5) FINAL CLUSTER-WISE BOXPLOT
# ----------------------------------------
p_box_wash <- ggplot(
  wash_box_long,
  aes(x = Kmeans_Cluster, y = Value, fill = Kmeans_Cluster)
) +
  geom_boxplot(outlier.size = 1.2, alpha = 0.8) +
  facet_wrap(~ Indicator, scales = "free_y", ncol = 3) +
  labs(
    title = "Cluster-wise Distribution of WASH Indicators (K-means)",
    x = "Cluster",
    y = "Indicator Value"
  ) +
  theme_minimal(base_size = 13) +
  theme(
    legend.position = "none",
    strip.text = element_text(face = "bold"),
    plot.title = element_text(face = "bold", hjust = 0.5)
  )

print(p_box_wash)
```

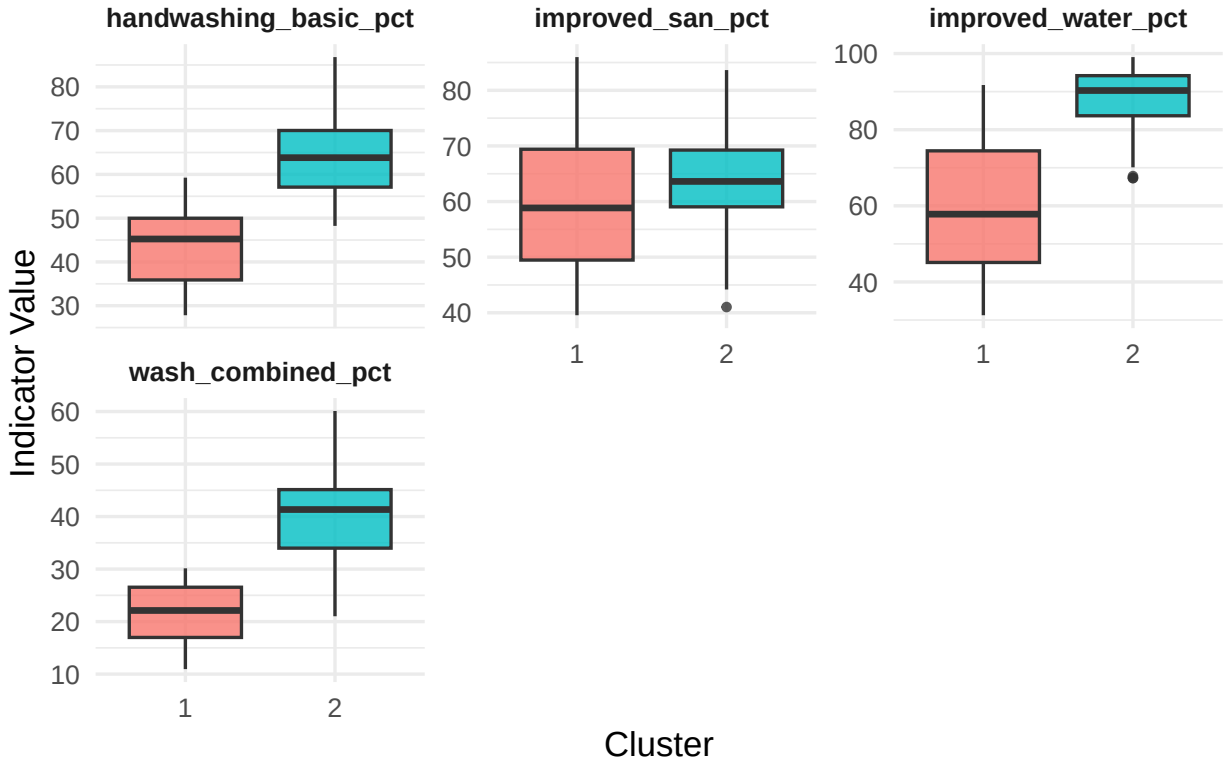



```
# ----------------------------------------
#  6) SAVE HIGH-RESOLUTION FIGURE
# ----------------------------------------
ggsave(
  "Figure_ClusterWise_WASH_Boxplots_Kmeans.png",
  plot = p_box_wash,
  width = 12,
  height = 7,
  dpi = 600
)
```

33. Cluster-wise Distribution of WASH Indicators (Fuzzy C-means)

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(e1071)

# ----------------------------------------
#  1) RUN FUZZY C-MEANS AGAIN SAFELY (k = 2)
# ----------------------------------------
set.seed(42)
fcm_result <- cmeans(wash_scaled, centers = best_k, m = 2)

#  Extract crisp cluster labels from fuzzy membership
cluster_fcm <- apply(fcm_result$membership, 1, which.max)
```

```r
# ----------------------------------------
#  2) REBUILD WASH DATA FROM wash_mat
# ----------------------------------------
wash_box_df <- as.data.frame(wash_mat)

# ----------------------------------------
#  3) ATTACH FUZZY CLUSTERS TO RAW WASH DATA
# ----------------------------------------
wash_box_df$Fuzzy_Cluster <- factor(cluster_fcm)

# ----------------------------------------
#  4) CONVERT TO LONG FORMAT
# ----------------------------------------
wash_box_long <- wash_box_df %>%
  dplyr::select(
    Fuzzy_Cluster,
    improved_water_pct,
    improved_san_pct,
    handwashing_basic_pct,
    wash_combined_pct
  ) %>%
  pivot_longer(
    cols = -Fuzzy_Cluster,
    names_to = "Indicator",
    values_to = "Value"
  )

# ----------------------------------------
#  5) FINAL CLUSTER-WISE BOXPLOT (FUZZY)
# ----------------------------------------
p_box_fuzzy <- ggplot(
  wash_box_long,
  aes(x = Fuzzy_Cluster, y = Value, fill = Fuzzy_Cluster)
) +
  geom_boxplot(outlier.size = 1.2, alpha = 0.8) +
  facet_wrap(~ Indicator, scales = "free_y", ncol = 3) +
  labs(
    title = "Cluster-wise Distribution of WASH Indicators (Fuzzy C-means)",
    x = "Cluster",
    y = "Indicator Value"
  ) +
  theme_minimal(base_size = 13) +
  theme(
    legend.position = "none",
    strip.text = element_text(face = "bold"),
    plot.title = element_text(face = "bold", hjust = 0.5)
  )

print(p_box_fuzzy)
```




```
# ----------------------------------------
#  6) SAVE HIGH-RESOLUTION FIGURE
# ----------------------------------------
ggsave(
  "Figure_ClusterWise_WASH_Boxplots_Fuzzy.png",
  plot = p_box_fuzzy,
  width = 12,
  height = 7,
  dpi = 600
)
```

#——————————-# # 34. SAVE OUTPUT #——————————-#

```
write.csv(district, "district_wash_vulnerability_full.csv", row.names = FALSE)
```

