Assignment 8 — DEA: Hope Valley Health Care Association

Sonali Shankeshi

2025-10-21

This report applies **Data Envelopment Analysis (DEA)** to six nursing homes (Hope Valley). We use **input orientation** and compute efficiencies under **FDH, CRS, VRS, IRS, DRS, FRH**. Peers and Lambdas are listed for each DMU.

Data and Preparation

The dmu_data contains six Hope Valley facilities, with two inputs: Staff_hours (hours/day) and Supplies (thousands of dollars/day), and two outputs: Reimb_thirdparty_days and Reimb_private_days (patient-days), as listed precisely in the exhibit.To satisfy the inputs required by Benchmarking::dea(), inputs are accumulated into matrix X (Staff_hours, Supplies), and outputs are accumulated into matrix Y (Reimb_thirdparty_days, Reimb_private_days).

The function kable() prints the full table in the report so they can glance through the entire table for a quick visual verification prior to estimating the model.

```
dmu_data <- tibble::tribble(
  ~DMU,         ~Staff_hours, ~Supplies, ~Reimb_thirdparty_days,
~Reimb_private_days,
  "Facility 1",       150,        0.2,             14000,
3500,
  "Facility 2",       400,        0.7,             14000,
21000,
  "Facility 3",       320,        1.2,             42000,
10500,
  "Facility 4",       520,        2.0,             28000,
42000,
  "Facility 5",       350,        1.2,             19000,
25000,
  "Facility 6",       320,        0.7,             14000,
15000
)

X <- as.matrix(dmu_data %>% select(Staff_hours, Supplies))
Y <- as.matrix(dmu_data %>% select(Reimb_thirdparty_days,
Reimb_private_days))

kable(dmu_data, caption = "Hope Valley data (inputs and outputs)")
```

Hope Valley data (inputs and outputs)

DMU	Staff_hours	Supplies	Reimb_thirdparty_days	Reimb_private_days
Facility 1	150	0.2	14000	3500
Facility 2	400	0.7	14000	21000
Facility 3	320	1.2	42000	10500
Facility 4	520	2.0	28000	42000
Facility 5	350	1.2	19000	25000
Facility 6	320	0.7	14000	15000

DEA Models

In this step, we first entered the Hope Valley table into dmu_data, which involves separating the input matrix X (Staff_hours, Supplies) from output matrix Y (Reimbursed and Private patient-days) into the matrices that the DEA function should contain. Let's just add a couple of small "safe" helpers to pull peers and lambda weights without errors (in particular, for FDH/FRH). Subsequently, we ran input-oriented DEA under all six assumptions (FDH, CRS, VRS, IRS, DRS, FRH) and, for each facility, compiled the efficiency score, peer set, and lambdas. Finally, we tidied the peers and lambdas into long-and-readable text before combining all information into one tidy results table for the subsequent summary tables and comparisons.

```
safe_peers <- function(fit, X, Y) {
  out <- tryCatch(Benchmarking::peers(fit, XREF = X, YREF = Y),
                  error = function(e) NULL)
  if (is.null(out)) replicate(nrow(X), integer(0), simplify = FALSE) else out
}

safe_lambda <- function(fit, K) {
  lam <- tryCatch(Benchmarking::lambda(fit), error = function(e) NULL)
  if (is.null(lam)) matrix(0, K, K) else {
    if (is.vector(lam)) lam <- matrix(lam, nrow = K, ncol = K, byrow = TRUE)
    lam
  }
}

run_dea <- function(rts_label, rts_code) {
  K <- nrow(X)
  fit <- Benchmarking::dea(X, Y, RTS = rts_code, ORIENTATION = "in")
  peers_list <- safe_peers(fit, X, Y)
  lambda_mat <- safe_lambda(fit, K)
  tibble(
    DMU = dmu_data$DMU,
    RTS = rts_label,
    eff = as.numeric(fit$eff),
    peers_idx = peers_list,
```

```
    lambda = list(lambda_mat)
  )
}

rts_map <- c(FDH = "fdh", CRS = "crs", VRS = "vrs",
             IRS = "irs", DRS = "drs", FRH = "add")

results_raw <- purrr::imap_dfr(rts_map, ~ run_dea(.y, .x))

format_peers <- function(idx_vec) {
  if (length(idx_vec) == 0 || all(is.na(idx_vec))) return("")
  paste(dmu_data$DMU[na.omit(as.integer(idx_vec))], collapse = ", ")
}
format_lambdas <- function(lambda_mat, row_i, tol = 1e-8) {
  if (is.null(dim(lambda_mat)) || any(dim(lambda_mat) == 0)) return("")
  w <- suppressWarnings(as.numeric(lambda_mat[row_i, ]))
  if (length(w) == 0 || all(is.na(w))) return("")
  sel <- which(w > tol)
  if (length(sel) == 0) return("")
  paste0(dmu_data$DMU[sel], ":", sprintf("%.3f", w[sel]), collapse = "; ")
}

results <- results_raw %>%
  group_by(RTS) %>%
  mutate(
    Peers = purrr::map_chr(peers_idx, format_peers),
    Lambdas = purrr::map2_chr(lambda, dplyr::row_number(), format_lambdas)
  ) %>%
  ungroup() %>%
  select(DMU, RTS, eff, Peers, Lambdas)
```

