UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CÂMPUS DE PRESIDENTE PRUDENTE

Modelo da Nova Distribuição Exponencial Inversa Transmutada Generalizada

Guilherme Teixeira

1 Introdução

Distribuições de probabilidade tem diversas aplicações com o objetivo de estudar e descrever o comportamento de fenômenos no mundo real. As distribuições Clássicas têm sido desenvolvidas durante décadas para modelar dados nas áreas de Engenharia, Atuária, Meio Ambiente, Ciências Médicas, Finanças, Demografia e Economia. Muitas destas distribuições têm sido propostas para modelar dados de tempo de vida na literatura, uma delas, é a distribuição exponencial inversa.

A distribuição exponencial inversa foi introduzida primeiramente por Keller e Kamath, para modelar o tempo de vida. A função distribuição acumulada (F.D.A) da distribuição é

$$G(x) = \exp(-\frac{\beta}{x}) \quad x > 0 \tag{1}$$

Onde β é o parâmetro de escala. A função densidade de probabilidade (f.d.p) correspondente de (1) é

$$g(x) = \frac{\beta}{x^2} \exp(-\frac{\beta}{x}) \quad x > 0 \tag{2}$$

Diversas distribuições univariadas têm sido expandidas para flexibilizar e melhorar a acurácia do modelo. Baseado neste entendimento, inúmeros autores, têm proposto e generalizado a distribuição exponencial inversa. A exponencial kumaraswamy-inversa [2], distribuição exponencial inversa generalizada [3], distribuição exponencial inversa generalizada exponenciada [4], distribuição exponencial inversa transmutada [5], entre outras.

Neste trabalho, introduzimos a Nova Distribuição Exponencial Inversa Transmutada Generalizada (NDEITG), na qual é uma forma generalizada da distribuição exponencial inversa que faz com que ela seja mais flexível. Esta distribuição pode ser usada para modelar dados de taxa de falha com comportamento semelhante a um formato de banheira. Este recurso, torna a distribuição mais maleável do que a distribuição exponencial inversa e a generalizada.

A continuidade deste estudo é dada na seguinte sequência: o tópico 2 fornece a função de distribuição, como ela é obtida, e em seguida, a sua função de risco e de

sobrevivência. Nós derivamos suas propriedades matemáticas para encontrar a função geratriz de momentos (fgm) no tópico 3, enquanto no tópico 4 estimamos os parâmetros da nova distribuição, através da máxima verossimilhança. No tópico 5, foram simuladas amostras de diferentes tamanhos para estudar a adequação dos parâmetros estimados. E por fim, concluímos o análise dos resultados obtidos.

2 Nova Distribuição Exponencial Inversa Transmutada Generalizada

A F.D.A de uma Outra Classe Transmutada Generalizada (OCTG) de distribuições modeladas por Fator Merovci, é o ponto de partida para encontrarmos a (NDEITG). Sua função distribuição acumulada (fda) é dada por

$$F(x) = (1+\lambda)[1-(\bar{G}(x))^{\alpha}] - \lambda[1-(G(x))^{\alpha}]^2 \quad (3)$$

Com sua correspondente fdp

$$f(x) = \alpha g(x)\big(\bar{G}(x)\big)^{\alpha-1}\{1+\lambda-2\lambda\big[1-\big(\bar{G}(x)\big)^{\alpha}\big]\} \quad (4)$$

Onde $\alpha > 0,\ |\lambda| < 1$ e $G(x)$ é linha de base da função distribuição acumulada. $\bar{G}(x)$ é a estrutura de base para a função densidade de probabilidade, que é representada por

$$\bar{G}(x) = 1 - G(x)$$

Substituindo (1), (2) e (5) em (3) e (4) obtemos a função distribuição acumulada

$$F(x) = (1+\lambda)[1-(1-\exp(-\tfrac{\beta}{x}))^{\alpha}] - \lambda[1-(1-\exp(-\tfrac{\beta}{x}))^{\alpha}]^2 \quad (6)$$

sua fdp

$$f(x) = \tfrac{\alpha\beta}{x^2}[(1-\exp(-\tfrac{\beta}{x})]^{\alpha-1}\{1+\lambda-2\lambda[1-(1-\exp(-\tfrac{\beta}{x}))^{\alpha}]\} \quad (7)$$

da Nova Distribuição Exponencial Inversa Transmutada Generalizada $(\text{NDEITG})(x, \alpha, \beta, \lambda)$. Onde $\alpha > 0,\ \beta > 0,\ |\lambda| \leq 1$

