# Elementos de Computação em R para Inferência Estatística

PJ

2 de fevereiro de 2012

## 1 Inferência para distribuição Gamma

### 1.1 Densidade

**Parametrização 1:** Parametrização com $\alpha = shape = a$ e $\beta = scale = s$ ($rate = 1/scale$) utilizada no R.

$a = 0$ define uma fdp trivial com toda massa de probabilidade em zero.

*Note that for smallish values of shape (and moderate scale) a large parts of the mass of the Gamma distribution is on values of x so near zero that they will be represented as zero in computer arithmetic. So rgamma can well return values which will be represented as zero. (This will also happen for very large values of scale since the actual generation is done for scale=1.)*

$$
\begin{aligned}
Y &\sim \mathrm{G}(\alpha, \beta) \\
f(y|\alpha,\beta) &= \frac{1}{\Gamma(\alpha)\,\beta^{\alpha}}\; y^{\alpha-1}\; \exp\{-y/\beta\} \quad y \geq 0 \;\; \alpha \geq 0 \;\; \beta > 0 \\
E[Y] &= \alpha\beta \quad Var[Y] = \alpha\beta^2 \\
L((\alpha,\beta)|y) &= \left(\frac{1}{\Gamma(\alpha)\,\beta^{\alpha}}\right)^{n} \prod_{i=1}^{n} y_i^{\alpha-1}\; \exp\{-y_i/\beta\} \\
l((\alpha,\beta)|y) &= n\left(-\log(\Gamma(\alpha)) - \alpha\log(\beta) + (\alpha-1)\overline{\log(y)} - \bar{y}/\beta\right)
\end{aligned}
$$

Função escore

$$
\begin{cases}
\frac{dl}{d\beta} &= n\left(-\frac{\alpha}{\beta} + \frac{\bar{y}}{\beta^2}\right) \\
\frac{d}{d\alpha} &= n\left(-\frac{\Gamma'(\alpha)}{\Gamma(\alpha)} - \log(\beta) + \overline{\log y}\right)
\end{cases}
$$

Igualando as funções a zero, da primeira equação temos $\hat{\alpha}\hat{\beta} = \bar{y}$. Substituindo $\beta$ por $\hat{\beta}$ a segunda expressão é escrita como:

$$n\left(-\frac{\Gamma'(\alpha)}{\Gamma(\alpha)} - \log(\bar{y}/\hat{\alpha}) + \overline{\log y}\right) = 0$$

MLE é solução conjunta de

$$\begin{cases} \overline{\log y} - \log\beta + & = \psi(\bar{y}/\beta) \\ \hat{\alpha}\hat{\beta} & = \bar{y} \end{cases}$$

em que $\psi(t) = \frac{d}{dt}\log(\Gamma(t)) = \frac{\Gamma'(t)}{\Gamma(t)}$ (função `digamma()`).

**Parametrização 2:** Rizzo (2008) e outros autores (e parametrização original do S e Splus) utilizam a reparametrização $r = \alpha, \lambda = 1/\beta$ em que $\lambda$ é o parâmetro de taxa (*rate*).

$$\begin{aligned} Y &\sim \mathrm{G}(r,\lambda) \\ f(y|r,\lambda) &= \frac{\lambda^r}{\Gamma(r)}\, y^{r-1}\ \exp\{-\lambda y\} \quad y \geq 0\ \ r \geq 0\ \ \lambda > 0 \\ E[Y] &= r/\lambda \quad Var[Y] = r/\lambda^2 \\ L((r,\lambda)|y) &= \left(\frac{\lambda^r}{\Gamma(r)}\right)^n \prod_{i=1}^{n} y_i^{r-1}\ \exp\{-\lambda y_i\} \\ l((r,\lambda)|y) &= n\left(r\log(\lambda) - \log(\Gamma(r)) + (r-1)\overline{\log(y)} - \lambda\bar{y}\right) \end{aligned}$$

Função escore

$$\begin{cases} \frac{dl}{d\lambda} & = n\left(\frac{r}{\lambda} - \bar{y}\right) \\ \frac{d}{dr} & = n\left(\log(\lambda) - \frac{\Gamma'(r)}{\Gamma(r)} + \overline{\log y}\right) \end{cases}$$

Igualando as funções a zero, da primeira equação temos $\hat{\lambda} = \hat{r}/\bar{y}$. Substituindo $\lambda$ por $\hat{\lambda}$ a segunda expressão é escrita como:

$$n\left(\log\frac{\hat{r}}{\bar{y}} + \overline{\log y} - \frac{\Gamma'(r)}{\Gamma(r)}\right) = 0$$

MLE é solução conjunta de

$$\begin{cases} \log\lambda + \overline{\log y} & = \psi(\lambda\bar{y}) \\ \bar{y} & = r/\lambda \end{cases}$$

em que $\psi(t) = \frac{d}{dt}\log(\Gamma(t)) = \frac{\Gamma'(t)}{\Gamma(t)}$ (função `digamma()`).