==========

45. FINAL: 64 DISTRICTS, NO NA, MANUAL NAME CONTROL (ONE CHUNK)

==========

```
# ============================================================
# CHAPTER 4 MAPS (FULL SAFE CHUNK - FIXED)
# - FCM map with Transitional districts (membership ambiguity)
# - K-means map (label-consistent)
```

```r
# - Hierarchical map (label-consistent)
#
# FIXES ADDED:
#  1) Safer membership alignment (unique key check)
#  2) Robust gap_top2 computation (no fragile select(.) inside mutate)
#  3) Do NOT stop on unmatched districts; print them and continue
#  4) vuln_score includes wash_combined_pct to stay consistent with 4-indicator analysis
#  5) Prints Status counts for quick sanity check
#
# REQUIREMENTS:
#   district must contain:
#     district_code (or HH7A), improved_*_pct, cluster_km, cluster_hc, cluster_fcm
#   fcm object OR U matrix must exist (membership aligned to SAME districts used in cmeans)
#   district_name_lookup.xlsx: col1 = district_gadm, col2 = district_analysis
#   gadm41_BGD_2.json present
# ============================================================

suppressPackageStartupMessages({
  library(sf)
  library(dplyr)
  library(ggplot2)
  library(stringr)
  library(readxl)
})

# --------------------------------
# 0) SAFETY: choose stable key for alignment + name column
# --------------------------------
stopifnot(exists("district"))

key_col <- dplyr::case_when(
  "district_code" %in% names(district) ~ "district_code",
  "HH7A"          %in% names(district) ~ "HH7A",
  TRUE ~ NA_character_
)
if (is.na(key_col)) stop("No stable key found in `district`. Add `district_code` or `HH7A`.")

name_col <- dplyr::case_when(
  "district_name" %in% names(district) ~ "district_name",
  "district"      %in% names(district) ~ "district",
  TRUE ~ NA_character_
)
if (is.na(name_col)) stop("No district name column found in `district` (expected district_name or distri

# Key must be unique to avoid silent join problems
stopifnot(!anyDuplicated(district[[key_col]]))

# Must have required cluster columns
stopifnot("cluster_km"  %in% names(district),
          "cluster_hc"  %in% names(district),
          "cluster_fcm" %in% names(district))

# Must have the indicators used for vuln_score
```

```r
stopifnot(all(c("improved_water_pct", "improved_san_pct", "handwashing_basic_pct", "wash_combined_pct")

# ------------------------------
# 1) Load GADM map
# ------------------------------
bd_map <- st_read("gadm41_BGD_2.json", quiet = TRUE) %>%
  mutate(district_gadm = str_to_lower(str_trim(NAME_2))) %>%
  dplyr::select(district_gadm, geometry)

# ------------------------------
# 2) Read Excel lookup (ROBUST)
#   col1 = district_gadm, col2 = district_analysis
# ------------------------------
lookup_raw <- read_excel("district_name_lookup.xlsx")
if (ncol(lookup_raw) < 2) stop("district_name_lookup.xlsx must have at least 2 columns (GADM name, analy

lookup <- lookup_raw %>%
  dplyr::rename(
    district_gadm     = 1,
    district_analysis = 2
  ) %>%
  mutate(
    district_gadm     = str_to_lower(str_trim(as.character(district_gadm))),
    district_analysis = str_to_lower(str_trim(as.character(district_analysis)))
  )

if (any(is.na(lookup$district_gadm)) || any(is.na(lookup$district_analysis))) {
  stop("Excel lookup has NA values in district_gadm or district_analysis. Fill all district names.")
}

# ------------------------------
# 3) Vulnerability score (lower = more vulnerable)
#    Consistent with 4-indicator analysis (includes wash_combined_pct)
# ------------------------------
district <- district %>%
  mutate(
    vuln_score = rowSums(
      across(c(improved_water_pct, improved_san_pct, handwashing_basic_pct, wash_combined_pct)),
      na.rm = TRUE
    )
  )

# ------------------------------
# 4) Identify which cluster is "Vulnerable" for each method
#    (cluster with LOWER mean vuln_score)
# ------------------------------
vuln_cluster_km <- district %>%
  group_by(cluster_km) %>%
  summarise(m = mean(vuln_score, na.rm = TRUE), .groups = "drop") %>%
  arrange(m) %>% slice(1) %>% pull(cluster_km)

vuln_cluster_hc <- district %>%
  group_by(cluster_hc) %>%
```

```r
  summarise(m = mean(vuln_score, na.rm = TRUE), .groups = "drop") %>%
  arrange(m) %>% slice(1) %>% pull(cluster_hc)

vuln_cluster_fcm <- district %>%
  group_by(cluster_fcm) %>%
  summarise(m = mean(vuln_score, na.rm = TRUE), .groups = "drop") %>%
  arrange(m) %>% slice(1) %>% pull(cluster_fcm)

# ------------------------------
# 5) FCM membership matrix U (alignment-safe)
# ------------------------------
# Ensure U exists
if (!exists("U")) {
  if (exists("fcm") && !is.null(fcm$membership)) {
    U <- fcm$membership
  } else if (exists("fcm_model") && !is.null(fcm_model$membership)) {
    U <- fcm_model$membership
  } else {
    stop("Membership matrix `U` not found. Run FCM first (fcm <- cmeans(...); U <- fcm$membership).")
  }
}

U <- as.matrix(U)

# Check dimensional match (necessary, not sufficient)
if (nrow(U) != nrow(district)) {
  stop(
    "Row mismatch: nrow(U) = ", nrow(U),
    " but nrow(district) = ", nrow(district),
    ". This usually means `district` was reordered/filtered AFTER fitting FCM.\n",
    "Fix: keep a modeling table in fixed order, and join membership back using the stable key."
  )
}

# Build membership DF tied to the stable key (prevents silent misalignment later)
U_df <- as.data.frame(U)
colnames(U_df) <- paste0("u", seq_len(ncol(U_df)))
if (key_col %in% names(U_df)) U_df[[key_col]] <- NULL
U_df[[key_col]] <- district[[key_col]]
stopifnot(!anyDuplicated(U_df[[key_col]]))

# Choose transitional threshold:
# 0.05 strict, 0.10 moderate, 0.15 lenient
delta <- 0.10

wash_fcm_df <- district %>%
  mutate(
    district_analysis = str_to_lower(str_trim(as.character(.data[[name_col]])))
  ) %>%
  left_join(U_df, by = key_col)

# Robust gap_top2 computation (works for any k)
u_mat <- as.matrix(dplyr::select(wash_fcm_df, dplyr::starts_with("u")))
```

```r
gap_top2_vec <- apply(u_mat, 1, function(u) {
  s <- sort(u, decreasing = TRUE)
  s[1] - s[2]
})

wash_fcm_df <- wash_fcm_df %>%
  mutate(
    gap_top2 = gap_top2_vec,
    transition_flag = as.integer(gap_top2 < delta),
    Status = case_when(
      transition_flag == 1 ~ "Transitional",
      cluster_fcm == vuln_cluster_fcm ~ "Vulnerable",
      TRUE ~ "Relatively Secure"
    )
  ) %>%
  select(district_analysis, Status)

# Join map + lookup + status
bd_map_fcm <- bd_map %>%
  left_join(lookup, by = "district_gadm") %>%
  left_join(wash_fcm_df, by = "district_analysis")

if (any(is.na(bd_map_fcm$Status))) {
  cat("\nUnmatched districts in FCM map (check lookup names):\n")
  print(bd_map_fcm$district_gadm[is.na(bd_map_fcm$Status)])
}

cat("\nFCM map status counts (delta = ", delta, "):\n", sep = "")
```

```
##
## FCM map status counts (delta = 0.1):
```

```r
print(table(bd_map_fcm$Status, useNA = "ifany"))
```

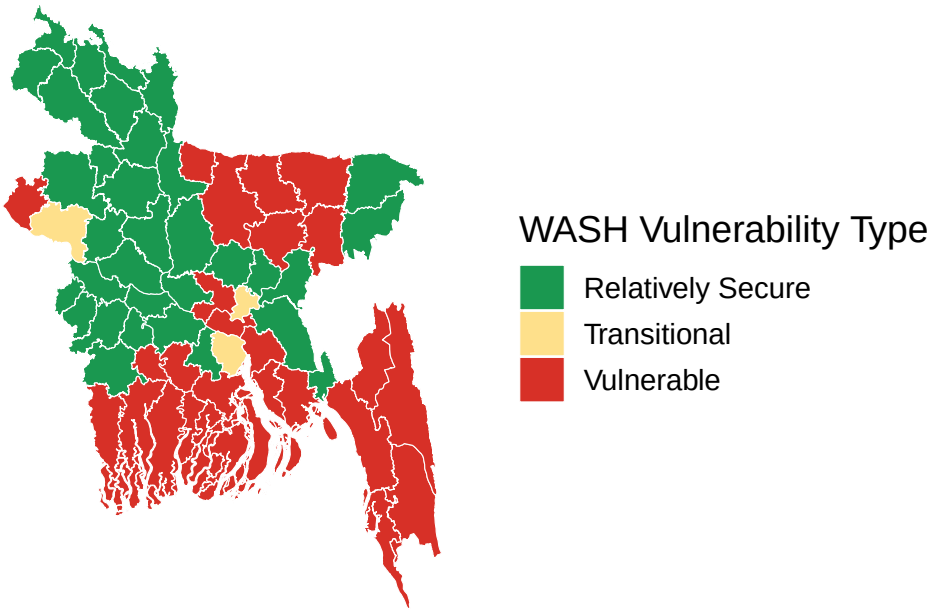
```
##
## Relatively Secure      Transitional        Vulnerable
##                33                 3                28
```

```r
p_fcm <- ggplot(bd_map_fcm) +
  geom_sf(aes(fill = Status), color = "white", linewidth = 0.2) +
  scale_fill_manual(
    values = c(
      "Relatively Secure" = "#1a9850",
      "Transitional"      = "#fee08b",
      "Vulnerable"        = "#d73027"
    ),
    name = "WASH Vulnerability Type",
    drop = FALSE
  ) +
  labs(
    title = "FCM vulnerability typology map.",
    subtitle = paste0("(delta = ", delta, ") vulnerable, transitional, relatively secure districts."),
    caption = "Source: MICS 2019; GADM v4.1; Author's computation"
  ) +
  theme_minimal(base_size = 14) +
```

```
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5),
    plot.caption = element_text(hjust = 0.5),
    axis.text = element_blank(),
    axis.title = element_blank(),
    panel.grid = element_blank()
  )

print(p_fcm)
```

FCM vulnerability typology map.

