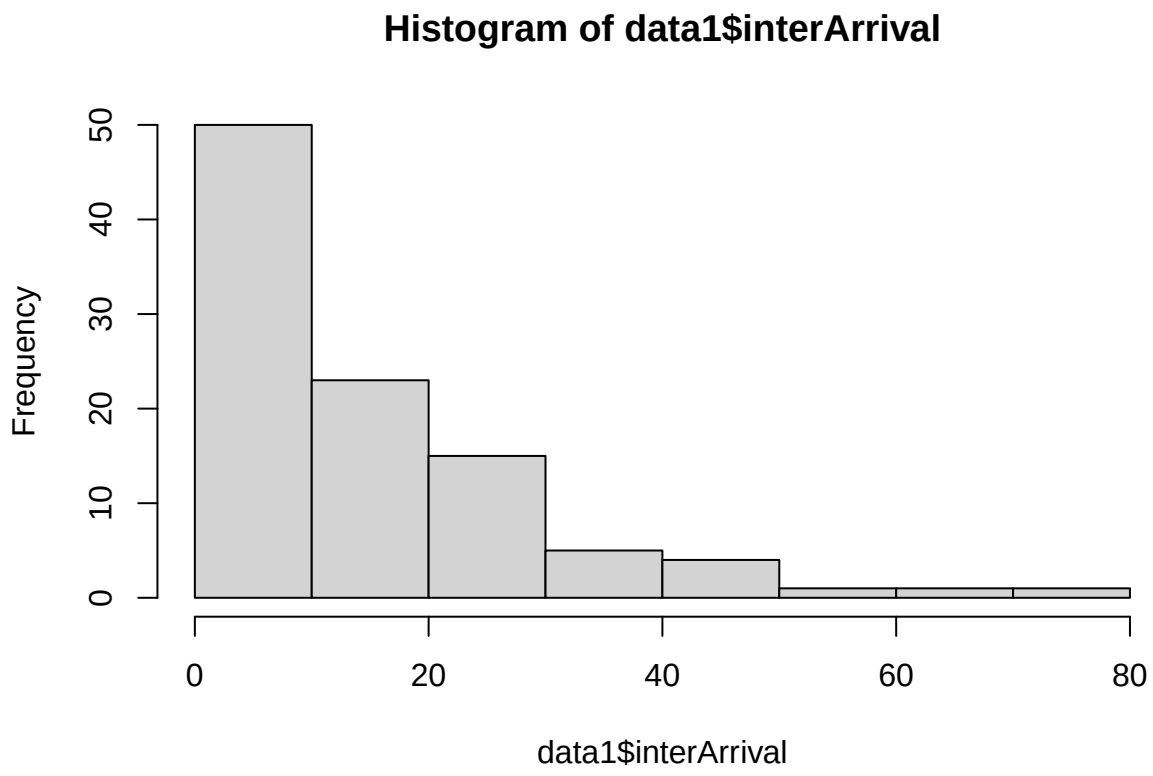
Emergency Room

Nahid Ferdous

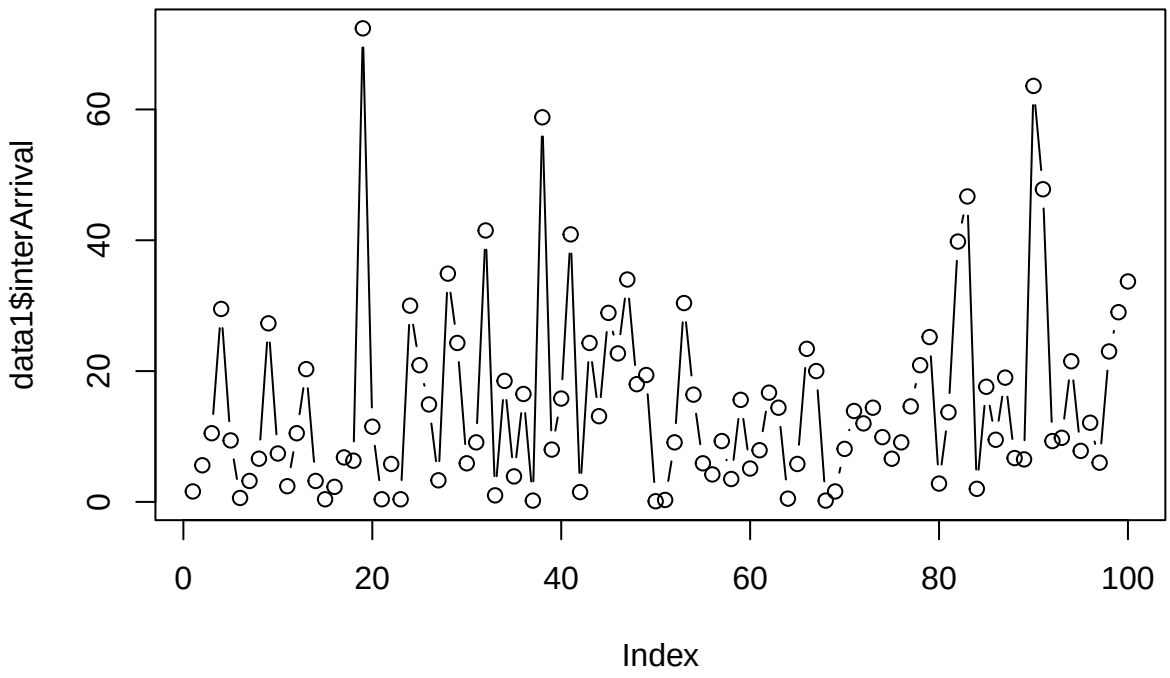
2024-05-04

Identyfy the interArrival distribution

```