**Parametrização 3:** Aitkin (2010) menciona ainda duas parametrizações sendo a primeira, a mesma do R, citada como a mais usual e uma segunda parametrizada por $r = \alpha$ e $\mu = \alpha\beta$.

Esta parametrização tem propriedades interessantes para inferência. A primeira é a ortogonalidade entre $r$ e $\mu$ na matriz de informação. Além disto em geral $\mu$ é o usualmente o parâmetro de interesse para inferências e $r$ é um parâmetro *nuisance*. Finalmente a parametrização é adequada para modelagem estatística na qual usualmente se propõe um modelo de regressão para média $\mu$, como por exemplo em modelos lineares generalizados (GLM).

$$
\begin{aligned}
Y &\sim \mathrm{G}(r, \mu) \\
f(y|r,\mu) &= \frac{r^r}{\Gamma(r)\,\mu^r}\; y^{r-1}\; \exp\{-ry/\mu\} \quad y \geq 0 \;\; r \geq 0 \;\; \mu \geq 0 \\
E[Y] &= \mu \quad Var[Y] = \mu^2/r \\
L((r,\mu)|y) &= \left(\frac{r^r}{\Gamma(r)\mu^r}\right)^n \prod_{i=1}^{n} y_i^{r-1}\; \exp\{-ry_i/\mu\} \\
l((r,\mu)|y) &= n\left(r(\log(r) - \log(\mu)) - \log(\Gamma(r)) + (r-1)\overline{\log(y)} - \frac{r}{\mu}\bar{y}\right)
\end{aligned}
$$

Função escore

$$
\begin{cases}
\frac{dl}{d\mu} &= n\left(-\frac{r}{\mu} + \frac{r\bar{y}}{\mu^2}\right) \\
\frac{d}{dr} &= n\left(\log(r) + 1 - \log(\mu) - \frac{\Gamma'(r)}{\Gamma(r)} + \overline{\log y} + \frac{\bar{y}}{\mu}\right)
\end{cases}
$$

Igualando as funções a zero, da primeira equação temos $\hat{\mu} = \bar{y}$. Substituindo $\mu$ por $\hat{\mu}$ a segunda expressão é escrita como:

$$
n\left(\log(\hat{r}) + 1 - \log(\bar{y}) - \frac{\Gamma'(\hat{r})}{\Gamma(\hat{r})} + \overline{\log y} + 1\right) = 0
$$

MLE é solução conjunta de

$$
\begin{cases}
\log\hat{r} - \psi(\hat{r}) &= \log\bar{y} - \overline{\log y} - 2 \\
\hat{\mu} &= \bar{y}
\end{cases}
$$

em que $\psi(t) = \frac{d}{dt}\log(\Gamma(t)) = \frac{\Gamma'(t)}{\Gamma(t)}$ (função `digamma()`).

