

Análise da fila de atendimento (restrita ao horário de pico 20h)

Carga e Transformação

```
#le dados de entrada
data <- read.csv("data.csv", stringsAsFactors = F, sep = ";")

#dias com ocupação acima de 90% durante os dois intervalos estudados
highDays <- c("16-02-18", "16-02-19", "16-02-22", "16-02-23", "16-02-24", "16-02-25", "16-03-08", "16-03-09")
#data <- head(data, 10)
#nrow(data)

#transforma data/hora de entrada em timestamp
data$arrivalTimestamp <- as.POSIXct(strptime(with(data, paste(Data, Hora.Chegada)), "%Y/%m/%d %H:%M:%S"))
data$servStartTimestamp <- as.POSIXct(strptime(with(data, paste(Data, Hora.Chamada)), "%Y/%m/%d %H:%M:%S"))

#calcula o turno
data$turno <- as.factor(floor(as.numeric(format(data$arrivalTimestamp, "%H"))/6))

data$prefer <- substr(data$Chamada, 2, 2)=="P"

#transforma Guiche em variavel categórica
data$Guiche <- as.factor(data$Guiche)

#extraí tipo de atendimento
data$Tipo <- as.factor(
  substr(data$Chamada, 1, attr(regexr("[A-Z]{1,2}", data$Chamada), "match.length")))
# table(data$Tipo)
# midpoints <- barplot(as.data.frame(table(data$Tipo))$Freq,
#   names.arg=as.data.frame(table(data$Tipo))$Var1)
# text(midpoints, 200, labels=as.data.frame(table(data$Tipo))$Freq)

#calcula o tempo na fila
data$waitingTime <- data$servStartTimestamp - data$arrivalTimestamp

#calcula o tempo de atendimento
data <- data[with(data, order(Guiche, servStartTimestamp)), ]
data <- dplyr::arrange(data, .(format(servStartTimestamp, "%Y/%m/%d"), Guiche), mutate, servDuration = c(as.numeric(servStartTimestamp) - as.numeric(arrivalTimestamp)))
data <- data[,-1]

#ordena por hora de chegada para cálculo da diferença de chegada
data <- data[order(data$arrivalTimestamp),]
data$timediff <- c(Inf, diff(data$arrivalTimestamp))

#filtra apenas os registros com tempo de serviço válido
data <- data[!is.na(data$servDuration), ]

#todos os dados transformados
write.csv(data, file="dataFull.csv", row.names = F)
```

```
#filtra apenas as chegadas ocorridas entre 20:00:00 e 20:59:59
data <- data[format(data$arrivalTimestamp, "%H")=="20", ]
write.csv(data, file="data20.csv", row.names = F)
```

Percentual de atendimentos preferenciais: 0.0158932

Distribuição de chegada

```
chegadas1 <-
  ggplot(data, aes(timediff, colour = format(arrivalTimestamp, "%H"))) +
  geom_freqpoly(aes(y = (..count..)/sum(..count..)), binwidth = 30) +
  xlim(0, 600) +
  ylab("Frequência Percentual") +
  xlab("Intervalo em segundos") +
  labs(colour = "Hora") +
  scale_y_continuous(labels = percent_format())
ggsave(filename = "chegadas120.png", plot = chegadas1)
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 9 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_path).
```

```
write.csv(data[, "timediff"], file="chegadasFiltro20.csv")
write.csv(data[format(data$arrivalTimestamp, "%y-%m-%d") %in% highDays, "timediff"], file="chegadasFiltro20.csv")
```

```
chegadas2 <- ggplot(data, aes(timediff)) +
  geom_histogram(aes(y = (..count..)/sum(..count..)), binwidth = 30) +
  #geom_freqpoly(aes(y = (..count..)/sum(..count..)), binwidth = 30) +
  xlim(0, 600) +
  ylab("Frequência Percentual") +
  xlab("Intervalo em segundos") +
  scale_y_continuous(labels = percent_format())
ggsave(filename = "chegadas220.png", plot = chegadas2)
```

