Taxas e Proporções

Thiago Tavares Lopes

17 abril 2025

Sumário

1 Distribuição Gompertz Unitária

A distribuição Gompertz Unitária (UGo), foi proposta em 2019 por Josmar Mazucheli, através como uma transformação do tipo $X = \exp(-Y)$ em que Y é da distribuição Gompertz. Já a distribuição Gompertz proposta pelo matemático e atuário Benjamin Gompertz em 1865.

Seja **X**, uma variável aleatória com suporte duplamente limitado $X \in (0, 1)$. Sua Função de densidade de probabilidade é dada por:

$$f(x|\mu;\sigma) = \mu\sigma x^{-(\sigma+1)} \exp\left[-\mu(x^{-\sigma} - 1)\right] \tag{1}$$

```r
dUGo<-function(x, mu=.5, sigma=1.2){

  fx<-mu*sigma * (x^-(sigma+1)) * exp(-mu*(x^(-sigma)-1))

  return(fx)

}
```

Seja, Fx seja uma função de densidade, a mesma deve seguir as seguintes propriedades:

Para testar

```r
integrate(dUGo,0,1)
```

```
## 1 with absolute error < 1.3e-05
```

Função Acumulada

$$f(x|\mu;\sigma) = exp[-\mu(x^{-\sigma} - 1)] \tag{2}$$

```r
pUGo<-function(q, mu=.5, sigma=1.2){

  cdf<-exp(-mu*(q^(-sigma)-1))

  return(cdf)

}
```

```r
pUGo(.25)
```

```
## [1] 0.1177707
```

```r
integrate(dUGo, 0, .25)
```

```
## 0.1177707 with absolute error < 1.3e-05
```

Função Quantílica

$$Q(p|\mu;\sigma) = \exp\left[-\frac{1}{\sigma}\log\left(\mu - \log p\right) - \log\mu\right] \tag{3}$$

```r
qUGo<-function(u, mu=.5, sigma=1.2)
{
   q<-((-log(u)/mu)+1)^(-1/sigma)

  return(q)
}

u=pUGo(.82)
qUGo(u)
```

```
## [1] 0.82
```

Geração de números aleatórios pelo menos da inversão.

```r
set.seed(123)
rUGO <- function(n, mu=.5, sigma=1.2) {
  u <- runif(n)
  y <- ((-log(u)/mu)+1)^(-1/sigma)
  return(y)
}

rUGO(100)
```

```
##   [1] 0.3526823 0.7230335 0.4254981 0.8309660 0.9080196 0.1935012 0.5037446
##   [8] 0.8428969 0.5202596 0.4557164 0.9319488 0.4535946 0.6189068 0.5356792
##  [15] 0.2398379 0.8524724 0.3284380 0.1899844 0.3764162 0.9284837 0.8392161
##  [22] 0.6321042 0.5880651 0.9905220 0.6005037 0.6460736 0.5149934 0.5517584
##  [29] 0.3536084 0.2690369 0.9412551 0.8557070 0.6302673 0.7305031 0.1695836
##  [36] 0.4695777 0.6929572 0.3110211 0.3706583 0.3199700 0.2663067 0.4289693
##  [43] 0.4284560 0.4008965 0.2723847 0.2637603 0.3207959 0.4617986 0.3400550
##  [50] 0.8001762 0.1937717 0.4464406 0.7341455 0.2527530 0.5271287 0.3051817
##  [57] 0.2564648 0.6879289 0.8462744 0.4043002 0.6083477 0.2340989 0.4100886
##  [64] 0.3449679 0.7510107 0.4504901 0.7460469 0.7485638 0.7293230 0.4449281
##  [71] 0.6890642 0.5790073 0.6475619 0.1004900 0.4679402 0.3132081 0.4075555
##  [78] 0.5660637 0.3906357 0.2455124 0.3269945 0.6108221 0.4309082 0.7229203
##  [85] 0.2397959 0.4417840 0.9754203 0.8437080 0.8353175 0.2863106 0.2585294
##  [92] 0.5983527 0.3856854 0.6013754 0.3719524 0.2939447 0.7168421 0.2331994
##  [99] 0.4623325 0.4922632
```

```r
# set.seed(123)
# x_sample <- runif(100, min = 0.01, max = 0.99)
#
# theta<-c(2,3)
#
# log_lik(theta, x_sample)
```