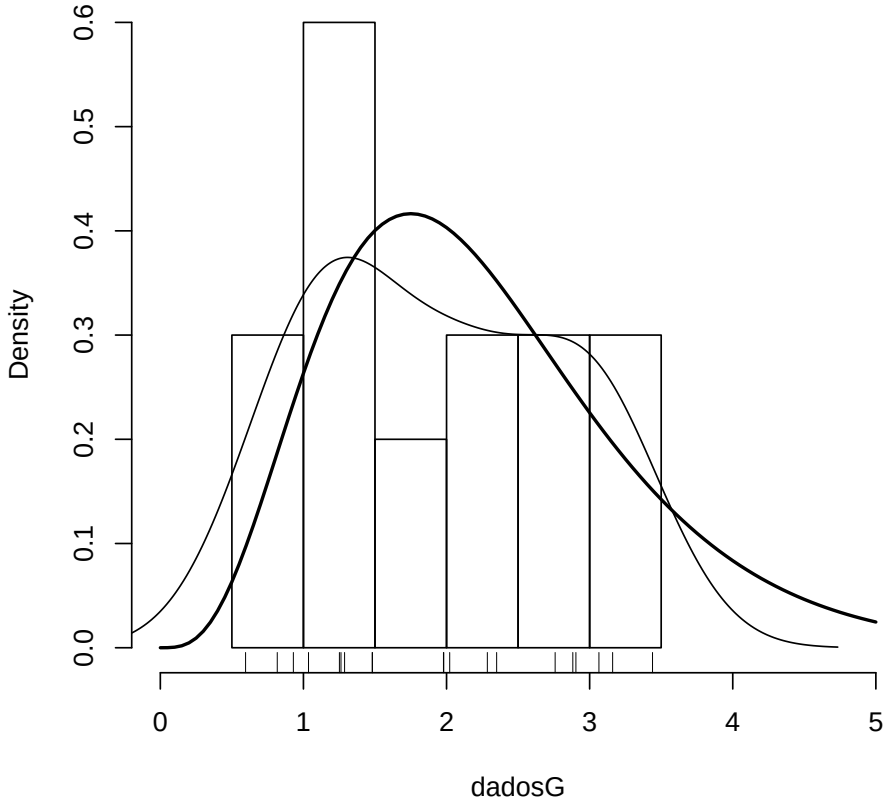
## 1.2 Códigos R

Não é possível obter estimadores de M.V. em forma analítica fechada para os parâmetros da distribuição Gama sendo necessário o uso de métodos numéricos. Algumas abordagens possíveis:

- solução de equações (fc escores)
  - conjunta para dois parâmetros, com resulução numérica duas equações
  - por substituição, com resolução numérica sistema de uma equação
- maximização de função de log-verossimilhança
  - bidimensional (2 parâmetros)
  - unidimensional (1 parâmetro - verossimilhança concentrada/perfilhada)

Dados (simulados)

```
> set.seed(201107)
> dadosG <- rgamma(20, shape = 4.5, rate=2)
> hist(dadosG, prob=T, xlim=c(0,5), main="", panel.first=lines(density(dadosG)));r
> curve(dgamma(x, shape = 4.5, rate=2), 0, 5, add=T, lwd=2)
```



```
> ## amostra = list(media, media.logs)
> am <- list(media=mean(dadosG),
+ media.logs = mean(log(dadosG)), n=length(dadosG))
```

Estimação utilizando métodos numéricos

**(i) Equações de estimação** (fç escore)

$$\begin{cases} \frac{dl}{d\lambda} & = n\left(\frac{r}{\lambda} - \bar{y}\right) = 0 \\ \frac{dl}{dr} & = n\left(\log(\lambda) - \frac{\Gamma'(r)}{\Gamma(r)} + \overline{\log(y)}\right) = 0 \end{cases}$$

Definimos uma função em R com sistema de equações.

```
> Umat <- function(par, amostra){
+     lambda <- par[1] ; r = par[2]
+     UlamUr <- local({
+           c(n * ((r/lambda) - media),
+             n*(log(lambda) - digamma(lambda*media) + media.logs))
+     }, env=amostra)
+     return(UlamUr)
+ }
> #Umat(c(2,3), am)
> require(rootSolve)
> multiroot(f=Umat, start=c(1,1), amostra=am)

$root
[1] 2.408413 4.692590

$f.root
[1] -9.365272e-07  1.964257e-07

$iter
[1] 6

$estim.precis
[1] 5.664765e-07
```

Simplificando para 1 equação

```
> Ulam <- function(lambda, amostra){
+         with(amostra, digamma(lambda*media) - media.logs - log(lambda))
+ }
> (lambda.est <- uniroot(Ulam, lower=1e-3, upper=1e5, amostra=am)$root)
```

```
[1] 2.40842
```

```
> (r.est <- lambda.est * am$media)
```

```
[1] 4.692604
```

**(ii) Maximização** da função de log-Verossimilhança

$$l(\theta) = l(r, \lambda) = nr \log \lambda - n \log \Gamma(r) + (r-1) \sum_{i=1}^{n} \log y_i - \lambda \sum_{i=1}^{n} y_i = \\ = n[r \log \lambda - \log \Gamma(r) + (r-1)\overline{\log(y)} - \lambda \bar{y}] \tag{1}$$