2.1 Submodelos da (NDEITG)

Os submodelos podem ser derivados do (NDEITG) reduzindo seus parâmetros, onde no caso, o tópico (II) resulta no modelo em que trabalharemos posteriormente

I. Se $\alpha = 1,$ a equação (7) é reduzida para a distribuição exponencial inversa transmutada com parâmetros β e λ[5]

II. Se $\lambda = 0$ e $\alpha = 1$, (6) é reduzida para um parâmetro β[1] com distribuição exponencial inversa

2.2 Formato das Funções Densidade de Probabilidade e Distribuição Acumulada

Funções de densidade de probabilidade e distribuição acumulada para diferentes valores de α e β

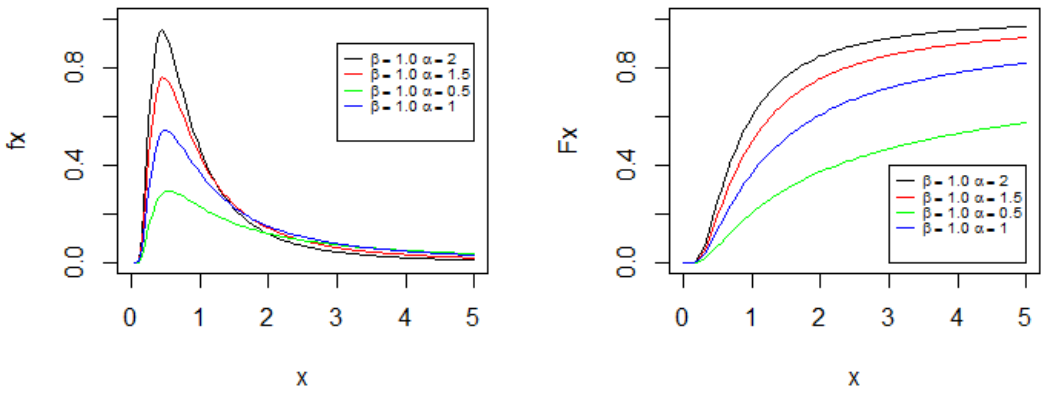

Fig. 1. fdp da (NDEITG) **Fig. 2. fda da** (NDEITG)

2.3 Função de Sobrevivência e de Risco

A função de sobrevivência pode ser encontrada facilmente fazendo $S(x) = 1 - F(x)$, enquanto a função de risco $h(x)$ de outa forma conhecida como taxa de falha da NDEITG com parâmetros reduzidos (x, α, β) é obtida através da

$$h(x) = \frac{f(x)}{1 - F(x)}$$

$$h(x) = \frac{\left(1 - e^{-\frac{\beta}{x}}\right)^{\alpha - 1} \alpha \beta e^{-\frac{\beta}{x}}}{x^2} \Bigg/ \left(1 - e^{-\frac{\beta}{x}}\right)^{\alpha} \tag{8}$$

2.4 O Comportamento da Função de Sobrevivência e de Risco

As possíveis formas da função de sobrevivência da nova distribuição exponencial inversa transmutada generalizada (NDEITG) para α e β são ilustradas nas figuras abaixo.

A função de risco na Fig. 3. mostra que a NDEITG permite o crescimento e decrescimento uni modal da taxa de falha.

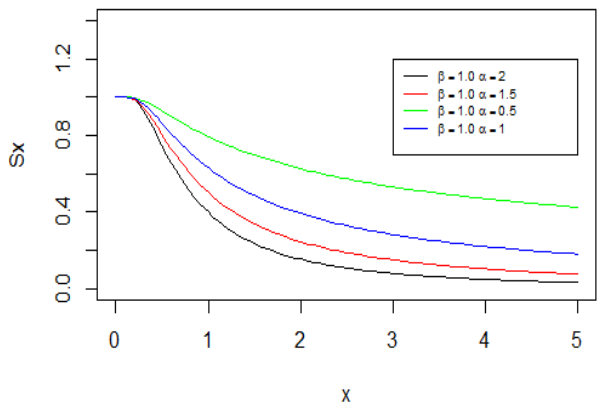

Fig. 1. Plotagem da Função de Sobrevivência da (NDEITG)