Results

Efficiency (wide view)

```
eff_wide <- results %>%
  select(DMU, RTS, eff) %>%
  tidyr::pivot_wider(names_from = RTS, values_from = eff)

kable(eff_wide, digits = 4)
```

DMU	FDH	CRS	VRS	IRS	DRS	FRH
Facility 1	1	1.0000	1.0000	1.0000	1.0000	1
Facility 2	1	1.0000	1.0000	1.0000	1.0000	1
Facility 3	1	1.0000	1.0000	1.0000	1.0000	1
Facility 4	1	1.0000	1.0000	1.0000	1.0000	1
Facility 5	1	0.9775	1.0000	1.0000	0.9775	1

DMU	FDH	CRS	VRS	IRS	DRS	FRH
Facility 6	1	0.8675	0.8963	0.8963	0.8675	1

Here, we took the results table and kept only the DMU name, RTS, and efficiency score. We then reshaped it into a wide format so each RTS (FDH, CRS, VRS, IRS, DRS, FRH) became its own column and each row represented a DMU. This type of table simply allowed us to compare each facility's efficiency across all assumptions side-by-side. Finally, we printed the table with kable(), rounding to four decimals, so scores of 1.000 (efficient) and any differences across RTS would be visible.

Peers & Lambdas by assumption

```
print_block <- function(tag) {
  df <- results %>% filter(RTS == tag) %>% arrange(desc(eff), DMU)
  kable(df, caption = paste0(tag, ": Efficiency, Peers, Lambdas"))
  cat("\n\n")
}
for (tag in names(rts_map)) print_block(tag)
```

In this section, we illustrated a small utility called print_block(tag), that displays a results table filtered to a given RTS, sorted by efficiency (descending) and DMU for ties, and formatted using kable() using a caption like "RTS: Efficiency, Peers, Lambdas". We explicitly added two extra line breaks to provide spacing before the remaining text after each table. We then looped through all of the RTS values that were found in rts_map and invoked print_block on each RTS value. This resulted in an output of six tables FDH, CRS, VRS, IRS, DRS, and FRH each displaying each DMU's efficiency, peer set, and nonzero lambda weights. The last argument results="asis" ensured that the intended kable table was rendered in document directly.

summary

```
efficient_counts <- results %>%
  group_by(RTS) %>% summarize(efficient_n = sum(abs(eff - 1) < 1e-8), .groups
= "drop")

eff_stats <- results %>%
  group_by(RTS) %>% summarize(avg_eff = mean(eff), median_eff = median(eff),
.groups = "drop")

kable(efficient_counts, caption = "Number of efficient DMUs (eff = 1)")
```

Number of efficient DMUs (eff = 1)

RTS	efficient_n
CRS	4
DRS	4
FDH	6
FRH	6

RTS	efficient_n
IRS	5
VRS	5

```
kable(eff_stats, digits = 4, caption = "Average and median efficiency")
```

Average and median efficiency

RTS	avg_eff	median_eff
CRS	0.9742	1
DRS	0.9742	1
FDH	1.0000	1
FRH	1.0000	1
IRS	0.9827	1
VRS	0.9827	1

We summarized the performance by RTS in two steps. First, we determined which DMUs are truly efficient for each technology (counting values that were within a tiny tolerance of 1 as inefficient) and saved those counts into efficient_counts. In the second step, we determined the average and median efficiency under each RTS in eff_stats. Finally, we printed both tables with kable(): the first table reports how many DMUs are on the frontier for each RTS. The second table implicitly rounds values to four decimals to indicate typical efficiency levels (mean and median) for easy FDH, CRS, VRS, IRS, DRS, and FRH comparisons.

The Number of efficient DMUs table indicates how restrictive each RTS is: higher counts usually are associated with more flexible frontiers (e.g., FDH or VRS), while lower counts are normally more evident under stricter assumptions (e.g., CRS or DRS). The Average and median efficiency table broadly depicts performance under each RTS. Higher averages/medians indicate the majority of DMUs sit relatively closer to the frontier; lower averages/medians suggest that the frontier is tougher or that there is greater variation in performance. If the average is well below the median, it signals that a few relatively low-scoring DMUs are dragging the mean downward (which will show left-tail variation). These summaries show both how many facilities achieve best-practice under each technology and how far, on average, the rest are from that benchmark.

