Otimização de parâmetros: Distribuição de Weibull

Prof. Paulo Cerqueira Junior

Introdução

A distribuição de Weibull é amplamente utilizada em diversas áreas, especialmente em análise de sobrevivência e confiabilidade, devido à sua flexibilidade em modelar diferentes tipos de falhas. Para estimar os parâmetros da distribuição de Weibull, um método comum é a máxima verossimilhança, que requer o cálculo das derivadas da função de log-verossimilhança em relação aos parâmetros.

Função de Densidade de Probabilidade de Weibull

A função de densidade de probabilidade (fdp) de Weibull é dada por:

$$f(x; s, a) = asx^{a-1} \exp(-sx^a)$$

Função de Log-Verossimilhança

A função de log-verossimilhança para uma amostra de n observações é dada por:

$$\ln L(s, a) = n \ln a + n \ln s + (a - 1) \sum_{i=1}^{n} \ln x_i - s \sum_{i=1}^{n} x_i^a$$

Derivadas de Primeira Ordem

As derivadas parciais da função de log-verossimilhança em relação a a e s são:

$$\frac{\partial \ln L}{\partial a} = \frac{n}{a} + \sum_{i=1}^{n} \ln x_i - s \sum_{i=1}^{n} x_i^a \ln x_i$$

e

$$\frac{\partial \ln L}{\partial s} = \frac{n}{s} - \sum_{i=1}^{n} x_i^a$$

```
## Vetor escore:

U <- function(theta, dados){

  a <- theta[1]
  s <- theta[2]
  n <- length(dados)
  x <- dados

  U_a    <- (n/a) + sum(log(x)) - s*sum((x^a)*log(x))
  U_sigma <- (n/s) - sum(x^a)

  return(c(U_a, U_sigma))

}
```

Derivadas de Segunda Ordem

As derivadas parciais de segunda ordem são:

$$\frac{\partial^2 \ln L}{\partial a^2} = -\frac{n}{a^2} - s \sum_{i=1}^{n} x_i^a (\ln x_i)^2,$$

$$\frac{\partial^2 \ln L}{\partial s^2} = -\frac{n}{s^2},$$

e

$$\frac{\partial^2 \ln L}{\partial s \partial a} = -\sum_{i=1}^{n} x_i^a \ln x_i.$$

```
H <- function(theta, dados){

  a <- theta[1]
  s <- theta[2]
  n <- length(dados)
  x <- dados

  a_2   <- (-n/(a^2)) - s*sum((x^a)*(log(x)^2))
  sig_2 <- -n/(s^2)
  a_sig <- -sum((x^a)*log(x))
  sig_a <- a_sig

  return(matrix(c(a_2, a_sig, sig_a, sig_2), nrow=2, ncol = 2))
}
```

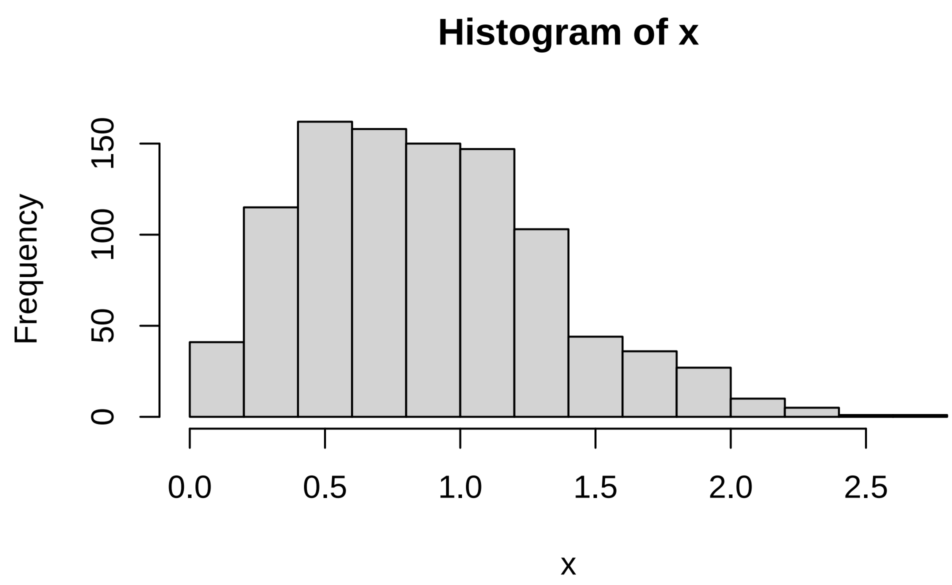
Gerando os dados:

Foi gerado uma amostra de tamanho `1000` de uma distribuição de Weibull com parâmetro de forma $a = 2$ e escala $s = 1$.

```
## Modelo Weibull:

set.seed(1234567890)

n <- 1000 # tamanho da amostra
a.p <- 2  # forma
s.p <- 1  # escala
x <- rweibull(n, shape = a.p, scale = 1/(s.p^a.p))  # dados gerados
hist(x)
```



Algorítmo Newton-Raphson:

```
theta0 <- c(1, 0.5) # Chute inicial
dif <- 1 #Diferença
erro <- 10^(-6) # Tolerancia
i <- 1 # contador

while( dif>erro ){

  H0 <- H(theta = theta0, dados = x) # Matrix Hessiana no theta0
  U0 <- U(theta = theta0, dados = x) # Vetor escore no theta0

  prodHU <- solve(H0, U0)
  theta1 <- theta0 - prodHU # Passo de atualização

  dif <- max(abs(theta1-theta0)) # Passo de avaliação da diferença

  theta0 <- theta1
  i<- i+1
  cat("Iter:", i, "est:", theta1, "\n")
```

```
  #if(i==10)break
}
```

```
Iter: 2 est: 1.622641 0.7789293
Iter: 3 est: 1.974736 0.9701761
Iter: 4 est: 1.975344 1.01744
Iter: 5 est: 1.97418 1.020094
Iter: 6 est: 1.974179 1.020101
Iter: 7 est: 1.974179 1.020101
```

```
## Estimativa:

cat("A estimativa final foi:", "a=", theta1[1],"-", "s=", theta1[2], sep="")
```

```
A estimativa final foi:a=1.974179-s=1.020101
```

Usando a função `optim` para otimização:

Neste passo precisamos somente da função de log verossimilhança.

```
logWeibull <- function(theta, dados){
  a <- theta[1]
  s <- theta[2]
  n <- length(dados)
  x <- dados

  l <- n*log(a)+n*log(s)+(a-1)*sum(log(x))-s*sum((x)^a)
  return(-l)
}
```

Dentro da função `logWeibull` o objeto `l` dará retorno negativo, pois a função `optim` determina ponto de mínimo.

```
theta0 <- c(3, 2) # Chute inicial
est <- optim(par = theta0, fn = logWeibull, gr =NULL , method ="BFGS" ,
      hessian = TRUE, dados=x)

est
```

```
$par
[1] 1.974179 1.020102

$value
[1] 595.8966

$counts
function gradient
      39        9

$convergence
[1] 0

$message
NULL

$hessian
         [,1]     [,2]
[1,] 459.6588 198.7491
[2,] 198.7491 960.9790
```