data1 <- read.csv("Case5_emergency-room.csv")

hist(data1$interArrival)
```

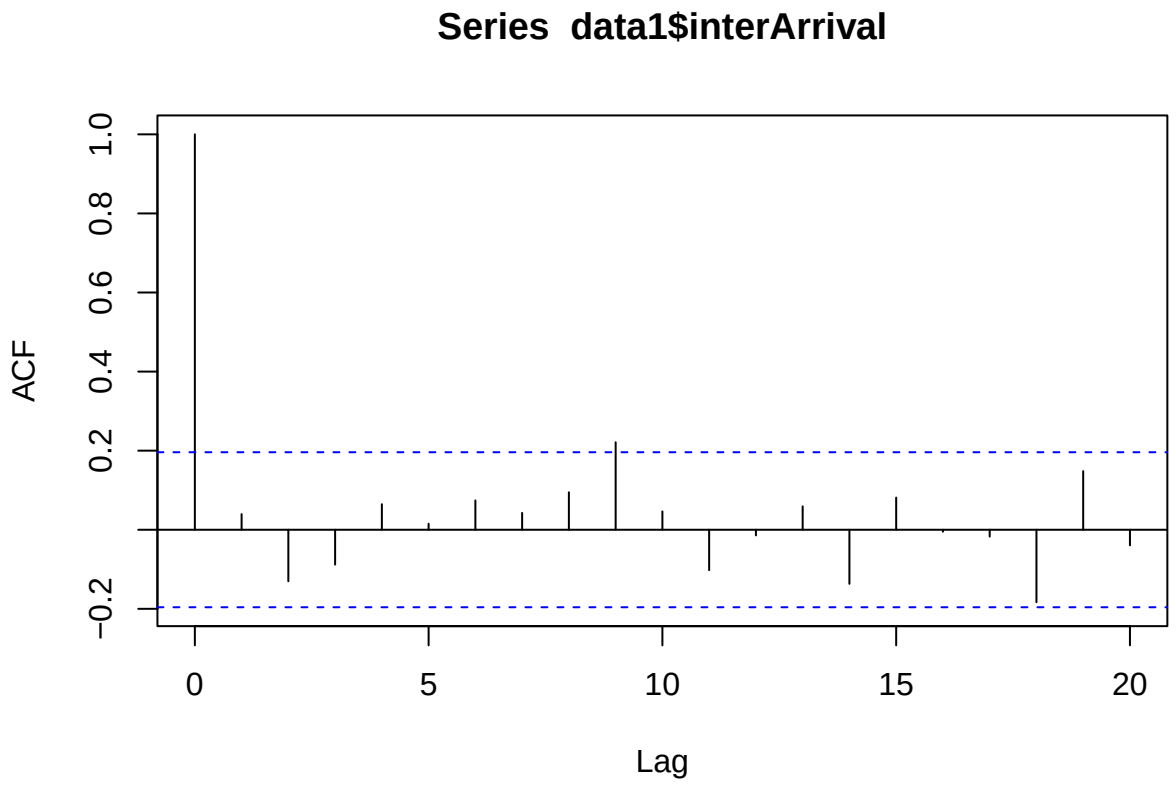



```
plot(data1$interArrival, type = "b")
```