0.1) vulnerable, transitional, relatively secure districts.



Source: MICS 2019; GADM v4.1; Author's computation

```
ggsave("Figure_Chapter_4.5_FCM_Map.pdf", plot = p_fcm, width = 10, height = 9, units = "in")

# ------------------------------
# 6) K-MEANS MAP (label-consistent)
# ------------------------------
wash_km_df <- district %>%
  mutate(
    district_analysis = str_to_lower(str_trim(as.character(.data[[name_col]]))),
    Status = ifelse(cluster_km == vuln_cluster_km, "Vulnerable", "Relatively Secure")
  ) %>%
  select(district_analysis, Status)

bd_map_km <- bd_map %>%
  left_join(lookup, by = "district_gadm") %>%
  left_join(wash_km_df, by = "district_analysis")
```

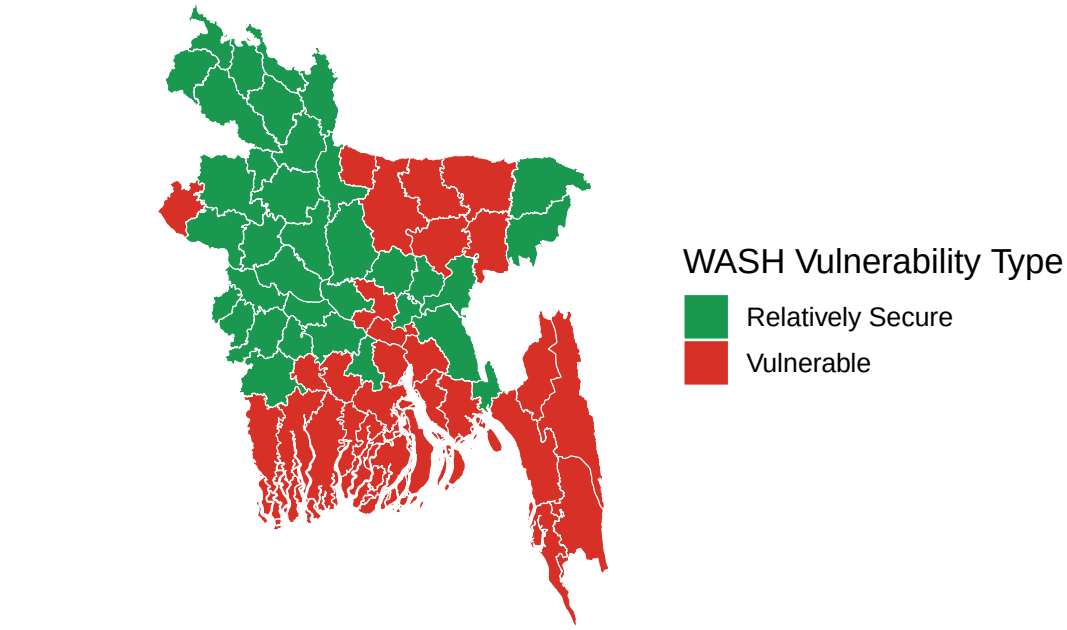
```r
if (any(is.na(bd_map_km$Status))) {
  cat("\nUnmatched districts in K-means map (check lookup names):\n")
  print(bd_map_km$district_gadm[is.na(bd_map_km$Status)])
}

cat("\nK-means map status counts:\n")
```

```
##
## K-means map status counts:
```

```r
print(table(bd_map_km$Status, useNA = "ifany"))
```

```
##
## Relatively Secure        Vulnerable
##                35                29
```

```r
p_km <- ggplot(bd_map_km) +
  geom_sf(aes(fill = Status), color = "white", linewidth = 0.2) +
  scale_fill_manual(
    values = c("Relatively Secure" = "#1a9850", "Vulnerable" = "#d73027"),
    name = "WASH Vulnerability Type",
    drop = FALSE
  ) +
  labs(
    title = "K-means vulnerability map (k = 2):\n higher vs lower vulnerability districts.",
    caption = "Source: MICS 2019; GADM v4.1; Author's computation"
  ) +
  theme_minimal(base_size = 13) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.text = element_blank(),
    axis.title = element_blank(),
    panel.grid = element_blank()
  )

print(p_km)
```



