MAE0XXX - Análise de Sobrevivência - Lista 1

Brunão
Rubens Santos Andrade Filho[1]

Maio de 2021

Sumário

[1]Número USP: 10370336

Exercício 6

Considere a distribuição Weibull, com função de sobrevivência dada por

$$S(t) = \exp\left\{-\lambda t^{\rho}\right\}, t \geq 0$$

Utilizando qualquer software ou pacote estatístico de sua preferência, construa gráficos da função de taxa de falha da distribuição Weibull, variando-se os valores dos parâmetros λ e ρ. Considere 6 combinações diferentes de valores de λ e ρ : utilize dois valores diferentes de λ e três valores diferentes de ρ, sendo um deles necessariamente igual a 1 (ou seja, $\rho = 1$). Construa também gráficos das respectivas funções de sobrevivência.

Dada a função de sobrevivência, obtemos a função de densidade de probabilidade:

$$f(t) = -\frac{d}{dt}\left[S(t)\right] = -\frac{d}{dt}\left(e^{-\lambda t^{\rho}}\right) = \lambda\rho t^{\rho-1}e^{-\lambda t^{\rho}}$$

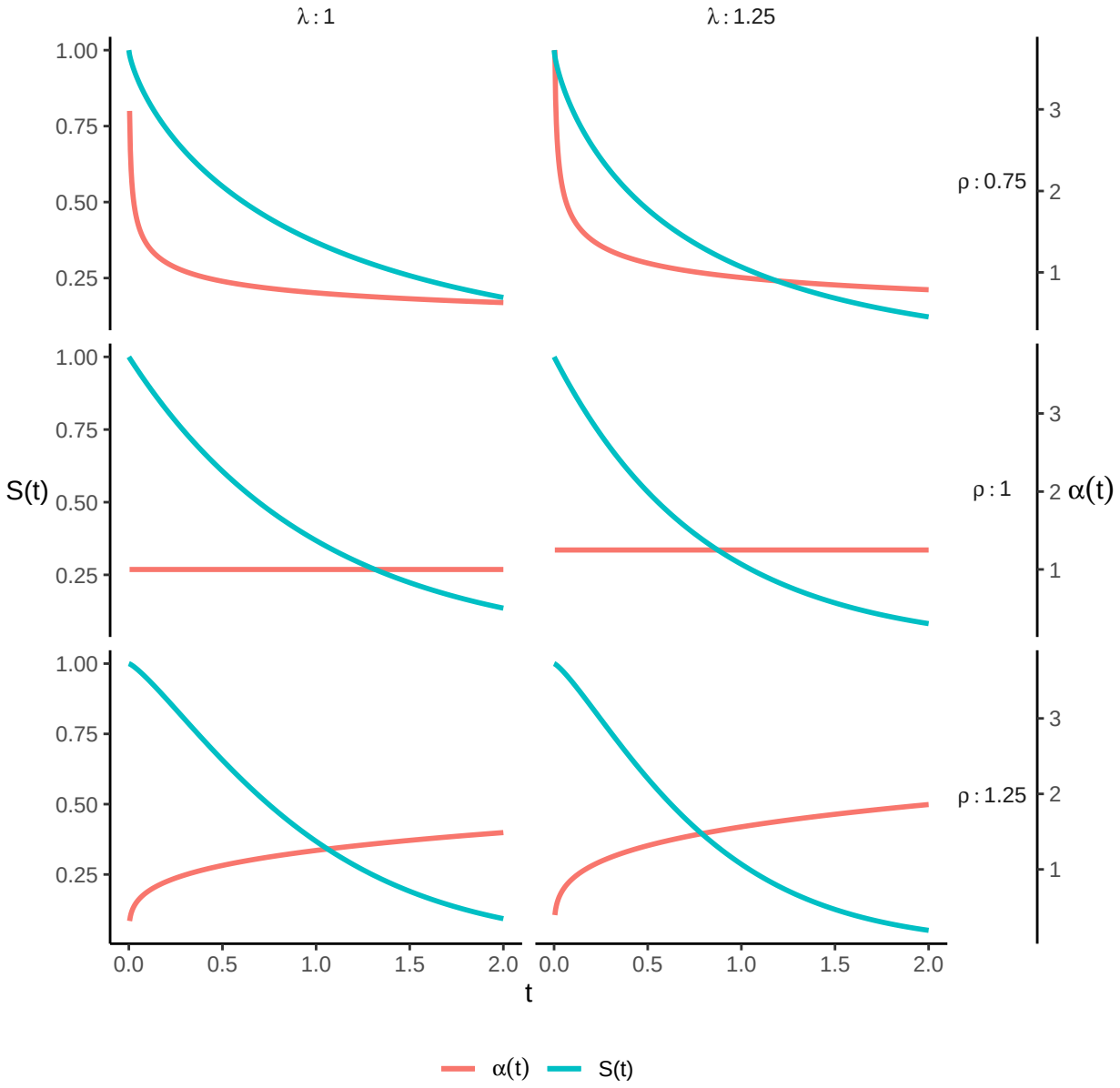
E a taxa de falha:

$$\alpha(t) = \frac{f(t)}{S(t)} = \frac{\lambda\rho t^{\rho-1}e^{-\lambda t^{\rho}}}{e^{-\lambda t^{\rho}}} = \lambda\rho t^{\rho-1}$$

Segue a implementação em `R`:

```r
# criar função de taxa de falha dados lambda e rho
alpha_factory <- function(lambda, rho){
  function(t) ifelse(t>0, lambda * rho * t^(rho-1), NA)
}

# criar função de sobrevivencia dados lambda e rho
s_factory <- function(lambda, rho){
  function(t) ifelse(t>0, exp(-lambda * t^rho), 1)
}

calc_stats <- function(lambda, rho) {
  # cria funcoes de taxa de falha e sobrevivencia
  alpha <- alpha_factory(lambda, rho)
  s <- s_factory(lambda, rho)
  # retorna df com taxas de falha e sobrevicencia
  tibble(lambda, rho, t=ts,
    alpha = alpha(ts), s = s(ts)
  )
}


N <- 500 # número de pontos
t0 <- 0
t1 <- 2
ts = seq(t0,t1, length.out=N)
# cria dados e calcula taxas de falhas e tempos de sobrevivencia
# para cada combinação de parametros
df <- expand_grid(lambda = c(1,5/4), rho = c(3/4,1,5/4)) %>%
  purrr::pmap_dfr(calc_stats)
```

```r
# grafico da funcao de taxa de falha

# para transf. do axis secundário
coef <- max(df$alpha,na.rm = T)

df %>%
  ggplot(aes(x=t))+
  geom_line(aes(y=alpha/coef, color='alpha'),size=1)+
  geom_line(aes(y=s, color="S"),size=1)+
  facet_grid(vars(rho),vars(lambda), scales = 'free_y',
             labeller = function(x) label_parsed(label_both(x)))+
  ylab("")+ylim(0,1)+
  scale_color_discrete(
    name="",labels=c("S"="S(t)", "alpha"=expression(alpha(t))))+
  scale_y_continuous(
    name = "S(t)",

    # Add a second axis and specify its features
    sec.axis = sec_axis(~.*coef, name=expression(alpha(t)))
  ) +
  theme_classic() +
  theme(
    legend.position = "bottom",
    strip.background = element_blank(),
    axis.text.y = element_text(angle = 0),
    axis.title.y.left = element_text(vjust = 0.5,angle = 0),
    axis.title.y.right = element_text(vjust = 0.5,angle = 0),
    strip.text.y = element_text(angle = 0),
    ) +
  NULL
```



Exercício 7

Considere a distribuição Weibull do exercício anterior e escolha uma das combinações de λ e ρ utilizadas. Utilizando qualquer software ou pacote estatístico de sua preferência, gere uma amostra com a distribuição Weibull escolhida de tamanho $n = 100$.

IMPORTANTE: Escreva claramente qual foi o software ou pacote estatístico utilizado e inclua necessariamente os códigos utilizados para a resolução do exercício.