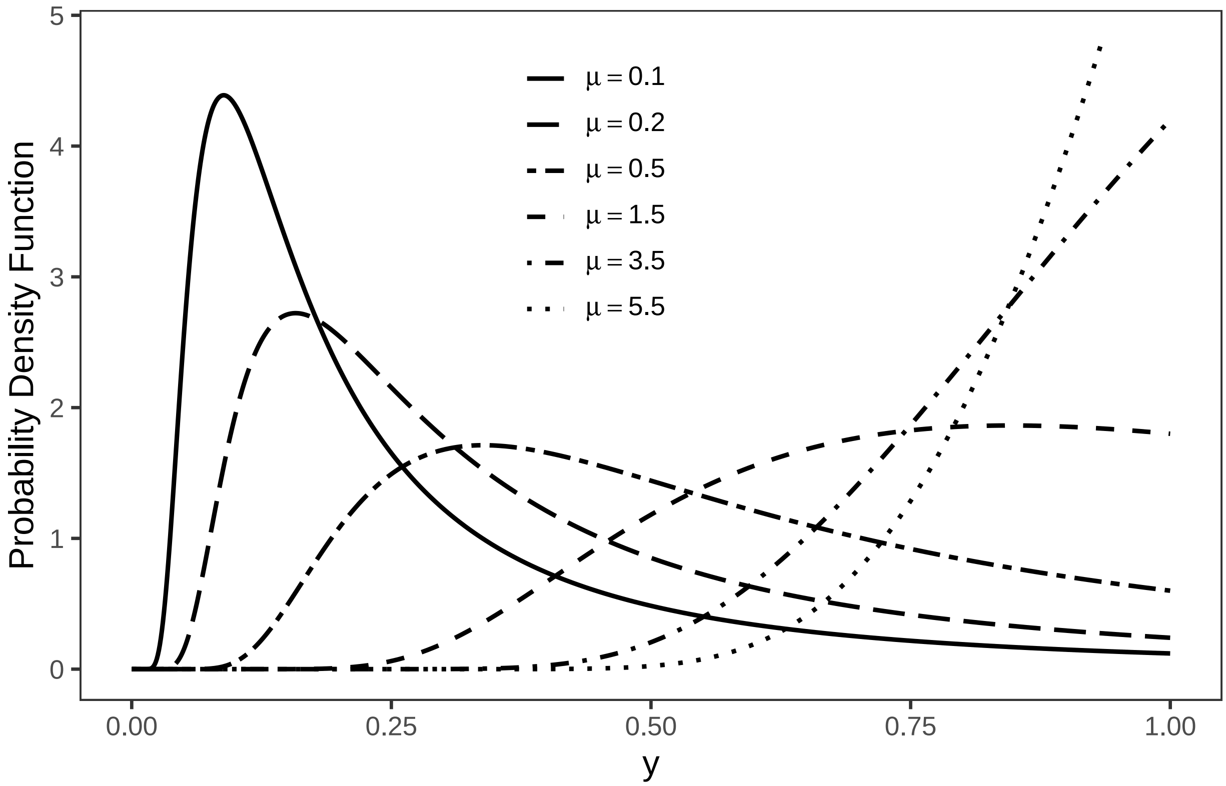
Log-Verossimilhança

$$\ell(\mu;\sigma|X) = \sum_{i=1}^{n}\left[\log(\mu) + \log(\sigma) - (\sigma+1)\log(x) - \mu(x^{-\sigma} - 1)\right] \tag{4}$$

