# Load required package

library(lpSolve)

# Function to generate a random DEA dataset

generate\_random\_dataset <- function(num\_dmus, num\_inputs, num\_outputs) {

inputs <- matrix(runif(num\_dmus \* num\_inputs, min = 1, max = 100),

nrow = num\_dmus, ncol = num\_inputs)

outputs <- matrix(runif(num\_dmus \* num\_outputs, min = 1, max = 100),

nrow = num\_dmus, ncol = num\_outputs)

dataset <- data.frame(

DMU = 1:num\_dmus,

inputs,

outputs

)

colnames(dataset) <- c("DMU",

paste0("Input\_", 1:num\_inputs),

paste0("Output\_", 1:num\_outputs))

return(dataset)

}

# Function to solve LP for efficiency

solve\_efficiency <- function(dmu\_idx, inputs, outputs, reference\_set) {

num\_inputs <- ncol(inputs)

num\_outputs <- ncol(outputs)

num\_reference <- length(reference\_set)

f.obj <- rep(0, num\_reference) # Objective: minimize weights

f.con <- rbind(

-t(inputs[reference\_set, , drop = FALSE]), # Input constraints

t(outputs[reference\_set, , drop = FALSE]) # Output constraints

)

f.dir <- c(rep("<=", num\_inputs), rep(">=", num\_outputs))

f.rhs <- c(-inputs[dmu\_idx, ], outputs[dmu\_idx, ])

# Solve LP

result <- lp("min", f.obj, f.con, f.dir, f.rhs)

if (result$status == 0) { # Successful optimization

list(efficiency = sum(result$solution), weights = result$solution)

} else {

list(efficiency = Inf, weights = NULL)

}

}

# DEA Iterative Algorithm

dea\_iteration <- function(dataset, max\_iterations = 100, subsample\_size = 4, efficiency\_threshold = 1.0) {

num\_dmus <- nrow(dataset)

num\_inputs <- grep("Input\_", colnames(dataset))

num\_outputs <- grep("Output\_", colnames(dataset))

inputs <- as.matrix(dataset[, num\_inputs])

outputs <- as.matrix(dataset[, num\_outputs])

efficiencies <- rep(0, num\_dmus)

reference\_set <- sample(1:num\_dmus, subsample\_size) # Initial subsample

iteration <- 0

while (iteration < max\_iterations) {

iteration <- iteration + 1

cat("Iteration", iteration, "\n")

reference\_updated <- FALSE

for (dmu\_idx in 1:num\_dmus) {

if (!(dmu\_idx %in% reference\_set)) {

result <- solve\_efficiency(dmu\_idx, inputs, outputs, reference\_set)

eff <- result$efficiency

if (eff < efficiency\_threshold) {

efficiencies[dmu\_idx] <- eff

reference\_set <- c(reference\_set, dmu\_idx)

reference\_updated <- TRUE

} else {

efficiencies[dmu\_idx] <- eff

}

}

}

if (!reference\_updated) {

cat("Convergence reached.\n")

break

}

}

list(efficiencies = efficiencies, reference\_set = reference\_set)

}

# Parameters for the random dataset

num\_dmus <- 10

num\_inputs <- 3

num\_outputs <- 2

# Generate random dataset

random\_dataset <- generate\_random\_dataset(num\_dmus, num\_inputs, num\_outputs)

# Run DEA iterative algorithm on the dataset

result <- dea\_iteration(random\_dataset)

# Display results

cat("Efficiencies of all DMUs:\n")

print(result$efficiencies)

cat("Final Reference Set:\n")

print(result$reference\_set)