Source: MICS 2019; GADM v4.1; Author's computation

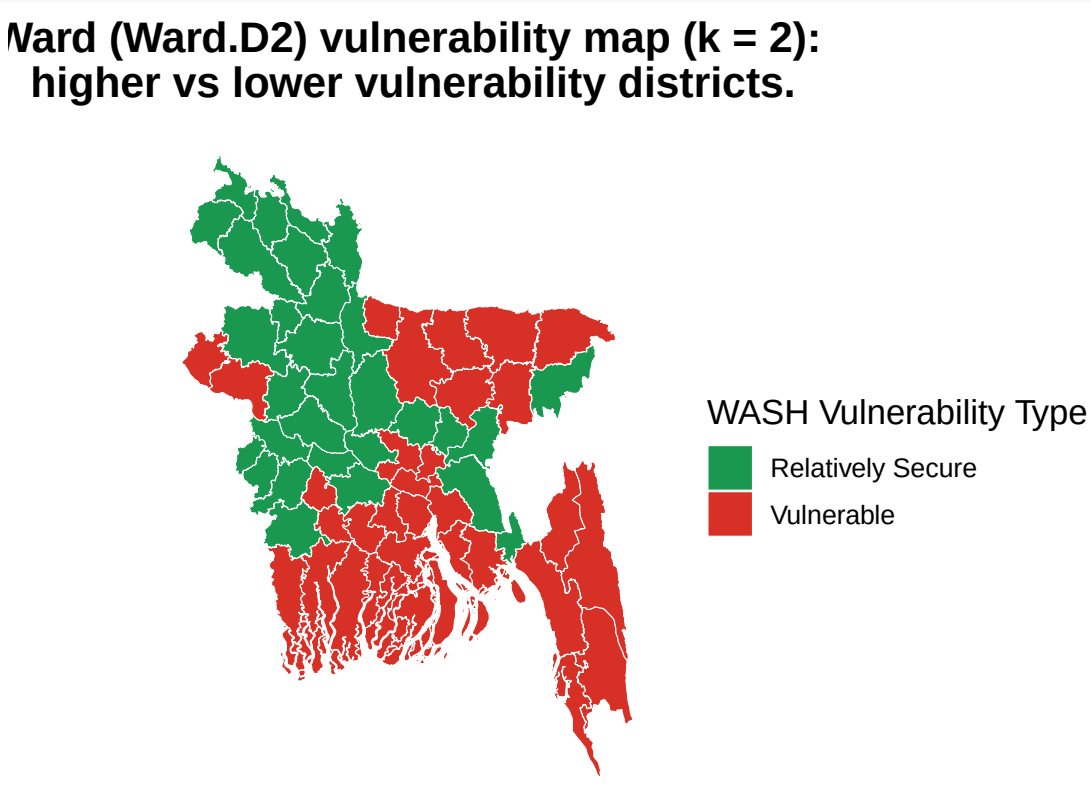
```
ggsave("Figure_Chapter_4.5_KMeans_Map.pdf", plot = p_km, width = 8, height = 9, units = "in")

# --------------------------------
# 7) HIERARCHICAL MAP (label-consistent)
# --------------------------------
wash_hc_df <- district %>%
  mutate(
    district_analysis = str_to_lower(str_trim(as.character(.data[[name_col]]))),
    Status = ifelse(cluster_hc == vuln_cluster_hc, "Vulnerable", "Relatively Secure")
  ) %>%
  select(district_analysis, Status)

bd_map_hc <- bd_map %>%
  left_join(lookup, by = "district_gadm") %>%
  left_join(wash_hc_df, by = "district_analysis")

if (any(is.na(bd_map_hc$Status))) {
  cat("\nUnmatched districts in Hierarchical map (check lookup names):\n")
  print(bd_map_hc$district_gadm[is.na(bd_map_hc$Status)])
}

cat("\nHierarchical map status counts:\n")
```

```
##
## Hierarchical map status counts:
```

```r
print(table(bd_map_hc$Status, useNA = "ifany"))
```

```
##
## Relatively Secure        Vulnerable
##                30                34
```

```r
p_hc <- ggplot(bd_map_hc) +
  geom_sf(aes(fill = Status), color = "white", linewidth = 0.2) +
  scale_fill_manual(
    values = c("Relatively Secure" = "#1a9850", "Vulnerable" = "#d73027"),
    name = "WASH Vulnerability Type",
    drop = FALSE
  ) +
  labs(
    title = "Ward (Ward.D2) vulnerability map (k = 2):\n higher vs lower vulnerability districts.",
    caption = "Source: MICS 2019; GADM v4.1; Author's computation"
  ) +
  theme_minimal(base_size = 13) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.text = element_blank(),
    axis.title = element_blank(),
    panel.grid = element_blank()
  )

print(p_hc)
```



```r
ggsave("Figure_Chapter_4.5_Hierarchical_Map.pdf", plot = p_hc, width = 9, height = 9, units = "in")
```

```r
# ============================================================
# CLUSTER-WISE INDICATOR PROFILE (MEANS) - FIXED (ROBUST)
# ============================================================

library(dplyr)

# If cluster_fcm_hard doesn't exist, fall back to cluster_fcm
if (!"cluster_fcm_hard" %in% names(district)) {
  district <- district %>% mutate(cluster_fcm_hard = cluster_fcm)
}

cluster_profile_table <- district %>%
  group_by(cluster_fcm_hard) %>%
  summarise(
    improved_water_pct      = mean(improved_water_pct, na.rm = TRUE),
    improved_san_pct        = mean(improved_san_pct, na.rm = TRUE),
    handwashing_basic_pct   = mean(handwashing_basic_pct, na.rm = TRUE),
    wash_combined_pct       = mean(wash_combined_pct, na.rm = TRUE),
    .groups = "drop"
  ) %>%
  # Pretty column labels (no backticks/quotes needed inside summarise)
  rename(
    `Improved drinking water (%)` = improved_water_pct,
    `Improved sanitation (%)` = improved_san_pct,
    `Basic handwashing facility (%)` = handwashing_basic_pct,
    `combined wash dacility(%)` = wash_combined_pct
  )
cluster_profile_table
```

```
## # A tibble: 2 x 5
##   cluster_fcm_hard `Improved drinking water (%)` `Improved sanitation (%)`
##   <fct>                                   <dbl>                     <dbl>
## 1 1                                        58.4                      59.9
## 2 2                                        87.5                      63.7
## # i 2 more variables: `Basic handwashing facility (%)` <dbl>,
## #   `combined wash dacility(%)` <dbl>
```

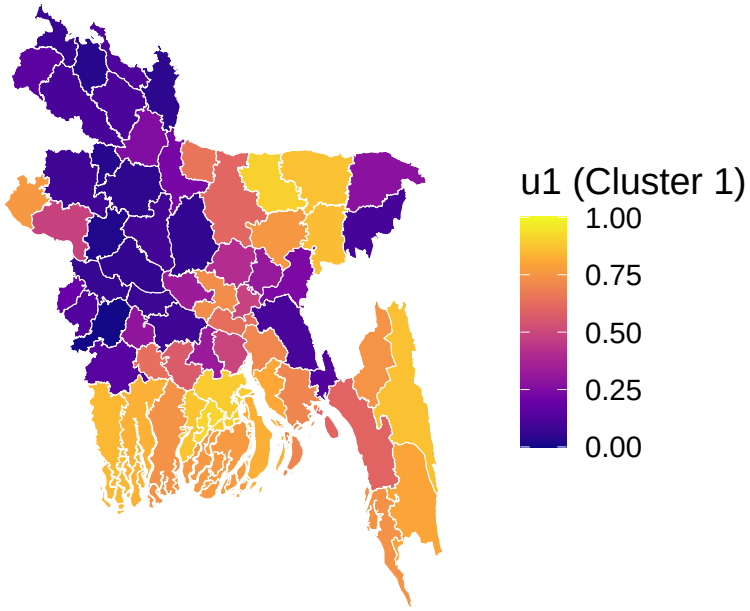
```r
# ============================================================
# FCM MEMBERSHIP MAP (using your SAME GADM + Excel lookup flow)
# Outputs:
#   1) Map of membership to Cluster 1 (u1)
#   2) Map of membership to Cluster 2 (u2)
#   3) Map of max membership (certainty)
#   4) Map of top-2 gap (ambiguity)  [optional but useful]
# ============================================================

library(sf)
library(dplyr)
library(ggplot2)
library(stringr)
library(readxl)
```

```r
# --------------------------------
# 1) Load GADM district map (ADM2)
# --------------------------------
bd_map <- st_read("gadm41_BGD_2.json", quiet = TRUE) %>%
  mutate(district_gadm = str_to_lower(str_trim(NAME_2))) %>%
  dplyr::select(district_gadm, geometry)

# --------------------------------
# 2) Read Excel lookup: GADM name -> Analysis name
#    Excel column 1 = GADM district name
#    Excel column 2 = your analysis district name
# --------------------------------
lookup_raw <- read_excel("district_name_lookup.xlsx")
lookup <- lookup_raw %>%
  rename(
    district_gadm     = 1,
    district_analysis = 2
  ) %>%
  mutate(
    district_gadm     = str_to_lower(str_trim(district_gadm)),
    district_analysis = str_to_lower(str_trim(district_analysis))
  )

if (any(is.na(lookup$district_analysis))) {
  stop("Excel lookup has NA values. Fill all district names.")
}

# --------------------------------
# 3) SAFE district name column in your `district` table
# --------------------------------
name_col <- dplyr::case_when(
  "district_name" %in% names(district) ~ "district_name",
  "district"      %in% names(district) ~ "district",
  TRUE ~ NA_character_
)
if (is.na(name_col)) stop("No district name column found in `district`.")

# --------------------------------
# 4) Ensure membership matrix U exists
#    (From your earlier FCM chunk: U <- fcm_model$membership)
# --------------------------------
if (!exists("U")) {
  if (exists("fcm_model") && !is.null(fcm_model$membership)) {
    U <- fcm_model$membership
  } else if (exists("fcm") && !is.null(fcm$membership)) {
    U <- fcm$membership
  } else {
    stop("Membership matrix `U` not found. Run FCM first: fcm_model <- cmeans(...); U <- fcm_model$membe
  }
}

U <- as.matrix(U)
stopifnot(nrow(U) == nrow(district))
```

```r
stopifnot(ncol(U) >= 2)

# --------------------------------
# 5) Build membership dataset aligned with `district`
# --------------------------------
membership_df <- district %>%
  mutate(
    district_analysis = str_to_lower(str_trim(.data[[name_col]])),
    u1 = U[, 1],
    u2 = U[, 2],
    max_membership = pmax(u1, u2),
    gap_top2 = apply(U, 1, function(u) { s <- sort(u, decreasing = TRUE); s[1] - s[2] })
  ) %>%
  select(district_analysis, u1, u2, max_membership, gap_top2)

# --------------------------------
# 6) Join map + lookup + membership
# --------------------------------
bd_map_mem <- bd_map %>%
  left_join(lookup, by = "district_gadm") %>%
  left_join(membership_df, by = "district_analysis")

if (any(is.na(bd_map_mem$u1))) {
  stop(
    "Unmatched districts (check Excel lookup names):\n",
    paste(bd_map_mem$district_gadm[is.na(bd_map_mem$u1)], collapse = ", ")
  )
}

# ============================================================
# MAP A) Membership to Cluster 1 (u1)
# ============================================================
p_u1 <- ggplot(bd_map_mem) +
  geom_sf(aes(fill = u1), color = "white", linewidth = 0.2) +
  scale_fill_viridis_c(option = "C", limits = c(0, 1), name = "u1 (Cluster 1)") +
  labs(
    title = "FCM Membership Map: Cluster 1 (u1)",
    subtitle = "Higher = stronger membership in Cluster 1",
    caption = "Source: MICS 2019; GADM v4.1; Author's computation"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5),
    plot.caption = element_text(hjust = 0.5),
    axis.text = element_blank(),
    axis.title = element_blank(),
    panel.grid = element_blank()
  )

print(p_u1)
```

