Mohd Khizer Hayat

2023-05-01

Case Study: Emergency Room (DES) Patients arrive at the Emergency Room following an unknown probability distribution (stream 1). They will be treated by either of two doctors. A proportion of the patients are classified as NIA (need immediate attention) and the rest as CW (can wait). NIA patients are given the highest priority, 3, see a doctor as soon as possible for 40 +(-) 30 minutes (stream 2), then have their priority reduced to 2 and wait until a doctor is free again, when they receive further treatment for 30 +(-) 20 minutes (stream 3) and are discharged. CW patients initially receive a priority of 1 and are treated (when their turn comes) for 15 +(-) 10 minutes (stream 4); their priority is then increased to 2, they wait again until a doctor is free, receive 10 +(-) 5 minutes (stream 5) of final treatment and are discharged. An important aspect of this system is that patients who have already seen the doctor once compete with newly arriving patients who need a doctor. As indicated, patients who have already seen the doctor once have a priority level of 2 (either increased from 1 to 2 or decreased from 3 to 2). Thus, there is one shared queue for the first treatment activity and the final treatment activity. In addition, we assume that the doctors are interchangeable. That is, it does not matter which of the two doctors performs the first or finnal treatment. Simulate for 20 days of continuous operation, 24 hours per day. Note, the inter-arrival time and type of 100 patients are collected (based on a randomly selected weekday data). The data is attached. 1. Analyze your results and explain your suggestions for reducing the waiting time of the patients. 2. What is the average ow-time for NIA and CW patients before or after applying suggestions. different suggestions. 3. Discuss the utilization of doctors before or after applying suggestions.

**STEP 01:** **Using Input Analysis to determine the patients arrival frequency.**

#Getting the libraries  
library(fitdistrplus)

## Warning: package 'fitdistrplus' was built under R version 4.2.3

## Loading required package: MASS

## Loading required package: survival

library(glogis)

## Warning: package 'glogis' was built under R version 4.2.3

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 4.2.2

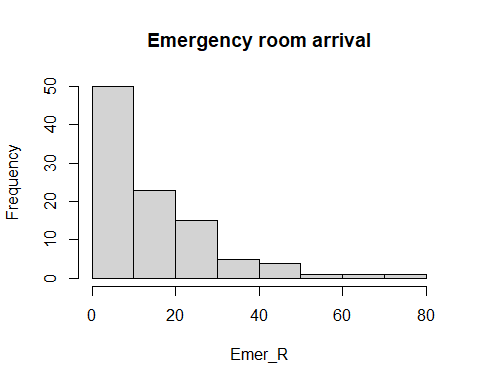
##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

#Reading the data from file  
EmergencyRoom = read.csv("Case5\_emergency-room.csv")  
  
head(EmergencyRoom)

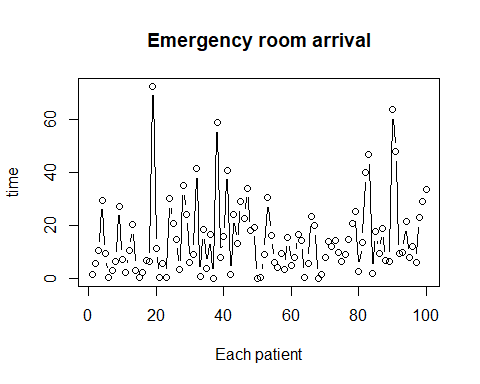
## interArrival type  
## 1 1.6 NIA  
## 2 5.6 CW  
## 3 10.5 CW  
## 4 29.5 CW  
## 5 9.4 CW  
## 6 0.6 NIA

Emer\_R = EmergencyRoom$interArrival  
  
# visualizing the data:  
  
hist(Emer\_R, main="Emergency room arrival")