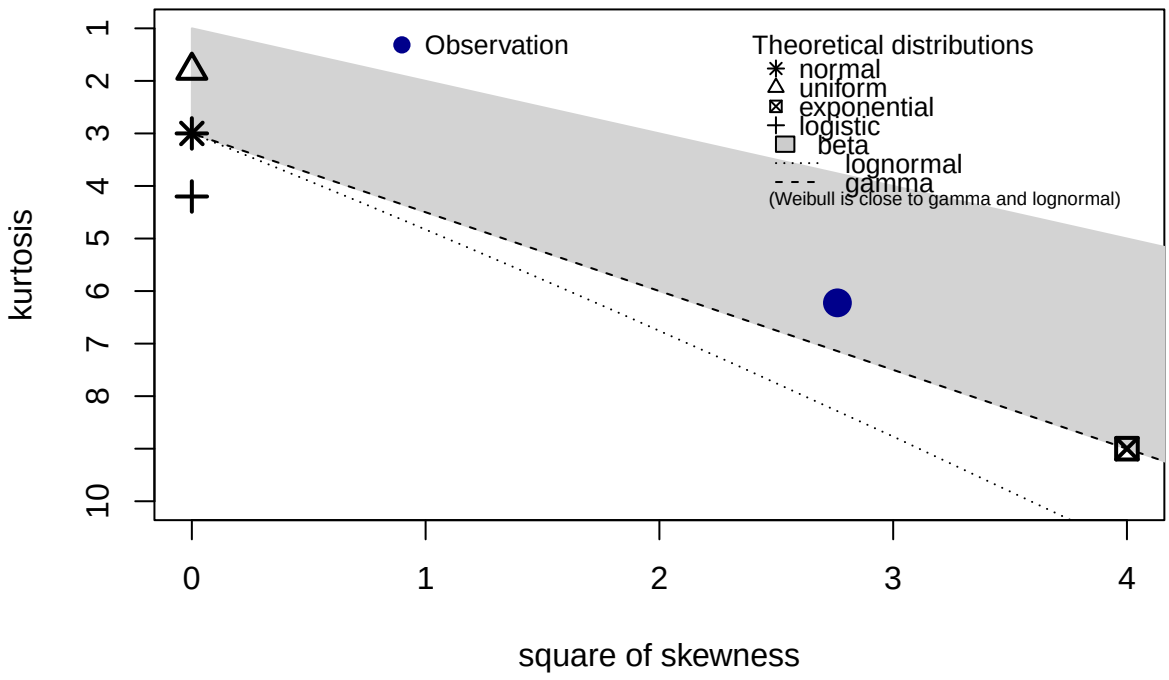



```
acf(data1$interArrival)

library(fitdistrplus)
```

```
## Loading required package: MASS

## Loading required package: survival
```




```
descdist(data1$interArrival, 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
```