FCM Membership Map: Cluster 1 (u1)

Higher = stronger membership in Cluster 1

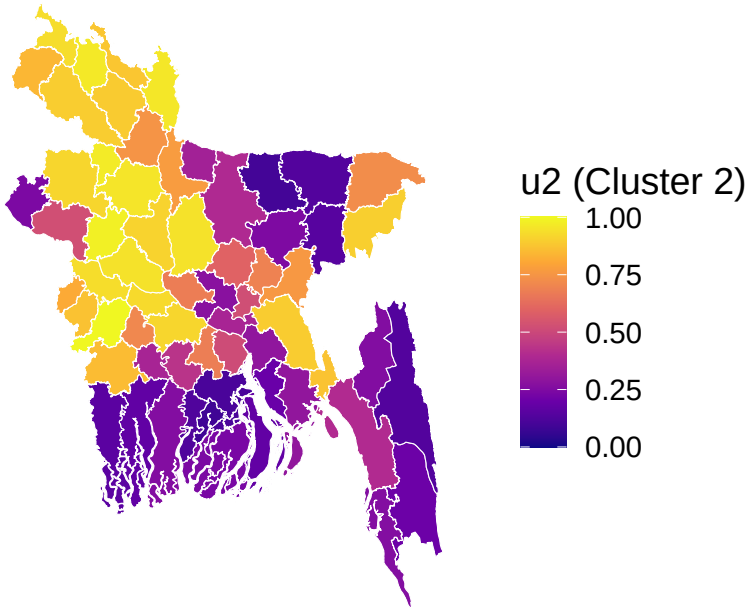


Source: MICS 2019; GADM v4.1; Author's computation

```
ggsave("Map_FCM_4.3.3_u1.pdf", p_u1, width = 8, height = 9, units = "in")

# ============================================================
# MAP B) Membership to Cluster 2 (u2)
# ============================================================
p_u2 <- ggplot(bd_map_mem) +
  geom_sf(aes(fill = u2), color = "white", linewidth = 0.2) +
  scale_fill_viridis_c(option = "C", limits = c(0, 1), name = "u2 (Cluster 2)") +
  labs(
    title = "FCM Membership Map: Cluster 2 (u2)",
    subtitle = "Higher = stronger membership in Cluster 2",
    caption = "Source: MICS 2019; GADM v4.1; Author's computation"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5),
    plot.caption = element_text(hjust = 0.5),
    axis.text = element_blank(),
    axis.title = element_blank(),
    panel.grid = element_blank()
  )

print(p_u2)
```

FCM Membership Map: Cluster 2 (u2)

Higher = stronger membership in Cluster 2

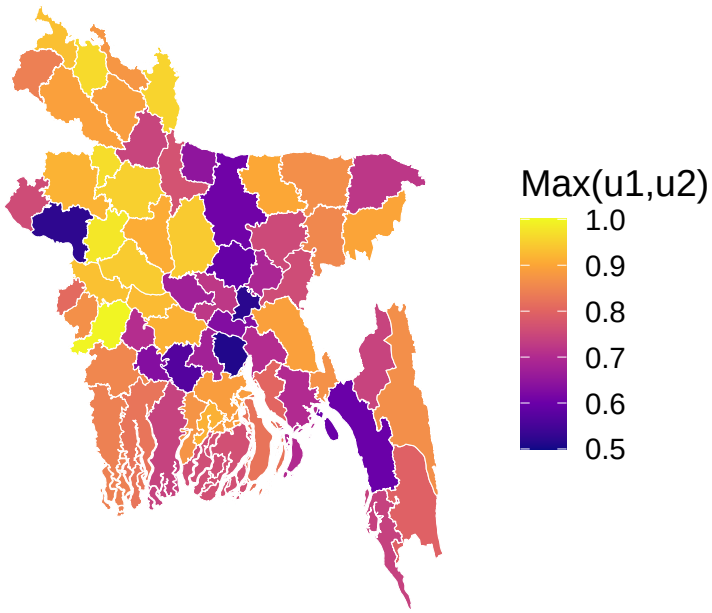


Source: MICS 2019; GADM v4.1; Author's computation

```
ggsave("Map_FCM_4.3.3_u2.pdf", p_u2, width = 8, height = 9, units = "in")

# ============================================================
# MAP C) Max membership (certainty)
# ============================================================
p_cert <- ggplot(bd_map_mem) +
  geom_sf(aes(fill = max_membership), color = "white", linewidth = 0.2) +
  scale_fill_viridis_c(option = "C", limits = c(0.5, 1), name = "Max(u1,u2)") +
  labs(
    title = "FCM Membership Certainty Map (Max Membership)",
    subtitle = "Higher = more certain membership",
    caption = "Source: MICS 2019; GADM v4.1; Author's computation"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5),
    plot.caption = element_text(hjust = 0.5),
    axis.text = element_blank(),
    axis.title = element_blank(),
    panel.grid = element_blank()
  )

print(p_cert)
```

M Membership Certainty Map (Max Membership)

Higher = more certain membership

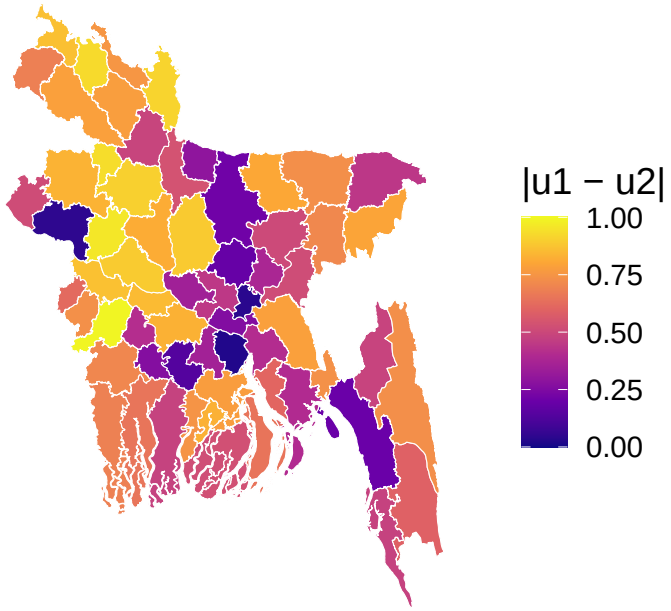


Source: MICS 2019; GADM v4.1; Author's computation

```
ggsave("Map_FCM_4.3.3_MaxMembership_certainity.pdf", p_cert, width = 8, height = 9, units = "in")

# ============================================================
# MAP D) Ambiguity map (top-2 gap)  [optional but very useful]
# Smaller gap = more ambiguous / more transitional
# ============================================================
p_gap <- ggplot(bd_map_mem) +
  geom_sf(aes(fill = gap_top2), color = "white", linewidth = 0.2) +
  scale_fill_viridis_c(option = "C", limits = c(0, 1), name = "|u1 - u2|") +
  labs(
    title = "FCM Transitional Map (|u1 - u2|)",
    subtitle = "Smaller gap = more uncertain (more transitional)",
    caption = "Source: MICS 2019; GADM v4.1; Author's computation"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5),
    plot.caption = element_text(hjust = 0.5),
    axis.text = element_blank(),
    axis.title = element_blank(),
    panel.grid = element_blank()
  )

print(p_gap)
```