Definindo função a 2 parâmetros

```
> neglLik <- function(par, amostra, modelo=2){
+         if(modelo == 1){
+     alpha <- par[1]; beta <- par[2]
+     ll <- with(amostra, n*(-alpha*log(beta) - log(gamma(alpha)) +
+           (alpha-1) * media.logs - media/beta))
+     }
+   if(modelo == 2){
+     r <- par[1]; lambda <- par[2]
+     ll <- with(amostra, n*(r*log(lambda) - log(gamma(r)) +
+           (r-1) * media.logs - lambda * media))
+     }
+   if(modelo == 3){
+     r <- par[1]; mu <- par[2]
+     ll <- with(amostra, n*(r*(log(r) - log(mu)) - log(gamma(r)) +
+           (r-1) * media.logs - (r/mu) * media))
+     }
+   return(-ll)
+ }
> (mod1 <- optim(c(1,1), neglLik, amostra=am, method="L-BFGS-B", modelo=1,
+           lower=c(0,0), upper=c(Inf, Inf))$par)
```

```
[1] 4.6924351 0.4152265
```

```
> (mod2 <- optim(c(1,1), neglLik, amostra=am, method="L-BFGS-B", modelo=2,
+           lower=c(0,0), upper=c(Inf, Inf))$par)
```

```
[1] 4.692593 2.408415

> (mod3 <- optim(c(1,1), neglLik, amostra=am, method="L-BFGS-B", modelo=3,
+           lower=c(0,0), upper=c(Inf, Inf))$par)

[1] 4.692591 1.948416

> 1/mod1[2]

[1] 2.408324

> prod(mod1)

[1] 1.948423
```

Comentários sobre reparametrização

Reparametrização para garantir validade do espaço paramétrico

Nesta caso como ambos parâmetros devem ser não negativos em todas as parametrizações, podemos redefinir a função com a opção de que os parâmetros sejam fornecidos em escala logaritmica. Note que isto exclui o valor nulo do expaço paramétrico.

```
> neglLik <- function(par, amostra, modelo=2, logpar=FALSE){
+     if(logpar) par <- exp(par)
+         if(modelo == 1){
+     alpha <- par[1]; beta <- par[2]
+     ll <- with(amostra, n*(-alpha*log(beta) - log(gamma(alpha)) +
+           (alpha-1) * media.logs - media/beta))
+     }
+   if(modelo == 2){
+     r <- par[1]; lambda <- par[2]
+     ll <- with(amostra, n*(r*log(lambda) - log(gamma(r)) +
+           (r-1) * media.logs - lambda * media))
+     }
+   if(modelo == 3){
+     r <- par[1]; mu <- par[2]
+     ll <- with(amostra, n*(r*(log(r) - log(mu)) - log(gamma(r)) +
+           (r-1) * media.logs - (r/mu) * media))
+     }
+   return(-ll)
+ }
> (mod1r <- optim(log(c(1,1)), neglLik, amostra=am, modelo=1, log=T)$par)
```

```
[1]  1.5460915 -0.8791401

> rbind(mod1, mod1r)

          [,1]       [,2]
mod1  4.692435  0.4152265
mod1r 1.546091 -0.8791401

> (mod2r <- optim(log(c(1,1)), neglLik, amostra=am, modelo=2, log=T)$par)

[1] 1.5458999 0.8788993

> rbind(mod2, exp(mod2r))

         [,1]     [,2]
mod2 4.692593 2.408415
     4.692192 2.408247

> (mod3r <- optim(log(c(1,1)), neglLik, amostra=am, modelo=3, , log=T)$par)

[1] 1.5458160 0.6670942

> rbind(mod3, exp(mod3r))

         [,1]     [,2]
mod3 4.692591 1.948416
     4.691798 1.948567
```

Reparametrização para interpretabilidade dos parâmetros

Reparametrização ortogonalização

Invariância dos estimadores de M.V. e transformação direta de limites de intervalos de confiança baseados na verossimilhança/deviance.

Reduzindo a dimensionalidade (verossimilhança concentrada)

```
> neglLik1 <- function(par, amostra, modelo=2, logpar=FALSE){
+   if(logpar) par <- exp(par)
+   if(modelo == 1){
+     alpha <- amostra$media/par; beta <- par
+     ll <- with(amostra, n*(-log(gamma(alpha)) - alpha*log(beta)  +  (alpha-1) *
+   }
+   if(modelo == 2){
+     r <- par * amostra$media; lambda <- par
```

