allocation <- function(y) {
  # Uses a "greedy" algorithm to allocate beds one by one,
  # in a way that gives an allocation that is as similar as possible to y,
  # with no zero-allocations for the output x
  n = length(y)
  x = 0*y + 1
  Y = sum(y) - sum(x)
  for (t in 1:Y) {
    min = 100000
    best_i = 0
    for (i in 1:n) {
      c = x[i] - y[i]
      if (c<min) {
        min = c
        best_i = i
      }
    }
    x[best_i] = x[best_i] +1
  }
  return(x)
}

elf <- function(n, a)
{
  # Erlang's loss function
  b <- 1 ; i<- 0
  while((i <= i+1)<=n) b <- b+a/(i+b*a)
  b
}

average_beds_in_use <- function(beds_vector,load_vector) {
  # Uses the elf function to calculate the utilization for a vector of MAU sizes
  # coupled to a vector of loads.
  sum = 0
  for (i in 1:length(beds_vector)) {
    beds = beds_vector[i]
    load = load_vector[i]
    sum = sum + (1-elf(beds,load))*load
  }
  return(sum)
}

# Encoding the empirical distribution of MAU sizes:
empirical_MAU_sizes = c(72,34,25,16,15,14,13,12,12,12,11,11,10,10,10,9,9,9,8.77,8,8,8,7,7,7,6,6,6,6,6,5,9,5,
# demand_vector is the (non-integer) vector of demands (load) for the 206 MAUs, 
# derived from the empirical distribution and rescaled to make it sum to 658: 
demand_vector = empirical_MAU_sizes*658 / sum(empirical_MAU_sizes)

# Calculating the baseline allocation of beds as the closest integer-vector that 
# approximates demand (not allowing zeros): 
baseline_allocation = allocation(demand_vector)

# Computing the average number of beds in use in default scenario 
# with demand_vector as load and baseline_allocation as the bed allocation, and 
multiplying with 365 for full-year average: 
average_beds_in_use(baseline_allocation,demand_vector) * 365 
# 170695.8, which is 71.1% of 240000

# With double load: 
average_beds_in_use(baseline_allocation,demand_vector*2) *365
# 204261.3, which is still only 85.1% of 240000

# Calculating model 2-4 results for number of beds (and/or load) ranging from 658 to 
658*2:
beds = (658:(658*2))
demand_adjusted = 0*beds 
capacity_adjusted = 0*beds 
parallel_adjusted = 0*beds 
for (i in (1:length(beds))) {
    n = beds[i]
    demand_adjusted[i] = 
    average_beds_in_use(baseline_allocation,demand_vector*n/sum(demand_vector))
    non_integer_allocation = demand_vector*n/sum(demand_vector)
    x = allocation(non_integer_allocation)
    capacity_adjusted[i] = average_beds_in_use(x,demand_vector)
    parallel_adjusted[i] =
    average_beds_in_use(x,demand_vector*n/sum(demand_vector))
    print(c(n,demand_adjusted[i],capacity_adjusted[i],parallel_adjusted[i]))
    flush.console()
}

# Finding the minimum capacity that satisfies > 216000 bed days a year: 
i = 1
while (capacity_adjusted[i]*365 < 216000) i = i+1
beds[i]
# Plotting the results - Figure 3:
options(scipen = 99)
library(calibrate)

par(mar = c(6.5, 6.5, 0.5, 0.5), mgp = c(5, 1, 0))
plot((0:658)/6.58,demand_adjusted*365, type = "l", ylim=c(0,400000), yaxt="n",lty=3,
xlab= "Percentage increase in capacity and/or average demand", ylab=
"Patient-days per year")
lines((0:658)/6.58,capacity_adjusted*365, type = "l", lty=2)
lines((0:658)/6.58,parallel_adjusted*365, type = "l", lty=1)
axis(2,las=2)
abline("h"=240000)
text(15, 250000, "national policy goal", col = "black")
points(x = 0, y = 170696, type = "p", col = "black", pch = c(16,15))
points(x = 100, y = 204261, type = "p", col = "black", pch = c(16,15))
points(x = 57, y = 216000, type = "p", col = "black", pch = c(16,15))
points(x = 34, y = 240000, type = "p", col = "black", pch = c(16,15))
textxy(0, 170696, labs = 'S1 (baseline)', cex = 1, offset = 0.8)
textxy(100, 204261, labs = 'S2', cex = 1, offset = 0.6)
textxy(57, 216000, labs = 'S3', cex = 1, offset = 0.8)
textxy(34, 240000, labs = 'S4', cex = 1, offset = 0.8)
legend(0,75000, c("increased capacity and demand (scenario 4)","increased capacity only (scenario 3)","increased demand only (scenario 2)"),lty=c(1,2,3))