FCM Transitional Map (|u1 − u2|)

Smaller gap = more uncertain (more transitional)



Source: MICS 2019; GADM v4.1; Author's computation

```
ggsave("FCM_4.3.3_Transitional_Map.pdf", p_gap, width = 8, height = 9, units = "in")
```

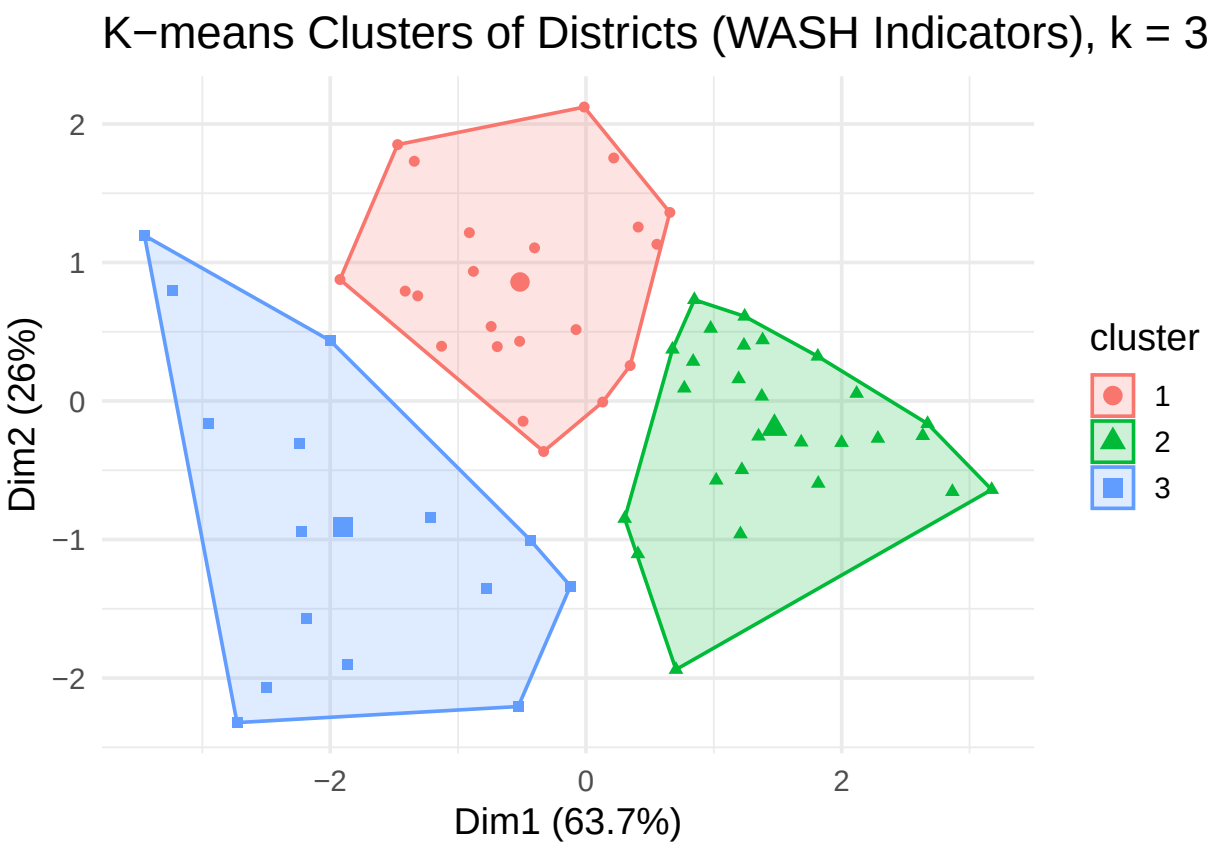
```
# ============================================================
# k = 3 SENSITIVITY RUN (MATCHING YOUR anal.pdf STYLE)
# - Hierarchical (Ward.D2) + dendrogram + rectangles (k=3)
# - K-means + fviz_cluster
# - Fuzzy C-means (FULL FIXED CHUNK + PRINT RESULTS) for k=3
# - PCA scatter plot (PC1 vs PC2) for each method
# ============================================================

suppressPackageStartupMessages({
  library(dplyr)
  library(cluster)
  library(factoextra)
  library(ggplot2)
  library(e1071)
})

set.seed(42)

# ----------------------------
# 0) SAFETY CHECKS (same spirit as your file)
# ----------------------------
stopifnot(exists("district"))
stopifnot(exists("wash_scaled"))

wash_scaled <- as.matrix(wash_scaled)
```

```r
stopifnot(all(is.finite(wash_scaled)))
stopifnot(nrow(wash_scaled) == nrow(district))

# Force k = 3
best_k <- 3
stopifnot(is.numeric(best_k) && best_k >= 2)

# Ensure district name exists for labels
name_col <- dplyr::case_when(
  "district_name" %in% names(district) ~ "district_name",
  "district"      %in% names(district) ~ "district",
  TRUE ~ NA_character_
)
if (is.na(name_col)) stop("No district name column found in `district`.")

# ============================================================
# 21) HIERARCHICAL CLUSTERING (k = 3)
# ============================================================

dist_mat_k3 <- dist(wash_scaled, method = "euclidean")
hc_k3 <- hclust(dist_mat_k3, method = "ward.D2")

# Attach cluster labels (DO NOT overwrite hc object)
district$cluster_hc_k3 <- factor(cutree(hc_k3, k = best_k))

# ---- DENDROGRAM (clean + wide save) ----
pdf("DENDROGRAM_WARD_k3.pdf", width = 16, height = 9)
par(mar = c(6, 4, 4, 2))
plot(
  hc_k3,
  labels = district[[name_col]],
  cex = 0.60,
  hang = -1,
  main = "Hierarchical Clustering of Districts (Ward.D2), k = 3",
  xlab = "",
  sub = "",
  ylab = "Height"
)
rect.hclust(hc_k3, k = best_k, border = c("red", "goldenrod3", "darkgreen"))
dev.off()
```