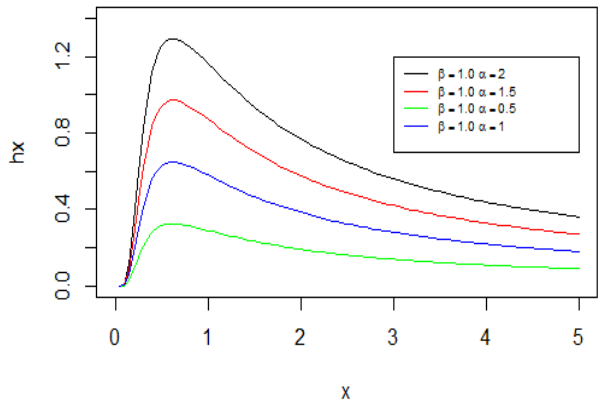

Fig. 2. Plotagem da Função de Risco da (NDEITG)

3 Momentos (NDEITG)

Neste tópico apresentamos o *rth moment*, a função geratriz de momentos (fgm), e o método para gerar valores da NDEITG (x,α, β).

3.1 Momento central

Por definição, temos que o *rth moment*, (momento central em torno da origem), é definido se $E(x^{k)}$ existe. Partindo deste princípio,

$$E(x^k) = \int_{-\infty}^{\infty} x^k f(x) dx$$

$$E(x^k) = \int_0^{\infty} x^{k-2} \left(1 - e^{-\frac{\beta}{x}}\right)^{\alpha-1} \alpha\beta e^{-\frac{\beta}{x}} dx$$

Com Variância

$$V(x) = \int_0^{\infty} \left(1 - e^{-\frac{\beta}{x}}\right)^{\alpha-1} \alpha\beta e^{-\frac{\beta}{x}} - \left(\int_0^{\infty} x f(\alpha, \lambda, x) dx\right)^2 dx$$

Com este resultado, nota-se que não há uma forma fechada para esta distribuição, e com isso surge a necessidade de fazer uma expansão de séries

3.2 Função Geratriz de Momentos

A função geratriz de momentos (fgm) da NDEITG é obtida através de expansão binomial, e sua expressão parte do seguinte princípio:

$$Mx(t) = \int_{-\infty}^{\infty} e^{tx} f(x) dx$$

E é definida na forma

$$Mx(t) = \int_{-\infty}^{\infty} e^{tx} \left\{ \frac{\left(1 - e^{-\frac{\beta}{x}}\right)^{\alpha-1} \alpha\beta e^{-\frac{\beta}{x}}}{x^2} \right\} dx$$

3.3 Método para gerar valores da Nova Distribuição EITG

Um método recomendado a ser usar é encontrar a inversa da fda, se baseando no Teorema Fundamental da Transformação de Probabilidades. Entretanto, através de testes realizados no software RStudio, pudemos notar que o método não é eficaz, neste caso, apresentando resultados distantes dos verdadeiros parâmetros. Um método alternativo, é encontrar o valor de $\mathrm{F(x)} - \mathrm{u}$ que zera a equação. Testando ambos procedimentos, conseguimos analisar e assim seguir com a ferramenta mais adequada para gerar os valores.

4 Estimação dos Parâmetros

O método da máxima verossimilhança é usado para a estimação dos parâmetros da NDEITG (x, α, β). A função de verossimilhança encontrada fazendo

$$L(\alpha, \beta; x) = \prod_{i=1}^{n} \left\{ \frac{\left(1-e^{-\frac{\beta}{x}}\right)^{\alpha-1} \alpha\beta e^{-\frac{\beta}{x}}}{x^2} \right\} 00$$

Aplicando o logaritmo natural nos dois lados da equação, ln $L(\alpha, \beta; x)$ = l então temos

$$l = n \ln \alpha + n \ln b - 2 \sum_{i=1}^{n} \ln xi - \beta \sum_{i=1}^{n} + (\alpha - 1) \sum_{i=1}^{n} \ln \left[1 - e^{-\frac{\beta}{xi}}\right]$$