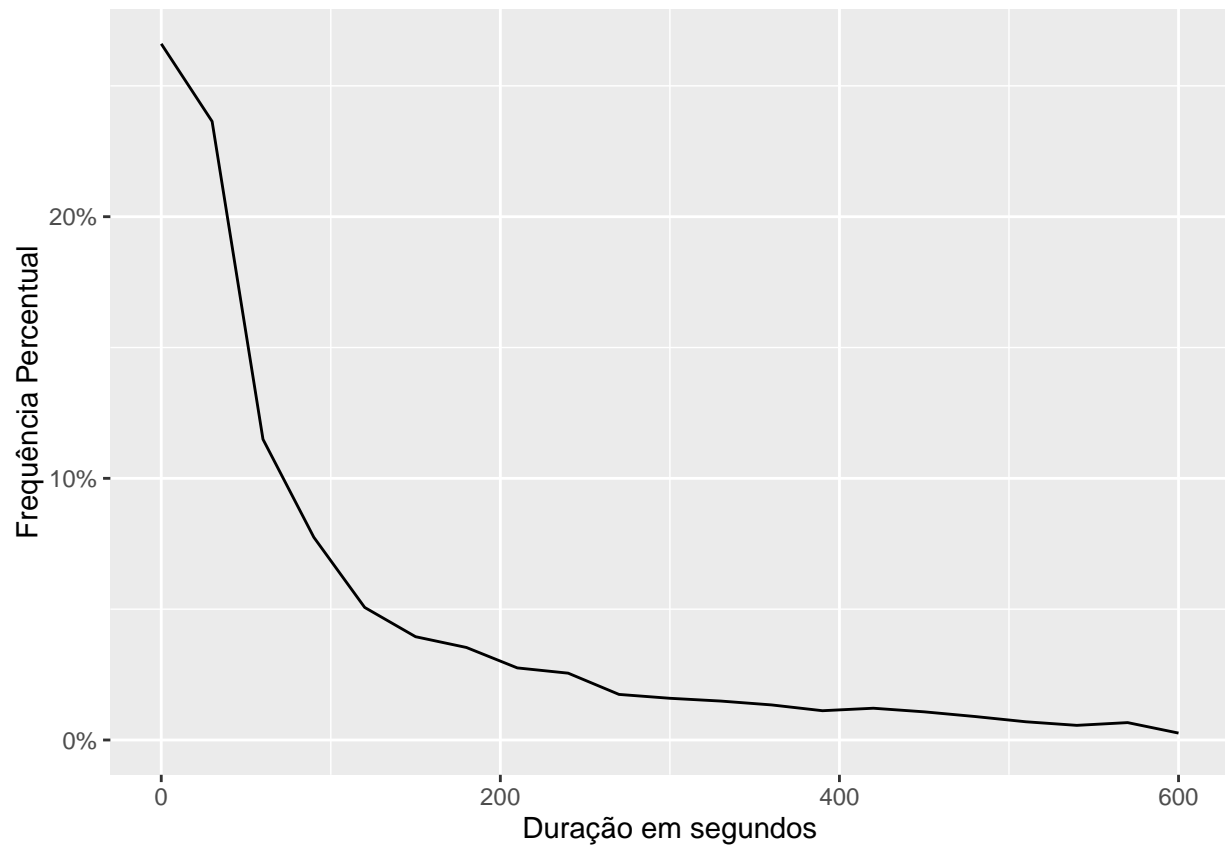
```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 9 rows containing non-finite values (stat_bin).
```

```
#-> para os dados gerais, EasyFit retornou Fatigue Life (Birnbau-Saunders) Distribution com shape=1.94
test <- rbisa(10000, scale=50.405, shape=1.9434)
ggplot(data.frame(test), aes(test)) +
  geom_freqpoly(aes(y = (..count..)/sum(..count..)), binwidth = 30) +
  xlim(0, 600) +
  ylab("Frequência Percentual") +
  xlab("Duração em segundos") +
  scale_y_continuous(labels = percent_format())
```

```
## Warning: Removed 525 rows containing non-finite values (stat_bin).
```

```
## Warning: Removed 2 rows containing missing values (geom_path).
```