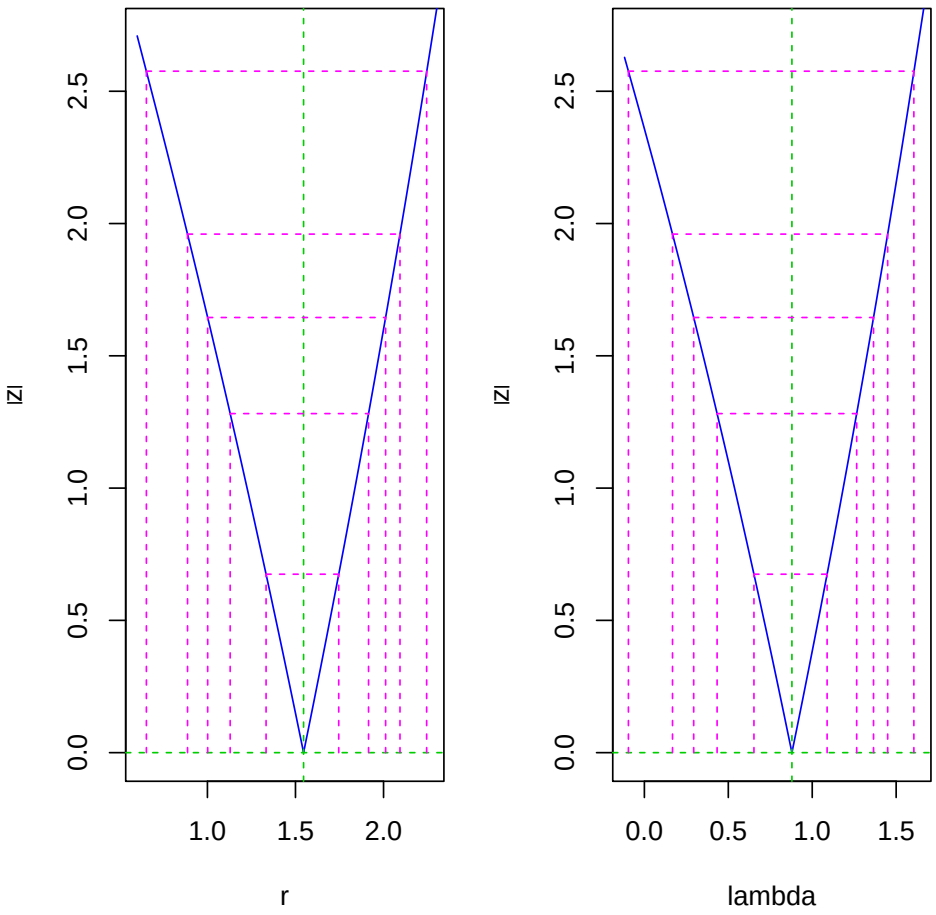
```
+     ll <- with(amostra, n*(r*log(lambda) - log(gamma(r)) +  (r-1) * media.logs -
+   }
+   if(modelo == 3){
+     ll <- with(amostra,
+            n*(par*(log(par)-log(media))-log(gamma(par))+(par-1)*media.logs-par))
+   }
+   return(-ll)
+ }
> (est1 <- optimize(neglLik1, lower=0.001, upper=100, amostra=am, modelo=1)$min)

[1] 0.4152092

> (est2 <- optimize(neglLik1, lower=0.001, upper=100, amostra=am, modelo=2)$min)

[1] 2.408401

> 1/est1

[1] 2.408424

> (est3 <- optimize(neglLik1, lower=0.001, upper=100, amostra=am, modelo=3)$min)

[1] 4.692589

> am$media/est1

[1] 4.692613

> am$media*est2

[1] 4.692568
```

Funções "facilitadoras" já prontas e disponíveis.

```
> require(stats4)
> args(mle)

function (minuslogl, start = formals(minuslogl), method = "BFGS",
    fixed = list(), nobs, ...)
NULL
```

```
> nlLik <- function(r, lambda){
+         lambda <- exp(lambda); r <- exp(r)
+   ll <- am$n*(r*log(lambda) - log(gamma(r)) +  (r-1) * am$media.logs - lambda *
+   return(-ll)
+ }
> est.mle1 <- mle(nlLik, start=list(r=log(2), lambda=log(2)))
> summary(est.mle1)

Maximum likelihood estimation

Call:
mle(minuslogl = nlLik, start = list(r = log(2), lambda = log(2)))

Coefficients:
        Estimate Std. Error
r      1.5459802  0.3056498
lambda 0.8789633  0.3226096

-2 log L: 49.52223

> logLik(est.mle1)

'log Lik.' -24.76112 (df=2)

> AIC(est.mle1)

[1] 53.52223

> coef(est.mle1); exp(coef(est.mle1))

        r    lambda
1.5459802 0.8789633

       r   lambda
4.692569 2.408402

> vcov(est.mle1)

                r     lambda
r      0.09342182 0.09342182
lambda 0.09342182 0.10407696
```

```
> confint(est.mle1, level=0.95)

Profiling...
           2.5 %   97.5 %
r      0.8870140 2.093177
lambda 0.1675541 1.449452

> par(mfrow=c(1,2))
> plot(profile(est.mle1))
```