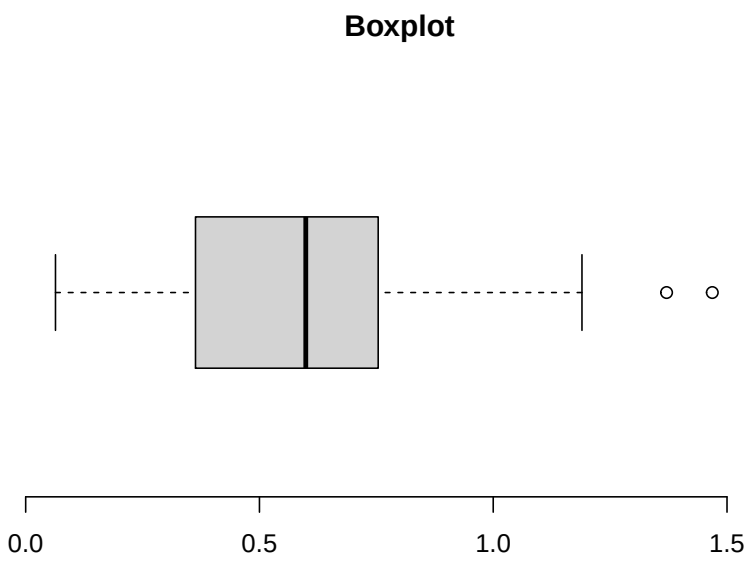
No `R`, a distribuição Weibull é parametrizada em termos de uma parâmetro de localização a e escala b :

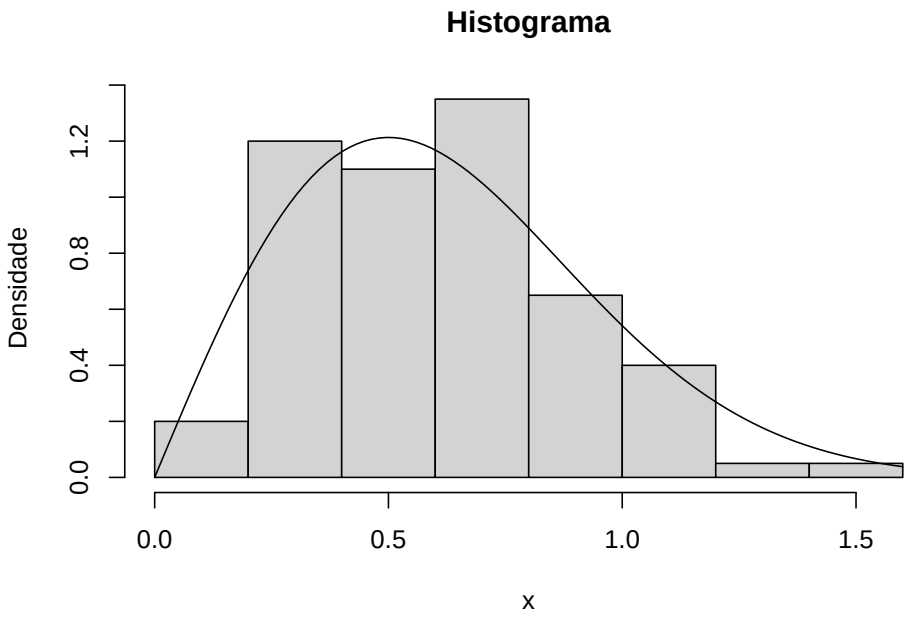
$$f(t) = \frac{a}{b}\left(\frac{t}{b}\right)^{a-1} e^{-\left(\frac{t}{b}\right)^a}$$

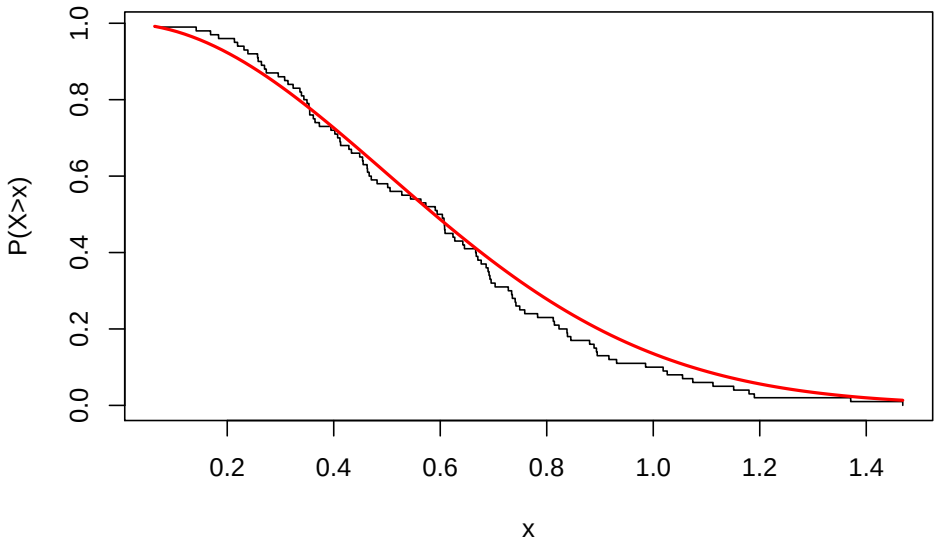
Reescrevendo, obtemos a e b em termos λ e ρ :