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

```r
# ============================================================
# 22) K-MEANS CLUSTERING (k = 3)
# ============================================================

set.seed(42)
km_k3 <- kmeans(wash_scaled, centers = best_k, nstart = 50)

district$cluster_km_k3 <- factor(km_k3$cluster)

# Visualize like your anal.pdf (fviz_cluster)
```

```
p_km_k3 <- fviz_cluster(
  object = km_k3,
  data = as.data.frame(wash_scaled),
  geom = "point",
  repel = TRUE,
  main = "K-means Clusters of Districts (WASH Indicators), k = 3"
) + theme_minimal(base_size = 14)

print(p_km_k3)
```




```
ggsave("PLOT_KMEANS_k3.pdf", p_km_k3, width = 10, height = 7)

# ============================================================
# 23) FUZZY C-MEANS (FCM) (k = 3) - FULL FIXED CHUNK STYLE
# ============================================================

cat("\n==============================\n")
```

```
##
## ==============================
```

```
cat("FUZZY C-MEANS (FCM) RUN SUMMARY\n")
```

```
## FUZZY C-MEANS (FCM) RUN SUMMARY
```

```
cat("==============================\n")
```

```
## ==============================
```

```r
cat("k (best_k) =", best_k, "\n")
```

```
## k (best_k) = 3
```

```r
m_val <- 2

# Fit fresh model for k=3 (do not reuse old k=2 membership)
fcm_model_k3 <- e1071::cmeans(wash_scaled, centers = best_k, m = m_val)
U_k3 <- as.matrix(fcm_model_k3$membership)  # n x 3
stopifnot(ncol(U_k3) == best_k)

# Row sum check
rs <- rowSums(U_k3)
cat("\nMembership row-sum check (should be ~1):\n")
```

```
##
## Membership row-sum check (should be ~1):
```

```r
print(summary(rs))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1       1       1       1       1       1
```

```r
stopifnot(max(abs(rs - 1)) < 1e-6)

# Crisp labels (dominant membership)
district$cluster_fcm_k3 <- factor(apply(U_k3, 1, which.max))

cat("\nFCM crisp cluster sizes (k=3):\n")
```

```
##
## FCM crisp cluster sizes (k=3):
```

```r
print(table(district$cluster_fcm_k3))
```

```
##
##  1  2  3
## 24 27 13
```

```r
# Certainty + ambiguity (works for any k)
district$max_membership_k3 <- apply(U_k3, 1, max)
district$top2_gap_k3 <- apply(U_k3, 1, function(u) {
  s <- sort(u, decreasing = TRUE)
  s[1] - s[2]
})

cat("\nMax-membership (certainty) summary (k=3):\n")
```

```
##
## Max-membership (certainty) summary (k=3):
```

```r
print(summary(district$max_membership_k3))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4408  0.6044  0.7272  0.7110  0.8370  0.9704
```

```r
cat("\nTop-2 membership gap summary (smaller = more ambiguous) (k=3):\n")
```

```
##
```

```
## Top-2 membership gap summary (smaller = more ambiguous) (k=3):
```

```
print(summary(district$top2_gap_k3))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.01613 0.36563 0.56062 0.51998 0.72733 0.95155
```

```
# Transitional sensitivity (like your thresholds block, but uses top2_gap for k>2)
thresholds <- c(0.05, 0.10, 0.15)
cat("\nTransitional district counts (k=3; using top2_gap thresholds):\n")
```

```
##
## Transitional district counts (k=3; using top2_gap thresholds):
```

```
for (thr in thresholds) {
  nm <- paste0("transition_fcm_k3_thr", gsub("\\.", "", sprintf("%.2f", thr)))
  district[[nm]] <- as.integer(district$top2_gap_k3 < thr)
  cat(" threshold <", thr, ": ", sum(district[[nm]], na.rm = TRUE), " districts\n", sep = "")
}
```

```
##  threshold <0.05: 3 districts
##  threshold <0.1: 6 districts
##  threshold <0.15: 7 districts
```

```
# List most ambiguous (top 15)
cat("\nMost ambiguous districts (TOP 15; smallest top2_gap = most ambiguous):\n")
```

```
##
## Most ambiguous districts (TOP 15; smallest top2_gap = most ambiguous):
```

```
amb_tbl_k3 <- district %>%
  mutate(.gap = top2_gap_k3) %>%
  arrange(.gap) %>%
  select(district_name = all_of(name_col), cluster_fcm_k3, max_membership_k3, .gap) %>%
  head(15)
print(amb_tbl_k3)
```

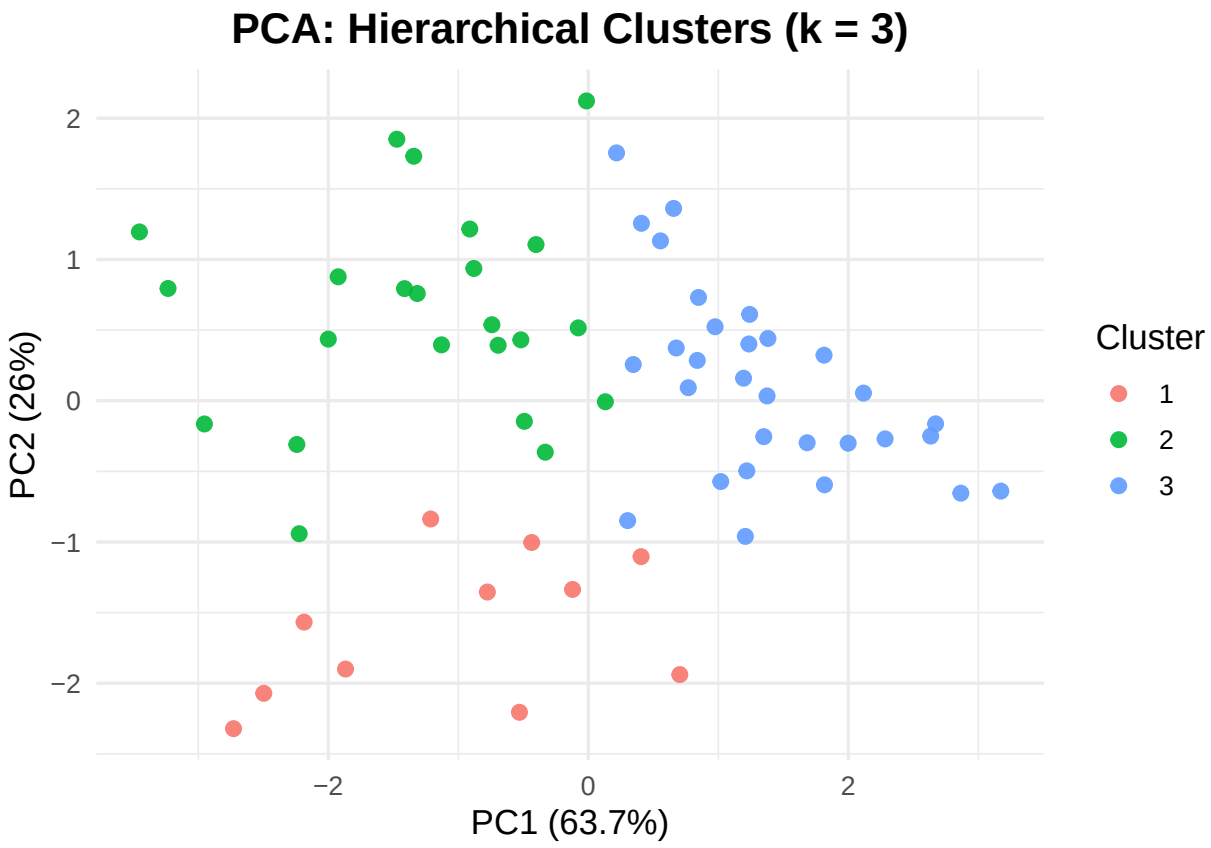
```
## # A tibble: 15 x 4
##    district_name cluster_fcm_k3 max_membership_k3   .gap
##    <chr>         <fct>                      <dbl>  <dbl>
##  1 Jamalpur      1                          0.448 0.0161
##  2 Bandarban     1                          0.441 0.0212
##  3 Rangamati     1                          0.453 0.0234
##  4 Bhola         3                          0.462 0.0501
##  5 Gaibandha     1                          0.487 0.0730
##  6 Brahmanbaria  1                          0.503 0.0998
##  7 Magura        1                          0.479 0.103
##  8 Shariatpur    3                          0.454 0.157
##  9 Manikganj     2                          0.454 0.161
## 10 Chattogram    1                          0.481 0.189
## 11 Madaripur     2                          0.492 0.191
## 12 Narsingdi     1                          0.587 0.271
## 13 Narail        3                          0.535 0.272
## 14 Sylhet        2                          0.522 0.272
## 15 Noakhali      1                          0.591 0.345
```

```
# Export membership table (k=3)
U_full_k3 <- as.data.frame(round(U_k3, 4))
```