Uma função mais conveniente para visualização: Deviance

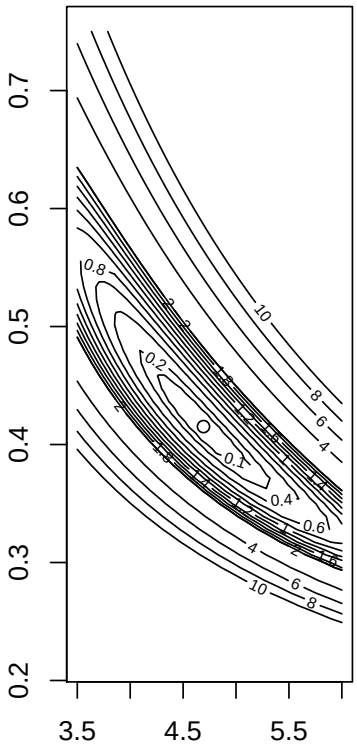
$$D(\theta) = -2[l(\theta) - l(\hat{theta})]$$

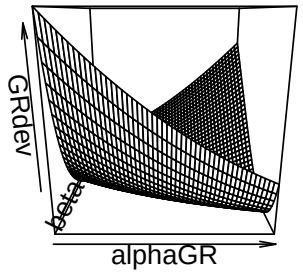
Definindo uma função deviance genérica

```
> devFun <- function(theta, est, llFUN, ...){
+ #  return(2 * (neglLik(theta, ...) - neglLik(est, ...)))
+   return(2 * (llFUN(theta, ...) - llFUN(est, ...)))
+ }

> (ests <- c(r.est, lambda.est))

[1] 4.692604 2.408420

> devFun(c(4,2.5), est=mod1, llFUN=neglLik, amostra=am, modelo=2)

[1] -171.0314

> devFun(2.5, est=mod2[2], llFUN=neglLik1, amostra=am)

[1] 0.01508588
```

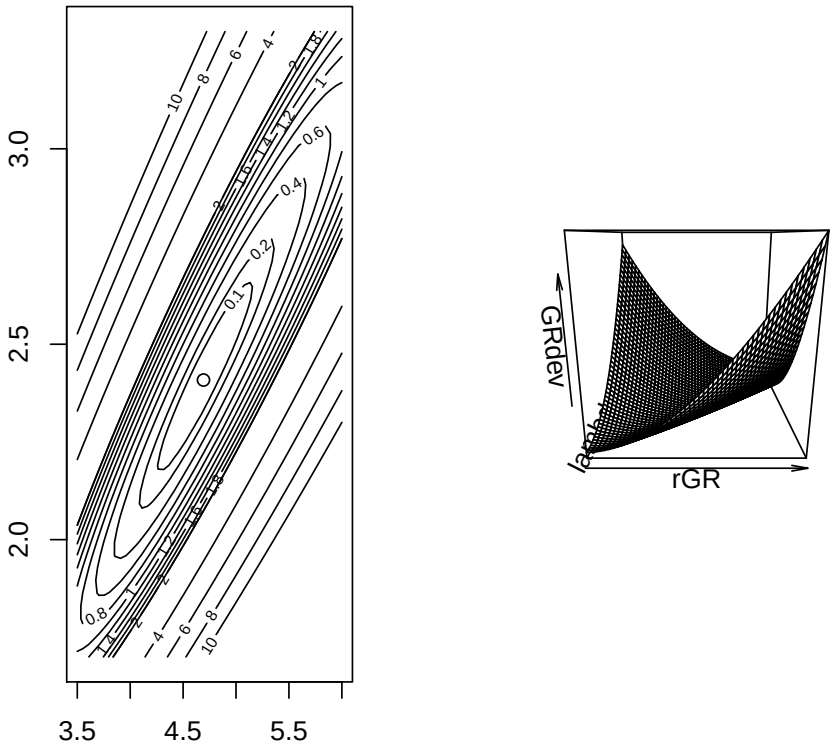
Gráficos da função de verossimilhança bidimensional

```
> Ofun <- Vectorize(function(x,y, ...) devFun(c(x,y), ...))
> Ofun(4, 2.5, est=ests, llFUN=neglLik, amostra=am, modelo=2)

[1] 3.579439

> alphaGR <- seq(3.5,6, len=50)
> betaGR <- seq(0.22,0.75, len=50)
> GRdev <- outer(alphaGR, betaGR, FUN = Ofun, est=mod1, llFUN=neglLik, amostra=am,

> par(mfrow=c(1,2))
> contour(alphaGR,betaGR, GRdev, levels=c(0.1,(1:10)/5, 2*(1:5))); points(t(mod1))
> persp(alphaGR, betaGR, GRdev)
```