$$f(t) = \frac{a}{b}\left(\frac{t}{b}\right)^{a-1} e^{-\left(\frac{t}{b}\right)^a} = a\frac{1}{b^a}t^{a-1}e^{-\frac{1}{b^a}t^a} = \rho\lambda t^{\rho-1}e^{-\lambda t^\rho} \Rightarrow a = \rho, \qquad \lambda = \frac{1}{b^a} \Rightarrow b = \frac{1}{\lambda^{\frac{1}{a}}}$$

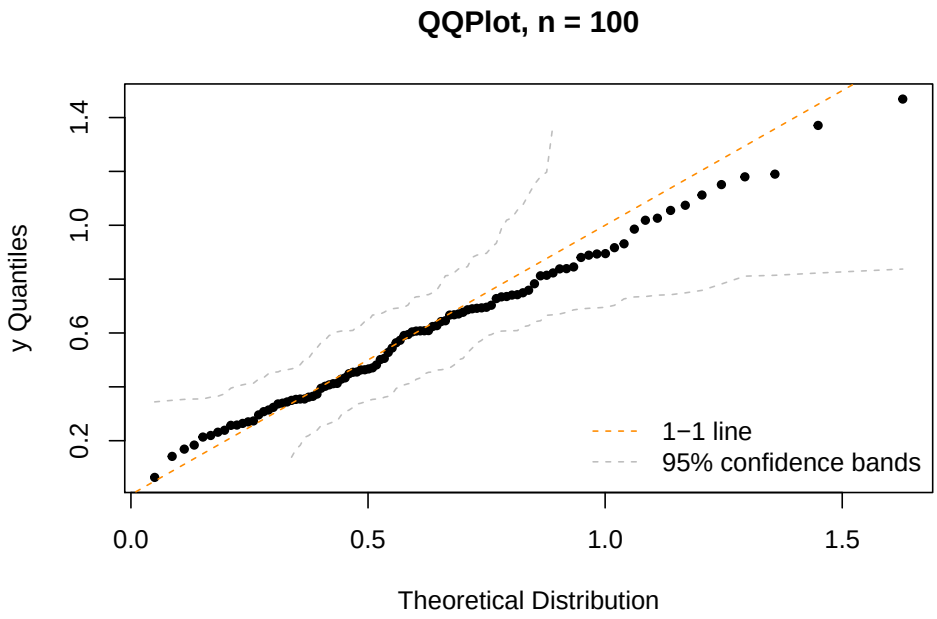
(a) Obtenha um boxplot dos dados e um histograma (com a curva da densidade teórica também no gráfico). Obtenha também a curva de sobrevivência empírica dos dados e coloque num mesmo gráfico a curva empírica e a curva teórica utilizada para gerar os gráficos.