Compare & Contrast

```
eps <- 1e-8
eff_tbl <- results

# Efficient sets by RTS
eff_sets <- eff_tbl %>%
  group_by(RTS) %>%
  summarize(eff_set = list(DMU[abs(eff - 1) < eps]), .groups = "drop")
```

```r
fmt <- function(x) if (length(x) == 0) "None" else paste(x, collapse = ", ")

# VRS vs CRS
vrs_set <- eff_sets$eff_set[eff_sets$RTS == "VRS"][[1]]
crs_set <- eff_sets$eff_set[eff_sets$RTS == "CRS"][[1]]
only_vrs <- setdiff(vrs_set, crs_set)

cat("**VRS vs CRS**\n\n")
```

VRS vs CRS

```r
cat("- Efficient under VRS: ", fmt(vrs_set), "\n")
```

- Efficient under VRS: Facility 1, Facility 2, Facility 3, Facility 4, Facility 5

```r
cat("- Efficient under CRS: ", fmt(crs_set), "\n")
```

- Efficient under CRS: Facility 1, Facility 2, Facility 3, Facility 4

```r
cat("- Efficient under VRS but not CRS: ", fmt(only_vrs), "\n\n")
```

- Efficient under VRS but not CRS: Facility 5

```r
# FDH vs VRS
fdh_set <- eff_sets$eff_set[eff_sets$RTS == "FDH"][[1]]
only_fdh <- setdiff(fdh_set, vrs_set)
only_vrs2 <- setdiff(vrs_set, fdh_set)
cat("**FDH vs VRS**\n\n")
```

FDH vs VRS

```r
cat("- Efficient under FDH only: ", fmt(only_fdh), "\n")
```

- Efficient under FDH only: Facility 6

```r
cat("- Efficient under VRS only: ", fmt(only_vrs2), "\n\n")
```

- Efficient under VRS only: None

```r
# Average efficiency ranking
avg_rank <- eff_tbl %>%
  group_by(RTS) %>%
  summarize(avg_eff = mean(eff), .groups = "drop") %>%
  arrange(desc(avg_eff))
cat("**Average efficiency by RTS (highest to lowest)**\n\n")
```

Average efficiency by RTS (highest to lowest)

```r
print(knitr::kable(avg_rank, digits = 4))
```

RTS	avg_eff
FDH	1.0000
FRH	1.0000
IRS	0.9827

RTS	avg_eff
VRS	0.9827
CRS	0.9742
DRS	0.9742

```
# Scale effects: IRS vs DRS
cmp <- eff_tbl %>% select(DMU, RTS, eff) %>% tidyr::pivot_wider(names_from =
RTS, values_from = eff)

if (all(c("IRS", "DRS") %in% names(cmp))) {
  cat("\n\n**Scale effects (IRS vs DRS)**\n\n")
  cat("- DMUs with IRS > DRS: ", fmt(cmp$DMU[cmp$IRS > cmp$DRS + eps]), "\n")
  cat("- DMUs with DRS > IRS: ", fmt(cmp$DMU[cmp$DRS > cmp$IRS + eps]), "\n")
}
```

Scale effects (IRS vs DRS)

- DMUs with IRS > DRS: Facility 5, Facility 6
- DMUs with DRS > IRS: None

VRS vs. CRS.

Under VRS (Variable Returns to Scale) and CRS (Constant Returns to Scale), the method has similarities and differences in regard to facility rankings. Under VRS condition, the set of efficient units includes: Facility 1, Facility 2, Facility 3, Facility 4, and Facility 5. Once we apply the more restrictive CRS assumption, the efficient (peer) set of units gets reduced to: Facility 1, Facility 2, Facility 3, and Facility 4. Thus, Facility 5 is the only unit that is efficient under VRS but not efficient under CRS. This means that there are scale conditions at play in Facility 5: when we assume variable returns to scale (VRS) it is at the frontier but, when we impose constant returns (CRS) it simply does not meet that frontier. In practical terms, Facility 5 appears efficient once we controlled for scale, which makes us consider the reason for difference in the case of CRS is because of a difference in scale not just differences in technical efficiency.

FDH vs. VRS.

The FDH (non-convex) frontier is the least restrictive and most flexible in evaluating efficiency. To some degree, based on the results, Facility 6 is efficient between the FDH frontier but not at the frontier under VRS. So, Facility 6 can “ride” along a non-convex (discrete/re-creation) hull, at least in observation, only when we impose the convex assumption with VRS.

Average efficiency rank by RTS.