Sumarização e visualização

Chegadas por hora

```
#sumarização de chegadas por hora
chegadas.hora.dia <- ddply(data, .(hora=format(arrivalTimestamp, "%H"), dia=format(arrivalTimestamp, "%Y-%m-%d")), summarize, mediaDiff=mean(meanTimeDiff), n)

#visualiza quantidade de chegadas por hora
chegadas3 <-
  ggplot(chegadas.hora.dia.medio, aes(x=factor(hora), y=mediaAtendDia)) +
  geom_bar(stat = "identity") +
  ylab("Quantidade de média novos clientes") +
  xlab("Horas do dia")
ggsave(filename = "chegadas320.png", plot = chegadas3)
```

```
## Saving 6.5 x 4.5 in image
```

```
#visualiza intervalo medio de chegada por hora
chegadas4 <-
  ggplot(chegadas.hora.dia.medio, aes(x=factor(hora), y=as.POSIXct(mediaDiff, origin = "1970-01-01", tz=
  geom_bar(stat = "identity") +
  ylab("Intervalo médio entre chegadas (hh:mm:ss)") +
  xlab("Horas do dia") +
  scale_y_datetime(labels = date_format("%H:%M:%S"))
ggsave(filename = "chegadas420.png", plot = chegadas4)
```

```
## Saving 6.5 x 4.5 in image
```

```
quantile(floor(data$waitingTime/60), probs = seq(0, 1, 0.1), na.rm = T)
```

```
## Time differences in secs
##   0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
##    0    0    0    1    5   10   13   17   23   28   68
```

```
#percentual de registros com tempo de espera < 1min
sum(data$waitingTime<=60, na.rm = T)/sum(!is.na(data$waitingTime))*100
```

```
## [1] 24.66624
```

```
#percentual de registros com tempo de espera < 30min
sum(data$waitingTime<=60*30, na.rm = T)/sum(!is.na(data$waitingTime))*100
```

```
## [1] 90.52765
```

```
#indica atendimentos imediatos
data$atendImediato <- data$waitingTime <= 60

tempo_fila_1 <-
  ggplot(data[!data$atendImediato,], aes(floor(waitingTime/60))) +
  geom_histogram(aes(y = (..count..)/sum(..count..)), binwidth = 1, boundary = 1) +
  xlim(0, 60) +
  ylab("Frequência Percentual") +
  xlab("Duração em minutos") +
  scale_y_continuous(labels = percent_format())
ggsave(filename = "tempo_fila_120.png", plot = tempo_fila_1)
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 3 rows containing non-finite values (stat_bin).
```

```
#atendimentos sem duração informada e com duração menor que 1min
#hist(data[data$servDuration < 60, "servDuration"])
#sum(data$servDuration < 60, na.rm = T)

#data$validServTime <- TRUE
sum(data$servDuration > 7200, na.rm = T)
```

```
## [1] 0
```

```
nrow(data)-sum(!is.na(data$servDuration))
```

```
## [1] 0
```

```
data$validServTime <-  
  with(data, !is.na(servDuration)  
    #& servDuration >= 60 # opcionalmente excluios atendimentos menores que 1 min (desistência?)  
    & servDuration < 7200)  
sum(data$validServTime)
```

```
## [1] 1573
```

```
#quantis da duracao  
quantile(data[data$validServTime, "servDuration"], probs = seq(from=0.1, to=1, by=0.1), na.rm = T)
```

```
##      10%      20%      30%      40%      50%      60%      70%      80%      90%     100%  
##     14.0     30.0    100.6    178.8    244.0    318.0    401.4    583.8    982.8   5556.0
```

```
duracao_atend <-  
  ggplot(data[data$validServTime, ], aes(floor(servDuration/60))) +  
  geom_histogram(aes(y = (..count..)/sum(..count..)), binwidth = 1) +  
  xlim(0, 60) +  
  ylab("Frequência Percentual") +  
  xlab("Duração em minutos") +  
  scale_y_continuous(labels = percent_format(), limits = c(0,0.1))  
ggsave(filename = "duracao_atend20.png", plot = duracao_atend)
```