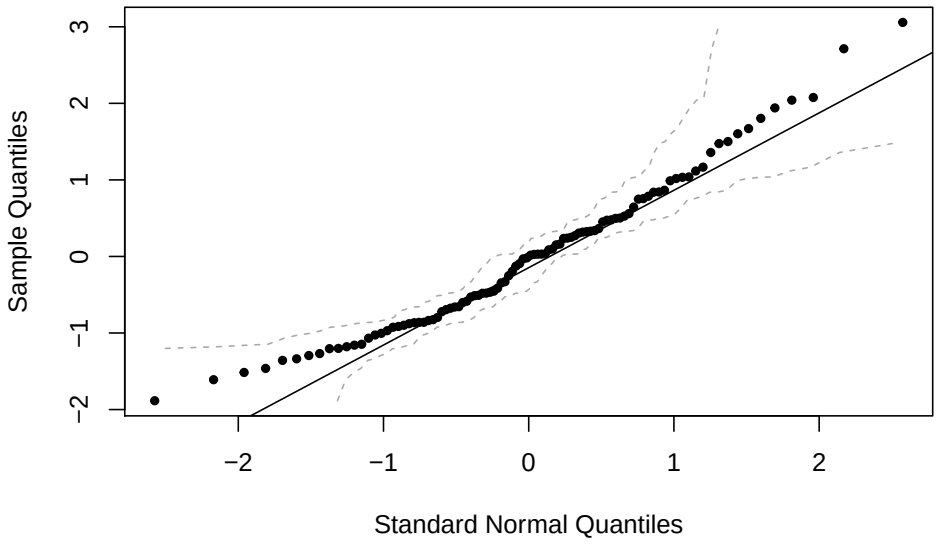




(b) Faça um gráfico de quantis (QQPlot) comparando os quantis empíricos com os quantis teóricos da distribuição Weibull utilizada para gerar os dados.



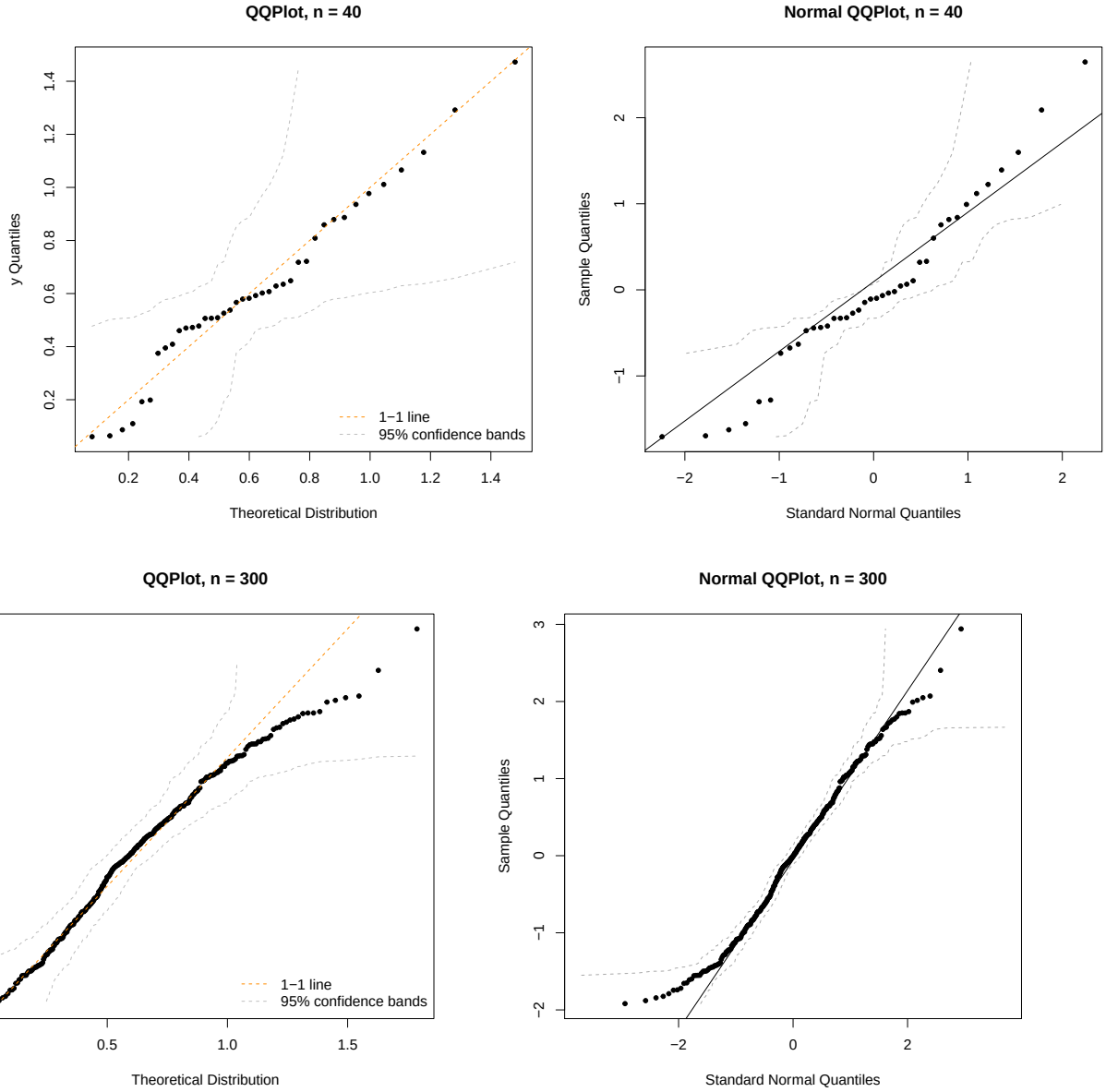
(c) Padronize os dados (ou seja, subtraia a média amostral e divida pelo desvio padrão) e faça um gráfico de quantis ($Q Q$ Plot) comparando os quantis empíricos da variável padronizada com os da normal padrão. Discuta a adequabilidade da distribuição normal aos dados.