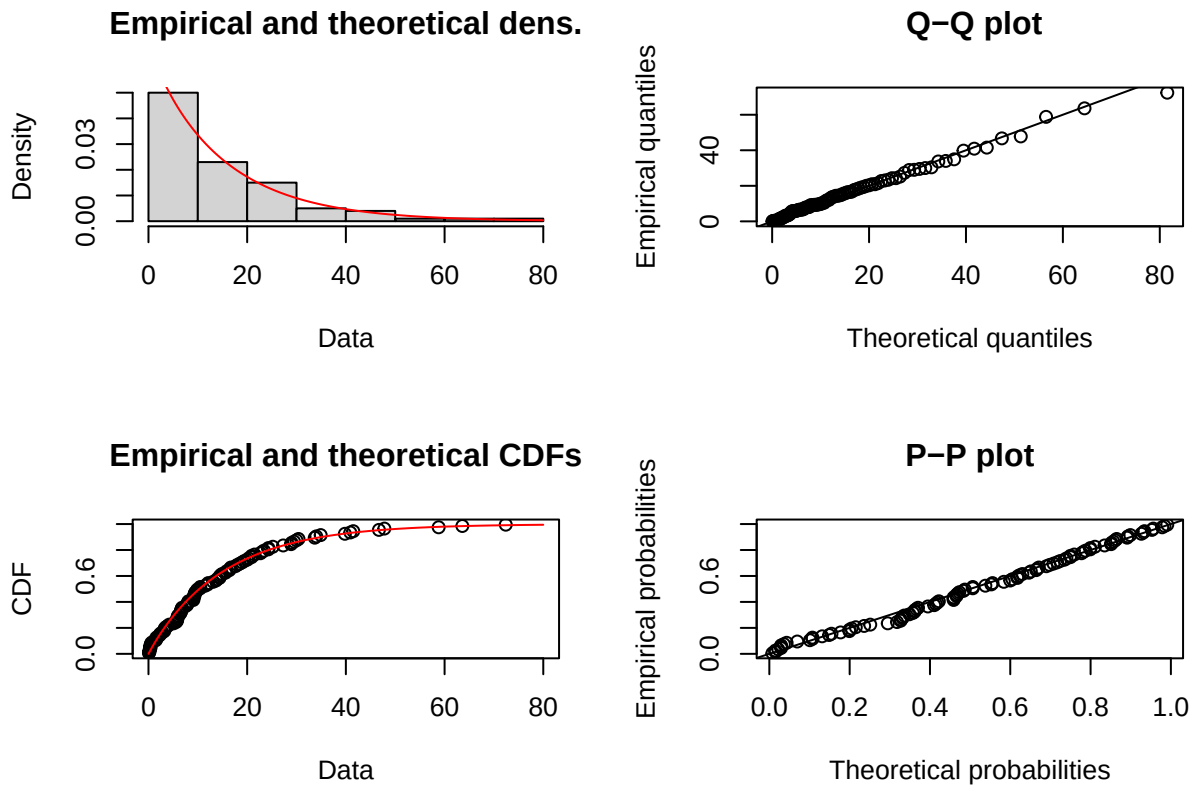
```
fit.gamma <- fitdist(data1$interArrival, "gamma")
summary(fit.gamma)
```

```
## 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
```

```
plot(fit.gamma)
```



Calculate the frequency of each category in the 'type' variable

```
type_counts <- table(data1$type)

# Calculate the proportions of each category
type_proportions <- prop.table(type_counts)
```

```r
# Print the proportions
print(type_proportions)
```

```
##
##   CW  NIA
## 0.82 0.18
```

```r
library(simmer)
```

```
## Warning: package 'simmer' was built under R version 4.3.1
```

```
##
## Attaching package: 'simmer'
```

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

```r
set.seed(123)
envs <- lapply(1:20, function(i) {
  env <- simmer("Emergency Room") %>%
    add_resource("doctor", 2)

  patient <- trajectory("patient path") %>%
    branch(
      function() sample(c(1, 2), size = 1, replace = TRUE, prob = c(0.82, 0.18)), continue = c(TRUE, TRU
      trajectory("NIA") %>%
        set_attribute("priority", 3) %>%
        set_prioritization(c(5, 7, TRUE)) %>%
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 70)) %>%
        release("doctor", 1) %>%

        set_attribute("priority", 2) %>%
        set_prioritization(c(4, 7, TRUE)) %>%
        seize("doctor", 1) %>%
        timeout(function() runif(1, 10, 50)) %>%
        release("doctor", 1),

      trajectory("CW") %>%
        set_attribute("priority", 1) %>%
        set_prioritization(c(3, 7, TRUE)) %>%
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 25)) %>%
        release("doctor", 1) %>%

        set_attribute("priority", 2) %>%
        set_prioritization(c(4, 7, TRUE)) %>%
        seize("doctor", 1) %>%
        timeout(function() runif(1, 5, 15)) %>%
        release("doctor", 1)
```

```
    )
  env %>%
    add_generator("patient", patient, function() rgamma(1, shape = 0.96354630, rate = 0.06388), mon = 2)
  env %>%
    run(1440)
})
```

01 Average of discharged patients per replication

```
patientAttr <- get_mon_attributes(envs)
colMeans(table(patientAttr$replication, patientAttr$value))
```

```
##     1     2     3
## 17.95 66.90 80.50
```

