

CE-227: Inferência Bayesiana – 1ª Avaliação Semanal (28/02/2014)

GRR: _____ **Nome:** _____ **Turma:** _____

1. Seja x_1, \dots, x_n uma a.a. de uma distribuição normal de média μ conhecida e variância σ^2 desconhecida. Considere a priori $[\sigma^2] \propto 1/\sigma^2$.

- (a) Obtenha a expressão da verossimilhança do modelo.
- (b) Obtenha a expressão da distribuição a posteriori.
- (c) É possível identificar a posteriori do modelo como alguma distribuição conhecida?
- (d) Considere que foi tomada a amostra dada pelos valores a seguir, e que $\mu = 10$. Obtenha a expressão da posteriori.

12,1 ; 8,7 ; 11,3 ; 9,2 ; 10,5 ; 9,7 ; 11,6

- (e) Indique (com comandos do R ou de alguma outra forma) como resumos pontuais e intervalares desta distribuição a posteriori poderiam ser obtidos para fins de inferência

Probabilidades

$$\mu_X = E[X] = \sum_i x_i P(X = x_i)$$

$$\sigma_X^2 = Var[X] = \sum_i (x_i - \mu_x)^2 P(X = x_i)$$

$$\mu_X = E[X] = \int x f_X(x) dx$$

$$\sigma_X^2 = Var[X] = \int (x - \mu_x)^2 f_X(x)$$

$$Y = g(X) \Rightarrow$$

$$f_Y(y) = f_X(g^{-1}(y)) \left| \frac{d}{dy} g^{-1}(y) \right|$$

Distribuições de Probabilidade

Distribuição	Probab./Densidade	Domínio	E[X]	Var[X]
$X \sim U(k)$	$\frac{1}{k}$	$x = 1, 2, \dots, k$	$\frac{\min(X)+\max(x)}{2}$	$\sqrt{\frac{(\max(X)-\min(X)+1)^2-1}{12}}$
$X \sim B(n, p)$	$\binom{n}{x} p^x (1-p)^{n-x}$	$x = 0, 1, 2, \dots, n$	np	$np(1-p)$
$X \sim HG(N, K, n)$	$\frac{\binom{K}{x} \binom{N-K}{n-x}}{\binom{N}{n}}$	$x = 0, 1, \dots, \min(K, n)$	np	$np(1-p) \frac{N-n}{N-1}$
$X \sim P(\lambda)$	$\frac{e^{-\lambda} \lambda^x}{x!}$	$x = 0, 1, 2, \dots$	λ	λ
$X \sim G(p)$	$(1-p)^x p$	$x = 0, 1, 2, \dots$	$\frac{1-p}{p}$	$\frac{(1-p)}{p^2}$
$X \sim BN(r, p)$	$\binom{x+r-1}{r-1} (1-p)^x p^r$	$x = 0, 1, 2, \dots$	$\frac{r(1-p)}{p}$	$\frac{r(1-p)}{p^2}$
$X \sim U[a, b]$	$\frac{1}{b-a}$	$a \leq x \leq b$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$
$X \sim N(\mu, \sigma^2)$	$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2\sigma^2}(x-\mu)^2\}$	$x \in (-\infty, \infty)$	μ	σ^2
$X \sim \text{HalfN}(0, \sigma^2)$	$\frac{\sqrt{2}}{\sigma\sqrt{\pi}} \exp\{-\frac{x^2}{2\sigma^2}\}$	$x \in (0, \infty)$	0	σ^2
$X \sim \text{LN}(\mu, \sigma^2)$	$\frac{1}{x\sqrt{2\pi\sigma^2}} \exp\{-\frac{1}{2\sigma^2}(\ln(x)-\mu)^2\}$	$x \in (-\infty, \infty)$	$\exp\{\mu + \sigma^2/2\}$	$\exp\{2\mu + \sigma^2\}(\exp\{\sigma^2\} - 1)$
$X \sim \text{Exp}(\lambda)$	$\lambda \exp(-\lambda x)$	$x \geq 0$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$
$X \sim \text{Erlang}(\lambda, r)$	$\lambda^r x^{r-1} \exp(-\lambda x) / (r-1)!$	$x \geq 0 ; r = 1, 2, \dots$	$\frac{r}{\lambda}$	$\frac{r}{\lambda^2}$
$X \sim \text{Ga}(\alpha, \beta)$	$\frac{1}{\Gamma(\alpha)\beta^\alpha} x^{\alpha-1} \exp\{-x/\beta\}$	$x \geq 0$	$\alpha\beta$	$\alpha\beta^2$
$X \sim \text{IGa}(\alpha, \beta)$	$\frac{1}{\Gamma(\alpha)\beta^\alpha} x^{-\alpha-1} \exp\{-1/(x\beta)\}$	$x \geq 0$	$1/(\beta(\alpha-1))$	
$X \sim \text{Weibull}(\alpha, \beta)$	$\frac{\alpha}{\beta} (x/\beta)^{\alpha-1} \exp\{-(x/\beta)^\alpha\}$	$x \geq 0$	$\beta \Gamma(1 + \frac{1}{\alpha})$	$\beta^2 [\Gamma(1 + \frac{2}{\alpha}) - \Gamma^2(1 + \frac{1}{\alpha})]$
$X \sim \text{Beta}(\alpha, \beta)$	$\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1}$	$0 \leq x \leq 1$	$\frac{\alpha}{\alpha+\beta}$	$\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$