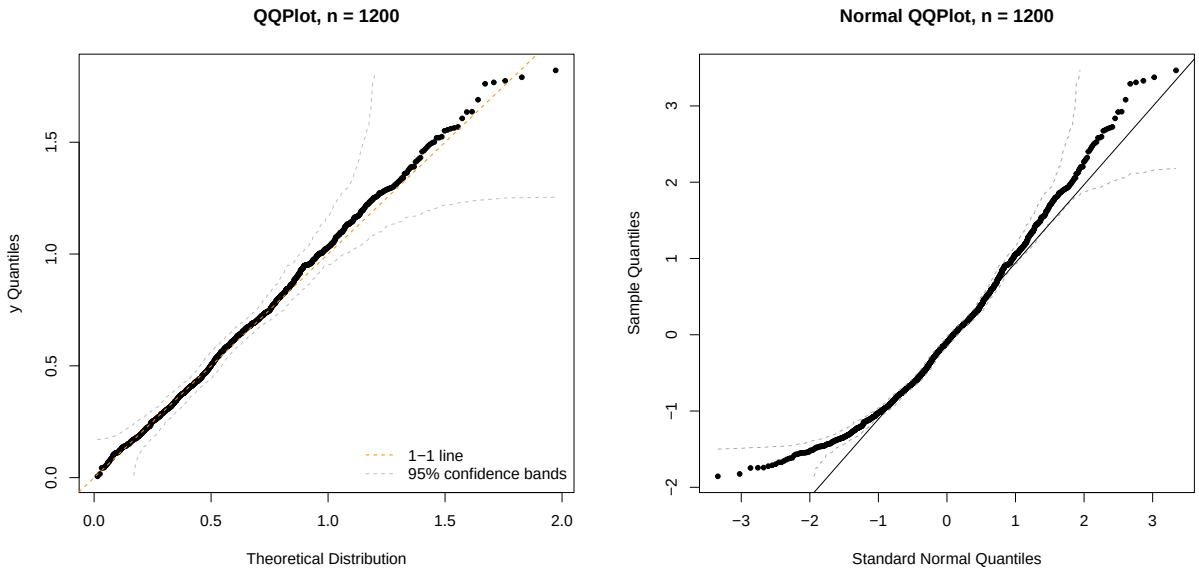


O gráfico Normal QQPlot mostra no eixo y os quantis da amostra padronizada, isto é, as estatísticas de ordem da amostra padronizada e, no eixo x, os quantis teóricos de uma normal padrão. Se a amostra é retirada de uma população normal, os pontos deveriam cair próximos da linha diagonal. Nota-se pelo gráfico

que os pontos desviam-se um pouco da linha nas extremidades, entretanto um desvio um pouco maior nas extremidades é esperado. Com isso, apesar de saber de qual distribuição foi tirada a amostra, o gráfico não descarta-se a adequabilidade da distribuição normal aos dados.

(d) Repita os itens (b) e (c) para $n = 40, n = 300$ e $n = 1200$.