Os estimadores de máxima verossimilhança são obtidos derivando parcialmente seus respectivos parâmetros e igualando suas equações a zero

$$\frac{\partial l}{\partial \alpha} = \frac{n}{\alpha} + \sum_{i=1}^{n} ln \left[1 - e^{-\frac{\beta}{xi}}\right]$$

$$\frac{\partial l}{\partial \beta} = \frac{n}{\beta} - \sum_{i=1}^{n} \frac{1}{xi} + (\alpha - 1) \sum_{i=1}^{n} \left[\frac{e^{-\frac{\beta}{xi}}}{xi\left(1 - e^{-\frac{\beta}{x}}\right)} \right]$$

5 Estudo da Simulação

Neste tópico, temos o objetivo de simular o comportamento da distribuição a partir de testes, gerando valores dentro dos parâmetros estimados, para testar a precisão do modelo.

Logo abaixo, temos os resultados gerados por meio do método de máxima verossimilhança com 500 amostras de tamanho n = (10, 25, 50, 100), com diferentes valores para $\alpha \; e \; \beta$.

Tabela 1 – Decorrência da estimação de máxima verossimilhança dos parâmetros para 500 amostras de tamanho *n* = (10, 25, 50, 100), com diferentes valores de α *e* β.

n	Parâmetro	Média	Desvio-padrão	EQM	Vício	IC		PC
						Inferior	Superior	
10	$\alpha = 2$	2.80	3.55	6.602	1.634	2.504	5.219	0.936
	$\beta = 1$	1.20	1.592	2.599	0.593	0.459	1.563	0.951
	$\alpha = 1.5$	1.323	2.153	1.549	0.760	0.365	3.954	0.912
	$\beta = 1$	0.967	0.932	0.970	1.450	0.233	2.440	0.946
	$\alpha = 0.5$	0.724	0.873	3.925	2.692	0.126	1.532	0.932
	$\beta = 1$	1.169	1.405	1.405	1.035	0.593	2.421	0.958
	$\alpha = 1$	1.243	1.052	4.591	1.974	-1.525	2.745	0.924
	$\beta = 1$	1.455	2.953	9.239	0.643	0.163	5.215	0.901
25	$\alpha = 2$	1.889	1.692	2.594	1.053	0.423	7.293	0.945
	$\beta = 1$	1.356	1.284	1.095	0.392	0.157	3.573	0.934
	$\alpha = 1.5$	1.571	0.988	5.697	1.475	0.215	4.503	0.951
	$\beta = 1$	1.050	1.039	0.602	0.206	0.451	4.692	0.967
	$\alpha = 0.5$	0.385	0.841	0.816	1.450	-0.191	1.536	0.929
	$\beta = 1$	0.873	0.765	1.495	0.683	0.510	3.682	0.931
	$\alpha = 1$	0.634	0.984	2.304	2.506	0.014	3.567	0.891
	$\beta = 1$	0.735	0.534	1.657	1.953	0.603	6.235	0.915
	$\alpha = 2$	1.872	1.906	1.995	1.604	1.115	5.804	0.945
	$\beta = 1$	1.590	0.893	2.640	0.860	0.362	4.035	0.925
50	$\alpha = 2$	1.396	1.495	4.457	1.696	1.098	6.953	0.939
	$\beta = 1$	1.246	1.687	2.680	1.552	0.002	3.599	0.949
	$\alpha = 1.5$	1.845	1.935	2.852	0.934	0.939	3.865	0.956
	$\beta = 1$	1.538	1.402	1.703	0.594	0.185	2.595	0.972
	$\alpha = 0.5$	0.335	0.196	0.693	0.665	0.225	1.473	0.937
	$\beta = 1$	0.742	0.935	2.405	1.074	-1.398	1.914	0.924
	$\alpha = 1$	1.631	1.436	1.234	1.754	0.406	1.734	0.951
	$\beta = 1$	1.984	1.705	0.523	0.993	0.351	1.606	0.938
100	$\alpha = 2$	3.134	2.591	6.392	2.529	1.274	2.708	0.919
	$\beta = 1$	2.496	2.643	4.940	1.601	0.551	1.354	0.944
	$\alpha = 1.5$	1.346	1.965	2.763	1.099	0.963	2.250	0.969
	$\beta = 1$	0.410	0.653	1.987	2.567	0.402	1.754	0.924
	$\alpha = 0.5$	0.957	0.864	0.943	1.343	0.357	1.029	0.978
	$\beta = 1$	1.622	1.473	0.865	0.429	0.522	1.226	0.953
	$\alpha = 1$	2.582	2.052	4.392	1.042	0.381	1.649	0.937
	$\beta = 1$	2.492	2.634	7.068	2.996	0.735	1.503	0.940

Com os resultados obtidos através da simulação, podemos analisar e dizer se os parâmetros estimados se ajustaram bem a distribuição, de modo a gerar valores aproximados aos já pré-definidos inicialmente.