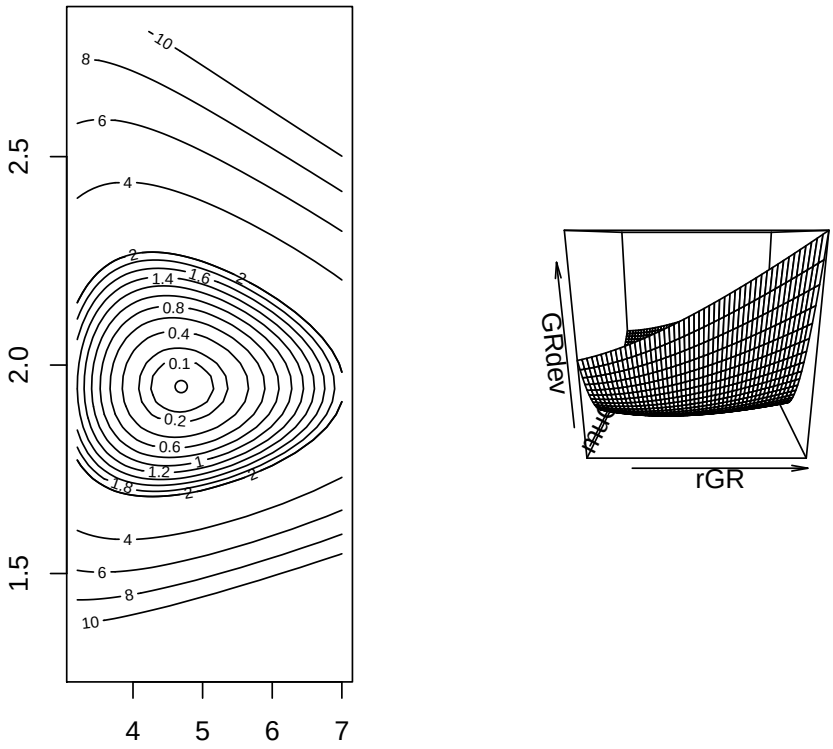



```
> rGR <- seq(3.5,6, len=50)
> lambdaGR <- seq(1.7,3.3, len=50)
> GRdev <- outer(rGR, lambdaGR, FUN = Ofun, est=ests, llFUN=neglLik, amostra=am)

> par(mfrow=c(1,2))
> contour(rGR, lambdaGR, GRdev, levels=c(0.1,(1:10)/5, 2*(1:5))); points(t(mod2))
> persp(rGR, lambdaGR, GRdev)
```




```
> rGR <- seq(3.2,7, len=50)
> muGR <- seq(1.3,2.8, len=50)
> GRdev <- outer(rGR, muGR, FUN = Ofun, est=mod3, llFUN=neglLik, amostra=am, model

> par(mfrow=c(1,2))
> contour(rGR,muGR, GRdev, levels=c(0.1,(1:10)/5, 2*(1:5))); points(t(mod3))
> persp(rGR, muGR, GRdev)
```