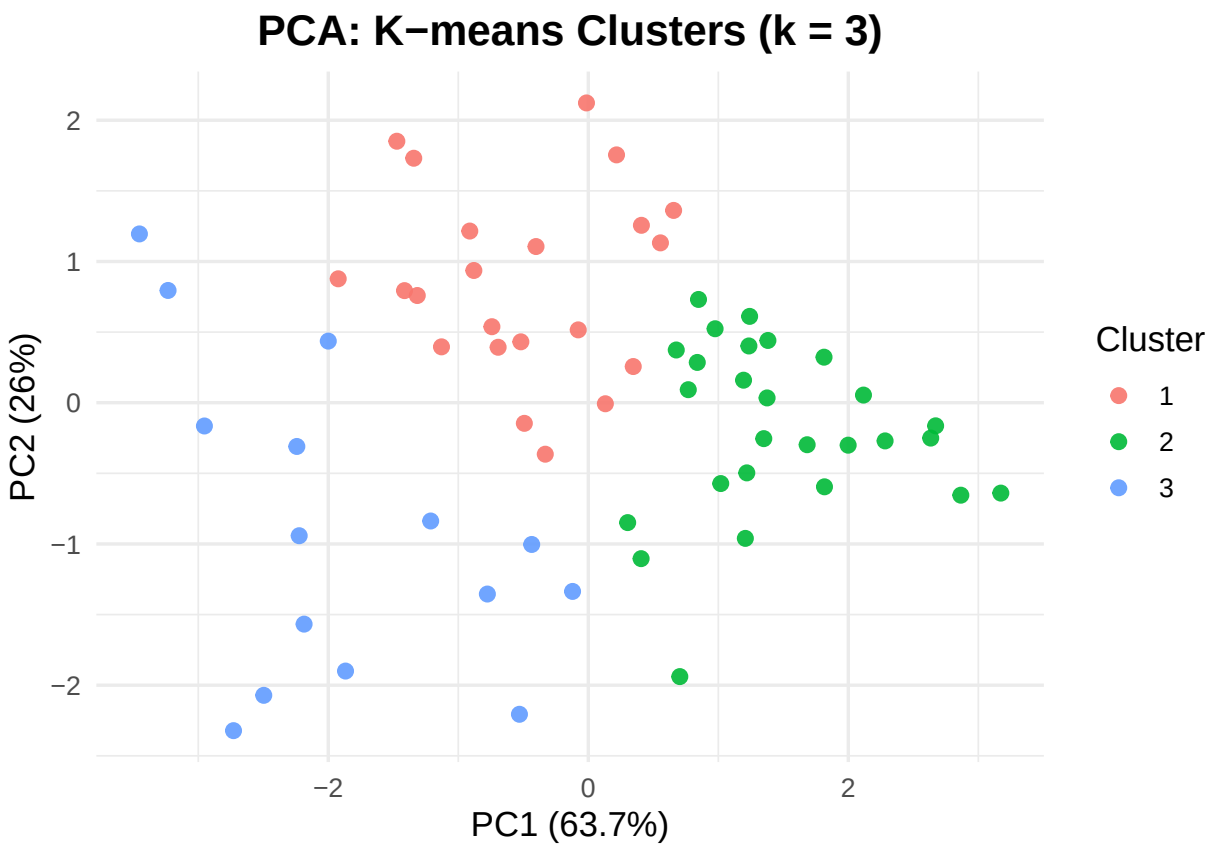
```r
colnames(U_full_k3) <- paste0("u", seq_len(ncol(U_full_k3)))
U_full_k3$district_name <- district[[name_col]]
U_full_k3$cluster_fcm_k3 <- district$cluster_fcm_k3
U_full_k3$max_membership_k3 <- round(district$max_membership_k3, 4)
U_full_k3$top2_gap_k3 <- round(district$top2_gap_k3, 4)

write.csv(U_full_k3, "FCM_membership_k3_full.csv", row.names = FALSE)
cat("\nSaved: FCM_membership_k3_full.csv\n")
```

```
##
## Saved: FCM_membership_k3_full.csv
```

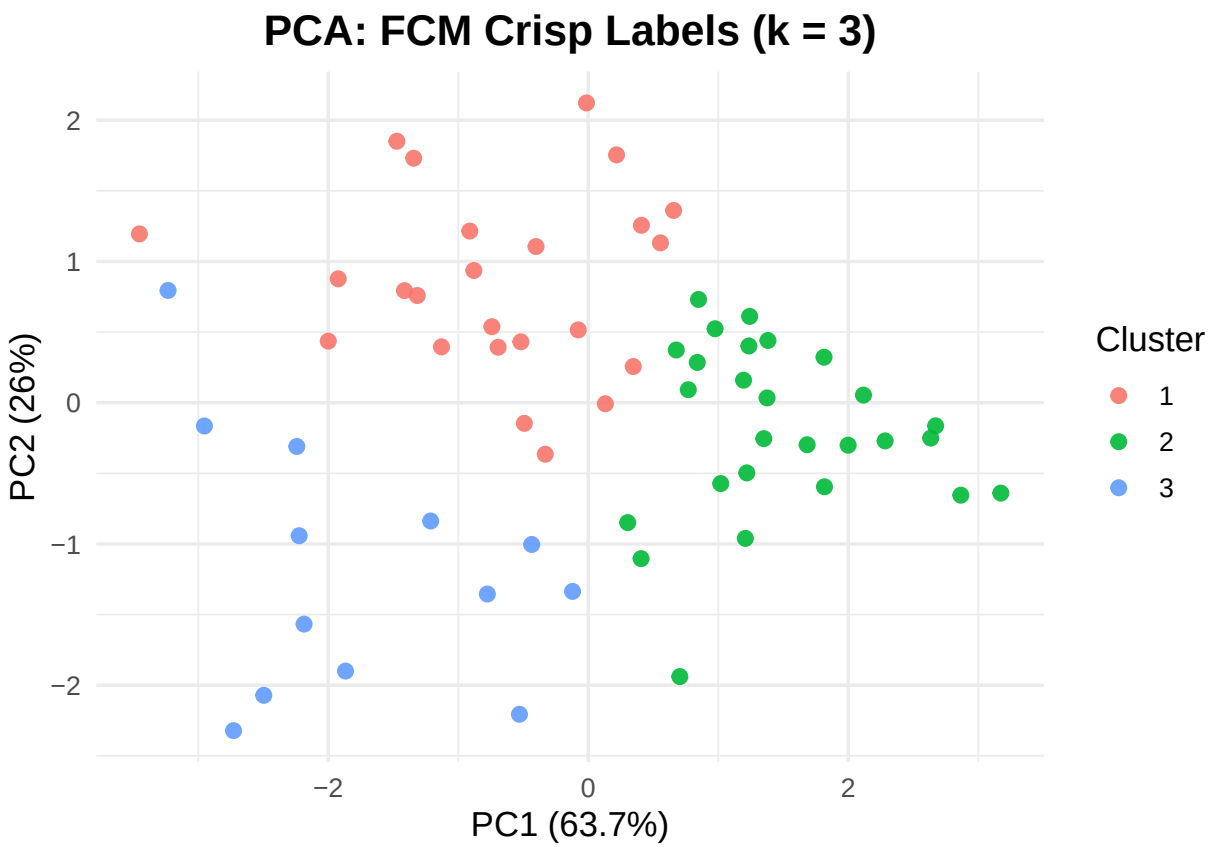
```r
# ============================================================
# PCA PLOTS (PC1 vs PC2) FOR EACH METHOD (k=3)
# ============================================================

pca <- prcomp(wash_scaled, center = TRUE, scale. = FALSE)
pca_df <- as.data.frame(pca$x[, 1:2])
names(pca_df) <- c("PC1", "PC2")
pca_df$District <- district[[name_col]]
pca_df$HC  <- district$cluster_hc_k3
pca_df$KM  <- district$cluster_km_k3
pca_df$FCM <- district$cluster_fcm_k3

var_exp <- 100 * summary(pca)$importance[2, 1:2]
xlab1 <- paste0("PC1 (", round(var_exp[1], 1), "%)")
ylab1 <- paste0("PC2 (", round(var_exp[2], 1), "%)")

plot_pca <- function(lbl, ttl, out){
  p <- ggplot(pca_df, aes(x = PC1, y = PC2, color = .data[[lbl]])) +
    geom_point(size = 2.2, alpha = 0.9) +
    labs(title = ttl, x = xlab1, y = ylab1, color = "Cluster") +
    theme_minimal(base_size = 13) +
    theme(plot.title = element_text(hjust = 0.5, face = "bold"))
  print(p)
  ggsave(out, p, width = 9, height = 6)
}

plot_pca("HC",  "PCA: Hierarchical Clusters (k = 3)", "PCA_HC_k3.pdf")
```




```
plot_pca("KM",  "PCA: K-means Clusters (k = 3)",      "PCA_KMEANS_k3.pdf")
```




```
plot_pca("FCM", "PCA: FCM Crisp Labels (k = 3)",      "PCA_FCM_k3.pdf")
```




```
cat("\nDONE: k=3 dendrogram + PCA + clustering outputs saved.\n")
```

```
##
## DONE: k=3 dendrogram + PCA + clustering outputs saved.
```