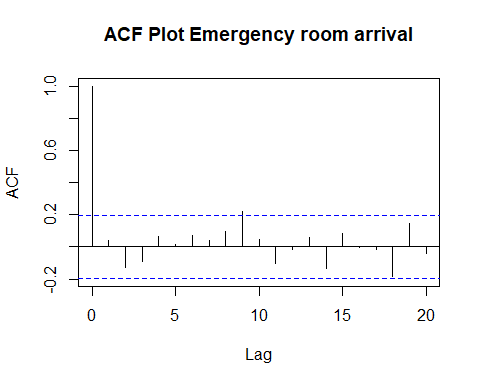
**The data distribution is positively (left) skewed.**

plot(Emer\_R, type = "b", main="Emergency room arrival", ylab = "time", xlab = "Each patient")



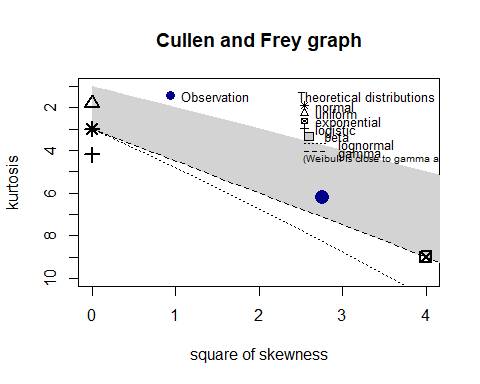
**There is no noticeable trend in the time-series plot.**

acf(Emer\_R, main = "ACF Plot Emergency room arrival")



**The autocorrelation plots shows no significant correlation for observation numbers (all lags are well within the confidence band without following a pattern). Therefore, the data appears to be stationary (IID).**

# Hypthesizing distributions  
descdist(Emer\_R, discrete = FALSE)



## summary statistics  
## ------  
## min: 0.1 max: 72.4   
## median: 10.2   
## mean: 15.077   
## estimated sd: 14.35904   
## estimated skewness: 1.661798   
## estimated kurtosis: 6.224021

**Possible distributions include the beta, lognormal, Weibull and Gamma distribution.**

**values must be in [0-1] to fit a beta distribution.**

# Checking for norm distribution fit  
fit.norm <- fitdist(Emer\_R, "norm")  
fit.norm$loglik

## [1] -407.8293

# Checking for exponential distribution fit  
fit.exp <- fitdist(Emer\_R, "exp")  
fit.exp$loglik

## [1] -371.317

# Checking for gamma distribution fit  
fit.gamma <- fitdist(Emer\_R, "gamma")  
fit.gamma$loglik

## [1] -371.2718

# Checking for Uniform distribution fit  
fit.unif <- fitdist(Emer\_R, "unif")  
fit.unif$loglik

## [1] -428.0824

# Checking for Log-normal distribution fit  
fit.lognorm <- fitdist(Emer\_R, "lnorm")  
fit.lognorm$loglik

## [1] -384.1864

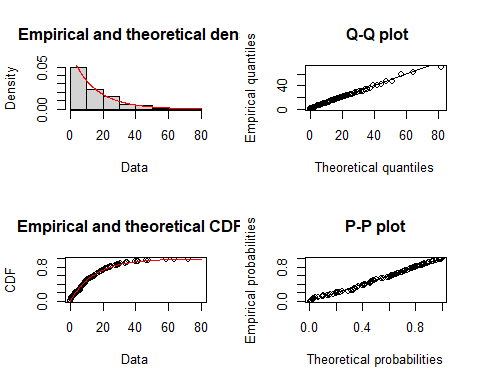
# Checking for weibull distribution fit  
fit.weibull <- fitdist(Emer\_R, "weibull")  
fit.weibull$loglik

## [1] -371.3167

# Checking for Logistic distribution fit  
fit.logist <- fitdist(Emer\_R, "logis")  
fit.logist$loglik

## [1] -401.8041

#gamma has the highest likelihood in the distribution  
  
#Plotting the distribution with largest log-likelihood  
plot(fit.gamma)



summary(fit.gamma)