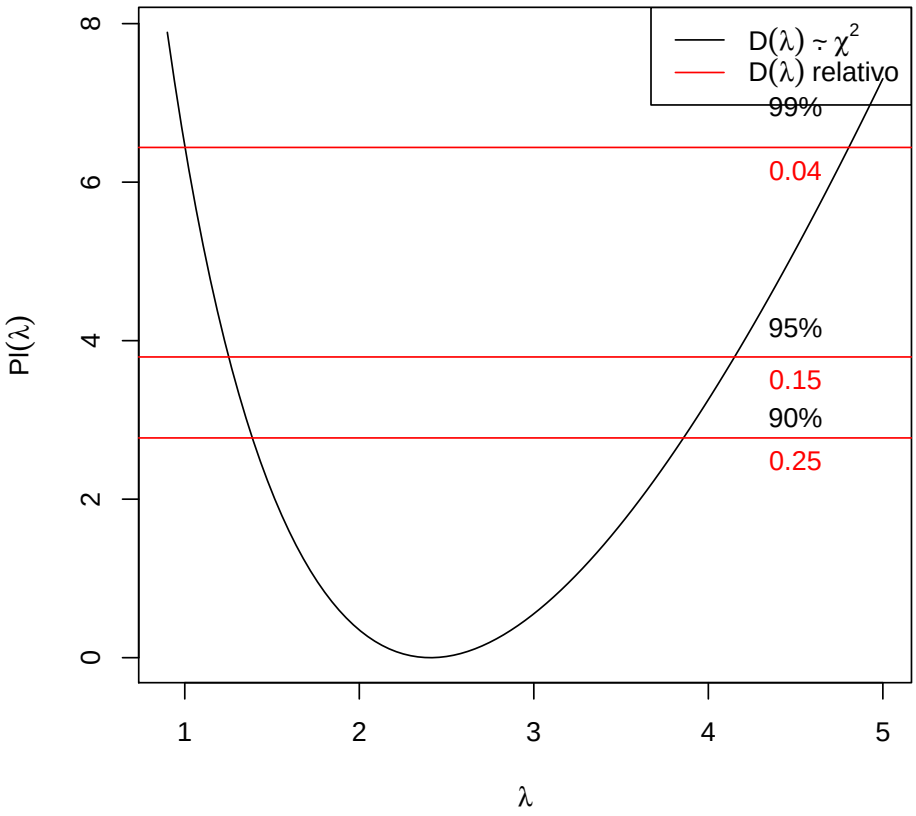



Gráficos similares poderiam ser feitos utilizando a reparametrização em excala logaritmica que reduziria a assimetria na função.

Unidimensional, verossimilhança perfilhada

```
> #devFun(lambdaGR, est=ests[2], llFUN=neglLik1, amostra=am)
> curve(devFun(x, est=ests[2], llFUN=neglLik1, amostra=am), from=0.9, to=5,
+         xlab=expression(lambda), ylab=expression(Pl(lambda)))
> ## cortes definidos por probabilidades
> (corteICs <- qchisq(c(0.90, 0.95, 0.99), df=1))

[1] 2.705543 3.841459 6.634897

> text(4.5, corteICs, c("90%", "95%", "99%"), pos=3)
> ## cortes definidos por percentuais da MV
> (corteICs <- -2*log(c(0.25, 0.15, 0.04)))
```

```
[1] 2.772589 3.794240 6.437752

> abline(h=corteICs, col=2)
> text(4.5, corteICs, c(0.25, 0.15, 0.04), pos=1, col=2)
> legend("topright", c(expression(D(lambda)~tilde(.)~chi^2),
+                      expression(D(lambda)~relativo)), lty=1, col=1:2)
```




Os gráficos e funções poderiam ser obtidos em escala logaritimoca para os parâmetros e os intervalos por transformação dos limitesdos IC's baseados em verossimilhança, dada a propriedade de invariância.

Aproximação Quadrática - Aproximação de Taylor de 2a ordem da função verossimilhança e deviance ao redor do estimador de M.V. $\hat{\theta}$.

$$
\begin{aligned}
l(\theta) &= l(\hat{\theta}) + (\theta - \hat{\theta})l'(\hat{\theta}) + \frac{1}{2}(\theta - \hat{\theta})^2 l''(\hat{\theta}) \\
&= l(\hat{\theta}) + \frac{1}{2}(\theta - \hat{\theta})^2 l''(\hat{\theta})
\end{aligned}
$$

Uma vez que pela definição de E.M.V. $l'(\hat{\theta}) = 0$. A approximação da função deviance fica:

$$
\begin{aligned}
D(\theta) &= -2[l(\theta) - l(\hat{\theta})] \\
&\approx -2[l(\hat{\theta}) + \frac{1}{2}(\theta - \hat{\theta})^2 l''(\hat{\theta}) - l(\hat{\theta})] \\
&= -(\theta - \hat{\theta})^2 l''(\hat{\theta})
\end{aligned}
$$

As expressões consideram $\theta$ escalar, mas a extensão para um vetor de parâmetros é trivial.