De fato, ao aumentar a quantidade de valores gerados, simulando 500 vezes a amostra, nos leva a concluir que o intervalo de confiança começa a diminuir, ao aumentar

de 25 para 50 e de 50 para 100, posteriormente. No que se refere ao desvio-padrão, erro quadrático médio e vício do estimador, nota-se que tendem a zero quando n é suficientemente grande, ou seja, são assintoticamente não viciados. A Probabilidade de cobertura se mostrou mais eficiente ao gerar 100 valores para $\alpha\ e\ \beta$ igual a 0.5 e 1, respectivamente.

6 Consolidações Finais

Neste trabalho foi proposto a flexibilização da distribuição exponencial inversa, com o intuito de flexibilizá-la de modo a se tornar uma nova distribuição exponencial inversa transmuta generalizada, com seus parâmetros reduzidos. O estimador de máxima verossimilhança se mostrou eficiente ao estimar os parâmetros, entretanto, o método da inversa de gerar valores, não foi preciso, vindo assim, a nos levar a um método alternativo onde achamos o zero da função acumulada.

Com isso, pudemos prosseguir e concluir que os parâmetros foram bem ajustados, são consistentes a assintoticamente não viciados para um n suficientemente grande.

Referências

https://stat.ethz.ch/R-manual/R-devel/library/stats/html/uniroot.html
www.wolframalpha.com/input/
http://www.informit.com/articles/article.aspx?p=1863432&seqNum=4
https://www.researchgate.net/profile/Faton_Merovci