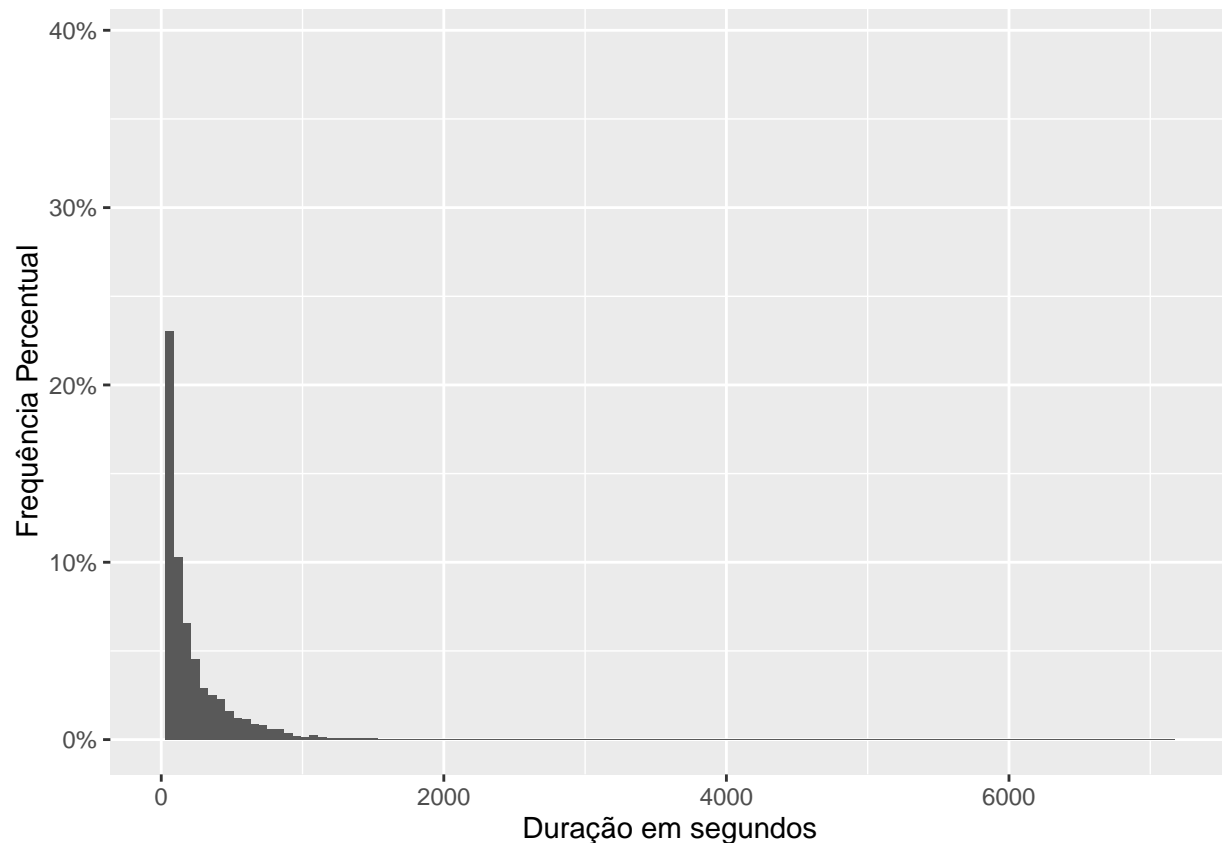
```
## Saving 6.5 x 4.5 in image
```

```
## Warning: Removed 8 rows containing non-finite values (stat_bin).
```

```
write.csv(data[data$validServTime, "servDuration"], file="duracao_atend20.csv")  
write.csv(data[data$validServTime & (format(data$arrivalTimestamp, "%y-%m-%d") %in% highDays), "servDura
```

```
#Weibull (data without atend < 60)  
#test <- floor(rweibull(10000, 0.85283, scale=705.25) + 60)  
  
#Log normal (data without atend < 60)  
#test <- floor(rlnorm(10000, meanlog=6.0411, sdlog=1.1442) + 41.277)
```

```
ggplot(data.frame(test), aes(test)) +  
  geom_histogram(aes(y = (..count..)/sum(..count..)), binwidth = 60) +  
  xlim(0, 7200) +  
  ylab("Frequência Percentual") +  
  xlab("Duração em segundos") +  
  scale_y_continuous(labels = percent_format())
```



```
citation(package = "base", lib.loc = NULL)
```

```
##
## To cite R in publications use:
##
## R Core Team (2016). R: A language and environment for
## statistical computing. R Foundation for Statistical Computing,
## Vienna, Austria. URL https://www.R-project.org/.
##
## A BibTeX entry for LaTeX users is
##
## @Manual{,
##   title = {R: A Language and Environment for Statistical Computing},
##   author = {{R Core Team}},
##   organization = {R Foundation for Statistical Computing},
##   address = {Vienna, Austria},
##   year = {2016},
##   url = {https://www.R-project.org/},
## }
##
## We have invested a lot of time and effort in creating R, please
## cite it when using it for data analysis. See also
## 'citation("pkgname")' for citing R packages.
```