## Fitting of the distribution ' gamma ' by maximum likelihood   
## Parameters :   
## estimate Std. Error  
## shape 0.96354630 0.11955644  
## rate 0.06388923 0.01025346  
## Loglikelihood: -371.2718 AIC: 746.5435 BIC: 751.7539   
## Correlation matrix:  
## shape rate  
## shape 1.0000000 0.7728234  
## rate 0.7728234 1.0000000

gofstat(fit.gamma)

## Goodness-of-fit statistics  
## 1-mle-gamma  
## Kolmogorov-Smirnov statistic 0.07804184  
## Cramer-von Mises statistic 0.05957576  
## Anderson-Darling statistic 0.41550172  
##   
## Goodness-of-fit criteria  
## 1-mle-gamma  
## Akaike's Information Criterion 746.5435  
## Bayesian Information Criterion 751.7539

**Gamma distribution: shape =0.963546, rate = 0.063889.**

#To determine the probability of patient arriving will be either CW or NIA.  
  
NIA\_count = nrow(EmergencyRoom[EmergencyRoom$type == "NIA", ])  
NIA\_count

## [1] 18

CW\_count = nrow(EmergencyRoom[EmergencyRoom$type == "CW", ])  
CW\_count

## [1] 82

**NIA patient is 18 out of 100 or 0.18 probability.**

**CW patient is 82 out of 100 or 0.82 probability.**

**STEP 02:** **Creating the Environment with trajectories.**

#Getting the libraries  
library(gridExtra)

## Warning: package 'gridExtra' was built under R version 4.2.3

library(simmer)

## Warning: package 'simmer' was built under R version 4.2.3

##   
## Attaching package: 'simmer'

## The following object is masked from 'package:MASS':  
##   
## select

library(simmer.plot)

## Warning: package 'simmer.plot' was built under R version 4.2.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.2.2

##   
## Attaching package: 'simmer.plot'

## The following objects are masked from 'package:simmer':  
##   
## get\_mon\_arrivals, get\_mon\_attributes, get\_mon\_resources

set.seed(123)  
   
envs = lapply(1:100, function(i) {   
 env = simmer("Emergency Room")  
  
patient = trajectory("patients' path") %>%  
   
 branch(function() sample(1:2, size=1, prob=c(0.18,0.82), replace = TRUE), continue=c(T,T),   
   
 trajectory("NIA") %>%  
 set\_attribute("stream 2", 3) %>%  
 set\_prioritization(c(3, 7, T)) %>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 10, 70)) %>%  
 release("doctor", 1) %>%  
   
 set\_attribute("stream 3", 2)%>%  
 set\_prioritization(c(2, 7, T)) %>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 10, 50)) %>%   
 release("doctor", 1),   
   
 trajectory("CW") %>%   
 set\_attribute("stream 4", 1)%>%  
 set\_prioritization(c(1, 7, T))%>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 5, 25)) %>%  
 release("doctor", 1) %>%  
   
 set\_attribute("stream 5", 2)%>%  
 set\_prioritization(c(2, 7, T)) %>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 5, 15)) %>%   
 release("doctor", 1)  
 )  
  
env%>%  
 add\_resource("doctor", 2) %>%   
   
 add\_generator("patient", patient, function() rgamma(1, 0.963, 0.0639), mon = 2)  
env %>%  
 run(600)  
})  
  
#plot(patient, verbose = T)

**STEP 03:** **Determining average waiting and flow time for all patients.**

#Getting the environment  
res <- get\_mon\_resources(envs)  
arrivals <- get\_mon\_arrivals(envs)  
  
#Determining the average waiting time for all patients  
wait\_time <- (arrivals$end\_time - arrivals$start\_time) - arrivals$activity\_time  
mean(wait\_time, na.rm=T)

## [1] 55.37779

#Determining the average flow time for all patients  
flow\_time = arrivals$end\_time - arrivals$start\_time  
mean(flow\_time, na.rm = T)