Apêndice A – Programas utilizados

```
n=800
a=2
b=1

##### alpha=a e beta=b ##### função densidade f(x) ####
fx= function(x)(0.1e1 - exp(-b / x)) ^ a * a * b * exp(-b / x) / x ^ 2 / (0.1e1 - exp(-b / x))



a=2
b=1
plot(fx, xlim = c(0,5), ylim = c(0,1), col = "black")
a=1.5
plot(fx, add = T, xlim = c(0,5), ylim = c(0,1), col = "red")
a=0.5
plot(fx, add = T, xlim = c(0,5), ylim = c(0,1), col = "green")
a=1
plot(fx, add = T, xlim = c(0,5), ylim = c(0,1), col = "blue")
legend(x = c(3,5),y = c(0.5,0.9),
      legend = c(expression(paste(beta=="1.0"," ", alpha==2.0)),
                  expression(paste(beta=="1.0"," ", alpha==1.5)),
                  expression(paste(beta=="1.0"," ", alpha==0.5)),
                  expression(paste(beta=="1.0"," ", alpha==1.0))),
      lty =1, cex = 0.65,
      col = c("black","red","green","blue"))

############################## função acumulada F(x)

Fx = function(x) 0.1e1 - (0.1e1 - exp(-b / x)) ^ a

a=2
```

```
b=1
plot(Fx, xlim = c(0,5), ylim = c(0,1), col = "black")
a=1.5
plot(Fx, add = T, xlim = c(0,5), ylim = c(0,1), col = "red")
a=0.5
plot(Fx, add = T, xlim = c(0,5), ylim = c(0,1), col = "green")
a=1
plot(Fx, add = T, xlim = c(0,5), ylim = c(0,1), col = "blue")
legend(x = c(3,5),y = c(0.,0.4),
      legend = c(expression(paste(beta=="1.0"," ", alpha==2.0)),
            expression(paste(beta=="1.0"," ", alpha==1.5)),
            expression(paste(beta=="1.0"," ", alpha==0.5)),
            expression(paste(beta=="1.0"," ", alpha==1.0))),
      lty =1, cex = 0.65,
      col = c("black","red","green","blue"))
```

```
######################## função de sobrevivência S(x)
########################3
Sx= function(x)(0.1e1 - exp(-b / x)) ^ a
```

```
a=2
b=1
plot(Sx, xlim = c(0,5), ylim = c(0,1.4), col = "black")
a=1.5
plot(Sx, add = T, xlim = c(0,5), ylim = c(0,1.4), col = "red")
a=0.5
plot(Sx, add = T, xlim = c(0,5), ylim = c(0,1.4), col = "green")
```

```
a=1
plot(Sx, add = T, xlim = c(0,5), ylim = c(0,1.4), col = "blue")
legend(x = c(3,5),y = c(0.7,1.2),
      legend = c(expression(paste(beta=="1.0"," ", alpha==2.0)),
            expression(paste(beta=="1.0"," ", alpha==1.5)),
            expression(paste(beta=="1.0"," ", alpha==0.5)),
            expression(paste(beta=="1.0"," ", alpha==1.0))),
      lty =1, cex = 0.65,
      col = c("black","red","green","blue"))


##### alpha=c e beta=d #### função de risco h(y) ####

hx= function(x) a * b * exp(-b / x) / x ^ 2 / (0.1e1 - exp(-b / x))


a=2
b=1
plot(hx, xlim = c(0,5), ylim = c(0,1.4), col = "black")
a=1.5
plot(hx, add = T, xlim = c(0,5), ylim = c(0,1.4), col = "red")
a=0.5
plot(hx, add = T, xlim = c(0,5), ylim = c(0,1.4), col = "green")
a=1
plot(hx, add = T, xlim = c(0,5), ylim = c(0,1.4), col = "blue")
legend(x = c(3,5),y = c(0.7,1.2),
      legend = c(expression(paste(beta=="1.0"," ", alpha==2.0)),
            expression(paste(beta=="1.0"," ", alpha==1.5)),
            expression(paste(beta=="1.0"," ", alpha==0.5)),
            expression(paste(beta=="1.0"," ", alpha==1.0))),
      lty =1, cex = 0.65,
      col = c("black","red","green","blue"))

warnings()
```

```
hx = function(x) lambda ^ alpha * x ^ (alpha - 1) * alpha * exp(lambda ^ alpha * x ^
alpha)
lambda=1
alpha=0.5
plot(hx, xlim = c(0,2), ylim = c(0,2), col = "green")
alpha=0.75
plot(hx, add = T, xlim = c(0,2), ylim = c(0,2), col = "red")
alpha=1
plot(hx, add = T, xlim = c(0,2), ylim = c(0,2), col = "blue")
alpha=1.5
plot(hx, add = T, xlim = c(0,2), ylim = c(0,2), col = "black")
alpha=1.75
plot(hx, add = T, xlim = c(0,2), ylim = c(0,2), col = "purple")
legend(x = c(1,2),y = c(0,1),