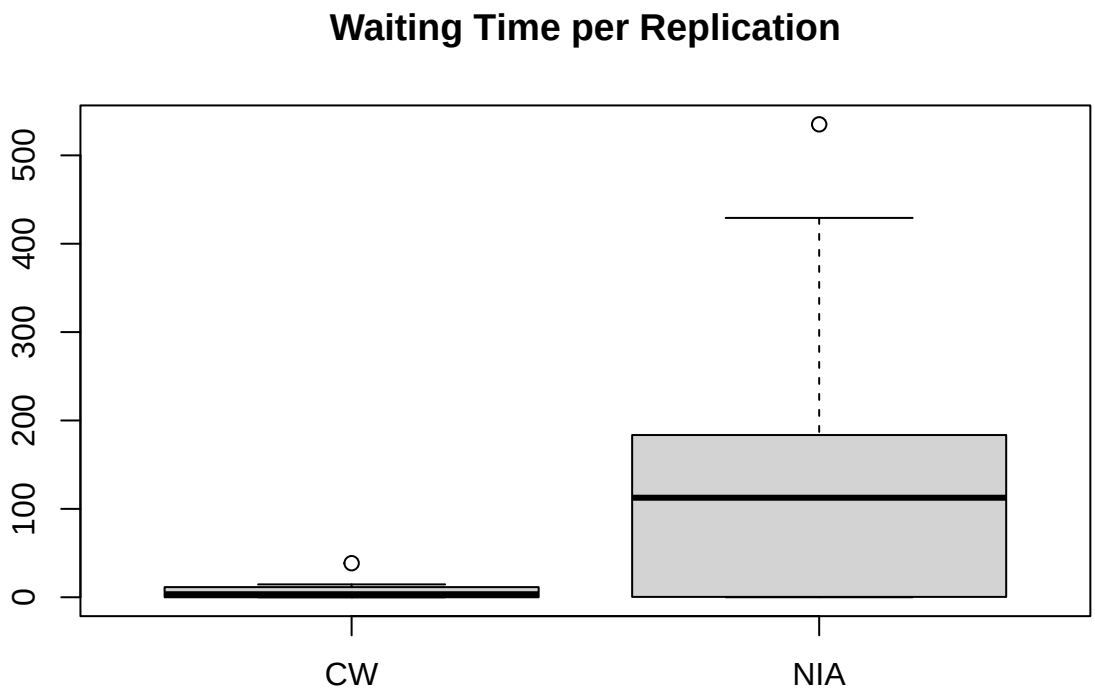
```
x1 <- get_mon_arrivals(envs)
x2<- get_mon_attributes(envs)

all <- merge(x1, x2, by= c("name", "replication"), all= T)
priority1 <- na.omit(subset(all, all$value ==1 ))
priority2 <- na.omit(subset(all, all$value ==2))
priority3 <- na.omit(subset(all, all$value ==3))
priority1.waiting <- (priority1$end_time - priority1$start_time) - priority1$activity_time
priority3.waiting <- (priority3$end_time - priority3$start_time) - priority3$activity_time
mean(priority1.waiting)
```

```
## [1] 10.05917
```

```
mean(priority3.waiting)
```

```
## [1] 215.9932
```

```
# Average waiting time per replication for each type
priority1.waiting.rep <- aggregate(priority1.waiting, by = list(priority1$replication), mean)
priority3.waiting.rep <- aggregate(priority3.waiting, by = list(priority3$replication), mean)

boxplot(priority1.waiting.rep$x, priority3.waiting.rep$x, names = c("CW", "NIA"), main = "Waiting Time
```



02 Average flow time of each type of patient

```
priority1.flowTime <- (priority1$end_time - priority1$start_time)
priority2.flowTime <- (priority2$end_time - priority2$start_time)
priority3.flowTime <- (priority3$end_time - priority3$start_time)

mean(priority1.flowTime)
```

```
## [1] 33.69845
```

```
mean(priority3.flowTime)
```

```
## [1] 281.1561
```

3 utilization

```
library(simmer.plot)
```

```
## Loading required package: ggplot2
```

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

```
## The following objects are masked from 'package:simmer':
##
##     get_mon_arrivals, get_mon_attributes, get_mon_resources
```

```
library(gridExtra)

resources <- get_mon_resources(envs)
arrivals <- get_mon_arrivals(envs)
p1 <- plot(resources, metric = "utilization")
p2 <- plot(resources, metric = "usage")
p3 <- plot(arrivals, metric = "activity_time")
p4 <- plot(arrivals, metric = "waiting_time")
grid.arrange(p1,p2,p3,p4)
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'

## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```