## [1] 88.72746

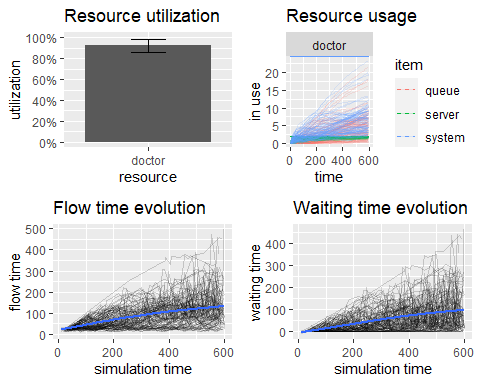
**Average waiting time for all patients is 55.38 minutes.**

**Average flow time for all patients is 88.73 minutes.**

**STEP 04:** **Visualization**

# Utilization plot   
util = plot(res, metric = "utilization")  
# usage plot  
usage = plot(res, metric = "usage")  
# Flow time plot   
flow = plot(arrivals, metric = "flow\_time")  
# waiting time plot  
wait = plot(arrivals, metric = "waiting\_time")  
grid.arrange (util, usage, flow, wait)

## `geom\_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'  
## `geom\_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'



**STEP 05:** **Determining the average wait and flow time for each type of patient.**

x1 <- get\_mon\_arrivals(envs)   
x2 <- get\_mon\_attributes(envs)   
  
all <- merge(x1, x2, by=c("name", "replication"), all = T)   
  
head(all)

## name replication start\_time end\_time activity\_time finished time  
## 1 patient0 1 5.877917 39.24065 33.36273 TRUE 28.726298  
## 2 patient0 1 5.877917 39.24065 33.36273 TRUE 5.877917  
## 3 patient0 2 1.124310 26.06151 17.77351 TRUE 9.182055  
## 4 patient0 2 1.124310 26.06151 17.77351 TRUE 1.124310  
## 5 patient0 3 2.285176 32.51367 30.22849 TRUE 26.713937  
## 6 patient0 3 2.285176 32.51367 30.22849 TRUE 2.285176  
## key value  
## 1 stream 5 2  
## 2 stream 4 1  
## 3 stream 5 2  
## 4 stream 4 1  
## 5 stream 5 2  
## 6 stream 4 1

# Seperating the data into NIA and CW patients  
NIA\_subset <- subset(all, all$key == "stream 2"|all$key == "stream 3")  
CW\_subset <- subset(all, all$key == "stream 4"|all$key == "stream 5")  
  
# Getting the flow-time per category of patient  
NIA.flowTime = (NIA\_subset$end\_time-NIA\_subset$start\_time)   
CW.flowTime = (CW\_subset$end\_time-CW\_subset$start\_time)  
  
# Average total time spent in the system per patient per category  
mean(NIA.flowTime, na.rm = T)

## [1] 100.9012

mean(CW.flowTime, na.rm = T)

## [1] 85.73382

**Average flow time for NIA Patients is 100.90 minutes.**

**Average flow time for CW Patients is 85.73 minutes.**

# Finding the average waiting time per patient per type  
  
NIA.waitTime = (NIA\_subset$end\_time-NIA\_subset$start\_time) - NIA\_subset$activity\_time   
CW.waitTime = (CW\_subset$end\_time-CW\_subset$start\_time)- CW\_subset$activity\_time   
  
mean(NIA.waitTime, na.rm = T)

## [1] 31.77745

mean(CW.waitTime, na.rm = T)

## [1] 61.18132

**Average wait time for NIA Patients is 31.77 minutes.**

**Average wait time for CW Patients is 61.18 minutes.**

**STEP 06:**

**Suggestion: Increasing the resources to 3 doctors for reducing the waiting time of the patients.**

set.seed(123)  
  
envs2 <- lapply(1:100, function(i) {   
 env2 = simmer("Emergency Room")  
  
patient2 <- trajectory("patients' path") %>%  
 branch(function() sample(1:2, size=1, replace = T, prob=c(0.18,0.82)), continue=c(T,T),   
   
 trajectory("NIA") %>%  
 set\_attribute("stream 2", 3)%>%  
 set\_prioritization(c(3, 7, T))%>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 10, 70)) %>%  
 release("doctor", 1) %>%  
   