Código Completo

```
knitr::opts_chunk$set(warning=FALSE,
                      # fig.dim = c(5,5),
                      # out.height = '40%',
                      # fig.align = 'center',
                      message=FALSE
                      )

library(tidyverse)
library(ggplot2)

library(extRemes) # bandas de confiança nos qqplot
# criar função de taxa de falha dados lambda e rho
alpha_factory <- function(lambda, rho){
  function(t) ifelse(t>0, lambda * rho * t^(rho-1), NA)
}

# criar função de sobrevivencia dados lambda e rho
s_factory <- function(lambda, rho){
  function(t) ifelse(t>0, exp(-lambda * t^rho), 1)
}

calc_stats <- function(lambda, rho) {
  # cria funcoes de taxa de falha e sobrevivencia
  alpha <- alpha_factory(lambda, rho)
  s <- s_factory(lambda, rho)
  # retorna df com taxas de falha e sobrevicencia
  tibble(lambda, rho, t=ts,
    alpha = alpha(ts), s = s(ts)
```

```
  )
}


N <- 500 # número de pontos
t0 <- 0
t1 <- 2
ts = seq(t0,t1, length.out=N)
# cria dados e calcula taxas de falhas e tempos de sobrevivencia
# para cada combinação de parametros
df <- expand_grid(lambda = c(1,5/4), rho = c(3/4,1,5/4)) %>%
  purrr::pmap_dfr(calc_stats)
# grafico da funcao de taxa de falha

# para transf. do axis secundário
coef <- max(df$alpha,na.rm = T)

df %>%
  ggplot(aes(x=t))+
  geom_line(aes(y=alpha/coef, color='alpha'),size=1)+
  geom_line(aes(y=s, color="S"),size=1)+
  facet_grid(vars(rho),vars(lambda), scales = 'free_y',
             labeller = function(x) label_parsed(label_both(x)))+
  ylab("")+ylim(0,1)+
  scale_color_discrete(
    name="",labels=c("S"="S(t)", "alpha"=expression(alpha(t))))+
  scale_y_continuous(
    name = "S(t)",

    # Add a second axis and specify its features
    sec.axis = sec_axis(~.*coef, name=expression(alpha(t)))
  ) +
  theme_classic() +
  theme(
    legend.position = "bottom",
    strip.background = element_blank(),
    axis.text.y = element_text(angle = 0),
    axis.title.y.left = element_text(vjust = 0.5,angle = 0),
    axis.title.y.right = element_text(vjust = 0.5,angle = 0),
    strip.text.y = element_text(angle = 0),
    ) +
  NULL


# parametros
rho <- 2
lambda <- 2

# semente do gerador de numeros aleatorios
set.seed(1)
# amostra aleatória da distribuição
n <- 100
x <- rweibull(n, shape=rho, scale = 1/lambda^(1/rho))
```

```r
# boxplot
boxplot(x, ylab="", main="Boxplot", horizontal = TRUE,  ylim=c(0, 1.6), frame=FALSE)

# histograma + fdp
hist(x,freq = FALSE, main="Histograma", ylab="Densidade")
fdp <- function(x) dweibull(x, shape=rho, scale = 1/lambda^(1/rho))
curve(fdp, add=TRUE)

# f. empirica de distribuição, sobrevivencia e teorica de sobrevivencia
fed <- ecdf(x)
fes <- function(x) 1 - fed(x)
fts <- function(x) pweibull(x, shape=rho, scale = 1/lambda^(1/rho), lower.tail=FALSE)

sort(x) %>% plot(fes(.), type="s", ylab="P(X>x)", xlab="x")
curve(fts, add=TRUE, col="red", lwd=2)
# função teorica quantilica
dfp <- function(p) qweibull(p, shape=rho, scale = 1/lambda^(1/rho))

qqplot(dfp(ppoints(x)), x, xlab = "Theoretical Distribution",
         main=paste("QQPlot, n =", n), regress = FALSE)
sx <- (x - mean(x))/sd(x)
qqnorm(sx)
qqline(sx)
# fig.dim = c(5,5), fig.ncol=2, out.height = NULL, out.width="50%", fig.align = 'left'

gerar_graficos <- function(n){
  # amostra aleatória da distribuição
  x <- rweibull(n, shape=rho, scale = 1/lambda^(1/rho))

  # b)
  # função teorica quantilica
  dfp <- function(p) qweibull(p, shape=rho, scale = 1/lambda^(1/rho))
  qqplot(dfp(ppoints(n)), x, xlab = "Theoretical Distribution",
         main=paste("QQPlot, n =", n), regress = FALSE)

  # c)
  sx <- (x - mean(x))/sd(x)
  qqnorm(sx, main=paste("Normal QQPlot, n =", n))
  qqline(sx)
}

gerar_graficos(n = 40)

gerar_graficos(n = 300)

gerar_graficos(n = 1200)
```