      legend = c(expression(paste(lambda=="1.0"," ", alpha==0.5)),
            expression(paste(lambda=="1.0"," ", alpha==0.75)),
            expression(paste(lambda=="1.0"," ", alpha==1.0)),
            expression(paste(lambda=="1.0"," ", alpha==1.5)),
            expression(paste(lambda=="1.0"," ", alpha==1.75))),
      lty =1, cex = 0.65,
      col = c("green","red","blue","black","purple"))




rm(list=ls(all=TRUE))

n<-10
amostra<-500
```

```
alpha=0.5
beta=1.0


alfa.hat <- double(amostra)
beta.hat <- double(amostra)
sd.alfa.hat <- double(amostra)
sd.beta.hat <- double(amostra)
a1 <- double(amostra) ## alpha
a2 <- double(amostra) ## sd(alpha)
b1 <- double(amostra) ## beta
b2 <- double(amostra) ## sd(beta)
ICONFai <- double(amostra)
ICONFas <- double(amostra)
ICONFbi <- double(amostra)
ICONFbs <- double(amostra)
prob_cob_a_mv <- rep(1,times=amostra)
prob_cob_b_mv <- rep(1,times=amostra)
Vicio.a <- double(amostra)
Vicio.b <- double(amostra)
EQM.a <- double(amostra)
EQM.b <- double(amostra)


require(maxLik)



ll1<-function(theta,x)

{

  alpha=theta[1]
```

```
  beta=theta[2]

  if(theta[1]<=0)return(-Inf)
  if(theta[2]<=0)return(-Inf)

  logL=  (n*log(alpha) + n*log(beta) - 0.2e1 * n * log(x) - beta * n / x + (alpha - 0.1e1) *
n *log(0.1e1 - exp(-beta / x)))

  if(is.na(logL)==TRUE)
  {
    return(-Inf)
  }



  else
  {
    return(logL)
  }
}

## "NR" (for Newton-Raphson), "BFGS" (for Broyden-Fletcher-Goldfarb-Shanno)
## "BFGSR" (for the BFGS algorithm implemented in R), "BHHH" (for Berndt-Hall-
Hall-Hausman)
## "SANN" (for Simulated ANNealing), "CG" (for Conjugate Gradients)
## "NM" (for Nelder-Mead). Lower-case letters (such as "NR" for Newton-Raphson) are
allowed


X=NULL

X<-matrix(0,nrow=amostra,ncol=n)


for (v in 1:amostra)
```

```
{


 x=c()
 for(i in 1:n)
 {
  u = runif(1)
  f = function(x)
  {
   ((( 0.1e1 - (0.1e1 - exp(-beta / x)) ^ alpha)) - u)
  }
  x[i] = uniroot(f=f,interval=c(0,150))$root
 }


 ml <- maxLik(ll1, x=x, start=c(alpha,beta), method="NR")


 fit=ml$estimate;sd=sqrt(solve(-ml$hessian))



 while (fit[1]== "NaN" | fit[2]== "NaN" | sd[1]== "NaN" | sd[4]=="NaN")

 {

  x=c()
  for(i in 1:n)
  {
   u = runif(1)
   f = function(x)
   {
    ((( 0.1e1 - (0.1e1 - exp(-beta / x)) ^ alpha)) - u)
```

```
    }
    x[i] = uniroot(f=f,interval=c(0,150))$root
   }


   ml <- maxLik(ll1, x=x, start=c(alpha,beta), method="NR")


   fit=ml$estimate ; sd=sqrt(solve(-ml$hessian))


  }

  {

   X[v,] <- rbind(x)


   alfa.hat[v]=ml$estimate[1];beta.hat[v]=ml$estimate[2]

   sd.alfa.hat[v]=sd[1];sd.beta.hat[v]=sd[4]

   a1[v]=alfa.hat[v];a2[v]=sd.alfa.hat[v];b1[v]=beta.hat[v];b2[v]=sd.beta.hat[v]


   ICONFai[v]=a1[v]-1.96*a2[v]
   ICONFas[v]=a1[v]+1.96*a2[v]

   ICONFbi[v]=b1[v]-1.96*b2[v]
   ICONFbs[v]=b1[v]+1.96*b2[v]

   if(alpha<ICONFai[v]|alpha>ICONFas[v])  prob_cob_a_mv[v]=0  else
prob_cob_a_mv[v]=1
```

```
   if(beta<ICONFbi[v]|beta>ICONFbs[v])          prob_cob_b_mv[v]=0          else
prob_cob_b_mv[v]=1

   Vicio.a[v]=a1[v]-alpha
   Vicio.b[v]=b1[v]-beta
   EQM.a[v]=(a1[v]-alpha)^2
   EQM.b[v]=(b1[v]-beta)^2

  }

}


guarda=cbind(a1,a2,ICONFai,ICONFas,b1,b2,ICONFbi,ICONFbs,prob_cob_a_mv,pro
b_cob_b_mv,Vicio.a,Vicio.b,EQM.a,EQM.b)



prob_a=sum(prob_cob_a_mv)/amostra;prob_b=sum(prob_cob_b_mv)/amostra


##
mean(a1);mean(b1);sd(a2);sd(b2);mean(prob_a);mean(prob_b);mean(Vicio.a);mean(Vi
cio.b);mean(EQM.a);mean(EQM.b)

c(amostra,n, alpha, beta)
mean(a1);mean(b1);mean(a2);mean(b2);
mean(prob_a);mean(prob_b);
mean(Vicio.a);mean(Vicio.b);
mean(EQM.a);mean(EQM.b);
mean(ICONFai)
mean(ICONFas)
mean(ICONFbi)
mean(ICONFbs)
```

```
mean(prob_cob_a_mv)
mean((prob_cob_b_mv))
```