 set\_attribute("stream 3", 2)%>%  
 set\_prioritization(c(3, 7, T)) %>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 10, 50)) %>%   
 release("doctor", 1),   
   
 trajectory("CW") %>%   
 set\_attribute("stream 4", 1)%>%  
 set\_prioritization(c(1, 7, T))%>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 5, 25)) %>%  
 release("doctor", 1) %>%  
   
 set\_attribute("stream 5", 2)%>%  
 set\_prioritization(c(2, 7, T)) %>%  
 seize("doctor", 1) %>%   
 timeout(function() runif(1, 5, 15)) %>%   
 release("doctor", 1)  
 )  
  
env2%>%  
 add\_resource("doctor", 3) %>%   
  
 add\_generator("patient2", patient2, function() rgamma(1, 0.963, 0.0639), mon = 2)  
env2 %>%   
 run(600)  
})  
  
  
#Plotting the path  
#plot(patient2, verbose = T)  
  
# Getting the environment  
res2 <- get\_mon\_resources(envs2)  
arrivals2 <- get\_mon\_arrivals(envs2)  
  
# Determining the average waiting time for all patients  
wait\_time2 <- (arrivals2$end\_time - arrivals2$start\_time) - arrivals2$activity\_time  
mean(wait\_time2, na.rm=T)

## [1] 10.72774

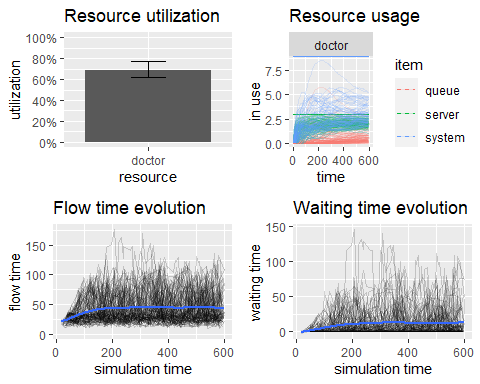
# Determining the average flow time for all patients  
flow\_time2 = arrivals2$end\_time - arrivals2$start\_time  
mean(flow\_time2, na.rm = T)

## [1] 42.43416

Average waiting time for all patient is 10.72 minutes. Average flow time for all patients is 42.43 minutes.

# Utilization plot   
util2 = plot(res2, metric = "utilization")  
# Usage plot  
usage2 = plot(res2, metric = "usage")  
# Flow time plot   
flow2 = plot(arrivals2, metric = "flow\_time")  
# waiting time plot  
wait2 = plot(arrivals2, metric = "waiting\_time")  
grid.arrange (util2, usage2, flow2, wait2)

## `geom\_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'  
## `geom\_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'



# Finding the average flow time and wait time for each type of patients:   
x3 <- get\_mon\_arrivals(envs2)   
x4 <- get\_mon\_attributes(envs2)   
  
all2 <- merge(x3, x4, by=c("name", "replication"), all = T)   
  
head(all2)

## name replication start\_time end\_time activity\_time finished time  
## 1 patient20 1 5.877917 39.24065 33.36273 TRUE 5.877917  
## 2 patient20 1 5.877917 39.24065 33.36273 TRUE 28.726298  
## 3 patient20 2 3.364896 37.53268 34.16779 TRUE 27.368236  
## 4 patient20 2 3.364896 37.53268 34.16779 TRUE 3.364896  
## 5 patient20 3 72.189614 92.97110 20.78149 TRUE 82.931617  
## 6 patient20 3 72.189614 92.97110 20.78149 TRUE 72.189614  
## key value  
## 1 stream 4 1  
## 2 stream 5 2  
## 3 stream 5 2  
## 4 stream 4 1  
## 5 stream 5 2  
## 6 stream 4 1

# Seperating the data into NIA and CW patients  
NIA\_subset2 <- subset(all2, all2$key == "stream 2"|all2$key == "stream 3")  
CW\_subset2 <- subset(all2, all2$key == "stream 4"|all2$key == "stream 5")  
  