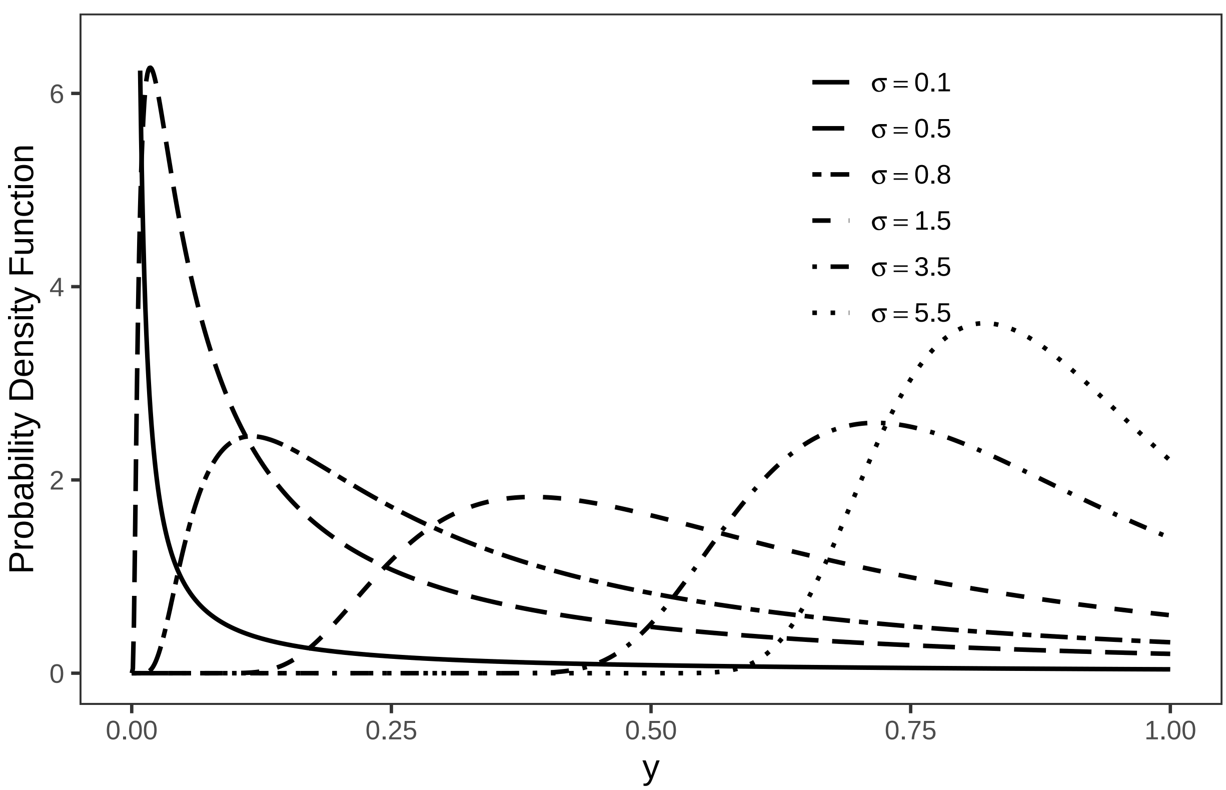
```r
log_lik <- function(theta, x) {

  n <- length(x)
  mu<-theta[1]
  sigma<-theta[2]

  term1 <- n * log(mu)
  term2 <- n * log(sigma)
  term3 <- -(sigma + 1) * sum(log(x))
  term4 <- -mu * sum(x^(-sigma) - 1)

  ll <- term1 + term2 + term3 + term4
  return(ll)
}
```

```r
set.seed(123)

x<-rUGO(100)

rUGO <- function(n, mu=.5, sigma=1.2) {
  u <- runif(n)
  y <- ((-log(u)/mu)+1)^(-1/sigma)
  return(y)
}

parametros_inicial <- c(0.5, 1.6)  # Valores iniciais para mu e beta

optim_result <- optim(parametros_inicial, log_lik, x=x, hessian = T,
                        method="BFGS",
                        control = list(fnscale=-1))


print(optim_result$par)
```

```
## [1] 0.497407 1.208987
```



μ = 0.1
μ = 0.2
μ = 0.5
μ = 1.5
μ = 3.5
μ = 5.5
Probability Density Function
y
0.00
0.25
0.50
0.75
1.00
0
1
2
3
4
5



σ = 0.1
σ = 0.5
σ = 0.8
σ = 1.5
σ = 3.5
σ = 5.5
Probability Density Function
y
0.00
0.25
0.50
0.75
1.00
0
2
4
6