The average-efficiency table adheres to the same hierarchy: FDH/FRH yield the highest average (as they are the least restrictive frontiers), VRS/IRS yield the average in the middle, and CRS/DRS yield the lowest average as they are the most restrictive frontiers. This hierarchy is not surprising: the more restrictions you add (constant or

directional scale) the more difficult it is to have efficiency = 1, so we would expect the averages to76 decline.

Scale effects (IRS vs. DRS).

Comparing scale effect lines indicate that Facility 5 and Facility 6 have higher efficiency in the IRS than DRS setting. This suggests that these two units reap more efficiency benefits from expanding scale than they do from contracting scale. No units accomplished the opposite. This implies that for these two units it might be reasonable in their actual operations (as long as it is practical) to expand operations to improve relative performance, whereas contraction would not help as much or at all.

Interpretation of Lamba :

The lambda (λ) values show how a DMU is benchmarked against its peer reference set. A higher λ on a peer means that peer contributes more to the composite "target" for the evaluated unit. In this write-up, we also treat the leftover portion as inefficiency to show how much improvement remains after accounting for learning from peers.

DMU 5 (peers: DMU 1 and DMU 4) :

Under CRS, DMU 5's row shows λ_1 = 0.26 (DMU 1) and λ_4 = 0.57 (DMU 4). The **inefficiency** is
$1 - 0.26 - 0.57 = 0.17$(i.e., **17%**).
Because DMU 5's projected point lies between DMU 1 and DMU 4 in the ratio **4:5** (based on outputs 20,000:25,000), we apportion the 0.17 gap proportionally:

- From **DMU 1**: $0.17 \times \frac{4}{9} = 0.075$
- From **DMU 4**: $0.17 \times \frac{5}{9} = 0.094$
  Interpretation: DMU 5 can reduce its 17% inefficiency by "learning" roughly **0.075** from DMU 1 and **0.094** from DMU 4. Because the target is not exactly midway, the learning is **not evenly split**

DMU 6 (peers: DMU 1 and DMU 2) :

Under CRS, λ_1 = 0.22 (DMU 1) and λ_2 = 0.71 (DMU 2), leaving **inefficiency** $1 - 0.22 - 0.71 = 0.07$(i.e., **7%**).
The line segment between DMU 1 and DMU 2 is divided in the ratio **14:15** (14,000:15,000), so we apportion the 0.07 gap accordingly:

- From **DMU 1**: $0.07 \times \frac{14}{29} = 0.033$

- From **DMU 2**: $0.07 \times \frac{15}{29} = 0.036$
  Interpretation: DMU 6 can close its 7% gap by "learning" approximately **0.033** from DMU 1 and **0.036** from DMU 2.

FDR Observation :

Efficient: Facility 1, Facility 2, Facility 3, Facility 4, Facility 5
Inefficient: Facility 6 = 88.24% (→ 11.76% inefficient)

Peers: Facility 6 benchmarks to Facility 2 (sole peer)

CRS Observation :

Efficient: Facility 1, Facility 2, Facility 4

Inefficient: Facility 3 = 87.93%, Facility 5 = 89.42%, Facility 6 = 70.48%

Peers: Facility 3 → Facility 1 & Facility 4 Facility 5 → Facility 1 & Facility 4 Facility 6 → Facility 1 & Facility 2

VRS Observation :

Efficient: Facility 1, Facility 2, Facility 3, Facility 4

Inefficient: Facility 5 = 92.39%, Facility 6 = 72.73%

Peers: Facility 5 → Facility 1 & Facility 4 Facility 6 → Facility 1 & Facility 2

IRS Observation :

Efficient: Facility 1, Facility 2, Facility 4

Inefficient: Facility 3 = 87.93%, Facility 5 = 92.39%, Facility 6 = 72.73%

Peers: Facility 3 → Facility 1 & Facility 4 Facility 5 → Facility 1 & Facility 4 Facility 6 → Facility 1 & Facility 2

DRS Observation :

Efficient: Facility 1, Facility 2, Facility 3, Facility 4

Inefficient: Facility 5 = 89.42%, Facility 6 = 70.48%

Peers: Facility 5 → Facility 1 & Facility 4 Facility 6 → Facility 1 & Facility 2

FRH Observation (free replicability hull / ADD):

Efficient: Facility 1, Facility 2, Facility 3, Facility 4, Facility 5

Inefficient: Facility 6 = 82.35% efficient (→ 17.65% inefficient)

Peers: Facility 6 benchmarks to Facility 2 (sole peer)

With respect to technologies, Facilities 1–4 each consistently represent frontiers; Facility 5 is scale-sensitive (efficient under VRS but not CRS); and Facility 6 shows the largest gap, generally calling upon Facility 1–2 (or just 2 under hull conjectures) as an antecedent neighbor for improvement efforts.