# Getting the flowtime per category of patient  
NIA.flowTime2 = (NIA\_subset2$end\_time-NIA\_subset2$start\_time)   
CW.flowTime2 = (CW\_subset2$end\_time-CW\_subset2$start\_time)  
  
# Average total time spent in the system per patient per category  
mean(NIA.flowTime2, na.rm = T)

## [1] 76.10729

mean(CW.flowTime2, na.rm = T)

## [1] 36.43075

**Average flow time for NIA Patients is 76.10 minutes.**

**Average flow time for CW Patients is 36.43 minutes.**

#Finding the average waiting time per patient per type  
  
NIA.waitTime2 = (NIA\_subset2$end\_time-NIA\_subset2$start\_time) - NIA\_subset2$activity\_time   
CW.waitTime2 = (CW\_subset2$end\_time-CW\_subset2$start\_time)- CW\_subset2$activity\_time   
  
mean(NIA.waitTime2, na.rm = T)

## [1] 5.464956

mean(CW.waitTime2, na.rm = T)

## [1] 11.66601

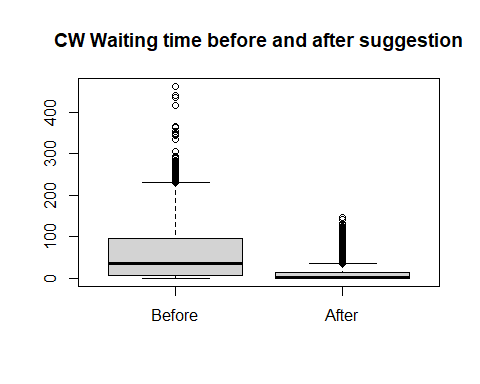
**Average wait time for NIA Patients is 5.46 minutes.**

**Average wait time for CW Patients is 11.67 minutes.**

**STEP 07:**

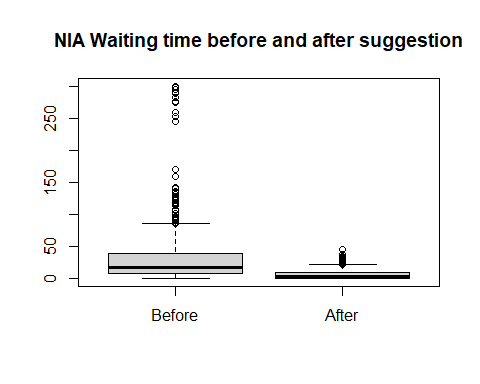
**Comparing the CW waiting time before and after suggestion**

boxplot(CW.waitTime,CW.waitTime2, names=c("Before","After"), main = "CW Waiting time before and after suggestion" )



**Comparing the NIA waiting time before and after suggestion**

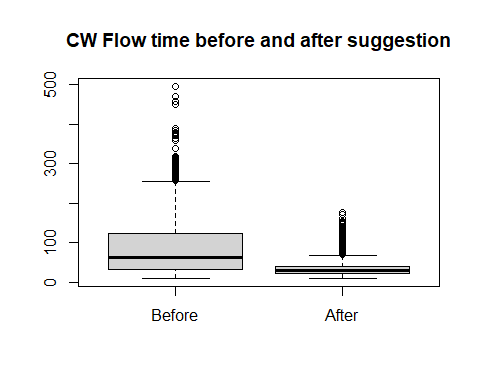
boxplot(NIA.waitTime, NIA.waitTime2, names=c("Before", "After"), main = "NIA Waiting time before and after suggestion" )



**STEP 08:**

**Comparing the CW flow time before and after suggestion**

boxplot(CW.flowTime,CW.flowTime2, names=c("Before","After"), main = "CW Flow time before and after suggestion" )



**Comparing the NIA flow time before and after suggestion**

boxplot(NIA.flowTime, NIA.flowTime2, names=c("Before", "After"), main = "NIA Flow time before and after suggestion" )