$$Z = (X - \mu)/\sigma$$

$$\Gamma(\alpha) = \int_0^\infty x^{\alpha-1} \exp\{-x\} dx$$

$$\Gamma(\alpha) = (\alpha - 1)!$$

$$\Gamma(\alpha + 1) = \alpha \Gamma(\alpha)$$

Solução:

1. `> y <- c(12.1,8.7,11.3,9.2,10.5,9.7,11.6)`

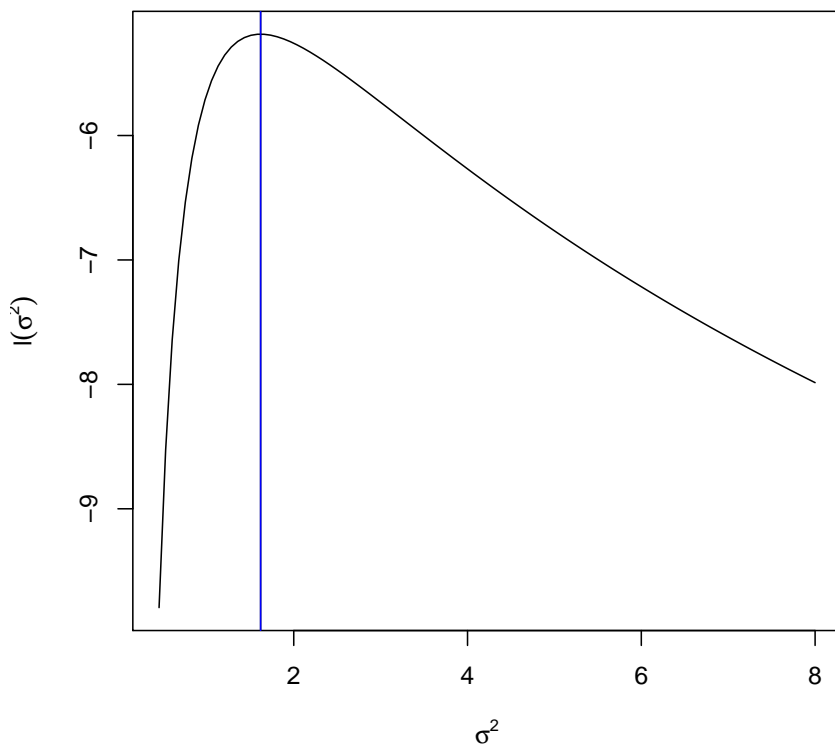
(a) Verossimilhança é dada por:

$$\begin{aligned} L(\sigma^2; y) &= \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(y_i - \mu)^2\right\} \\ &= (2\pi)^{-n/2} (\sigma^2)^{-n/2} \exp\left\{-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu)^2\right\} \end{aligned}$$

```
> (MLE <- sum((y-10)^2)/length(y))
[1] 1.619
> vero <- function(par, mu, dados, log=TRUE){
+ res <- -(length(dados)/2)*log(par) - sum((dados-mu)^2)/(2*par)
+ if(!log) res <- exp(res)
+ res
+ }
> (MLEnum <- optimize(vero, interval=c(0, 10), mu=10, dados=y, log=TRUE, maximum=TRUE))
$maximum
[1] 1.619

$objective
[1] -5.185

> curve(vero(x, mu=10, dados=y, log=TRUE), from=0.45, to=8,
+       xlab=expression(sigma^2), ylab=expression(l(sigma^2)))
> abline(v=c(MLE,MLEnum$maximum), col=c(1,4))
```



(b) A expressão da distribuição a posteriori é obtida da forma:

$$\begin{aligned} [\sigma^2|y] &\propto [\sigma^2] \cdot L(\sigma^2; y) \\ &= (\sigma^2)^{-1} \cdot (2\pi)^{-n/2} (\sigma^2)^{-n/2} \exp \left\{ -\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu)^2 \right\} \\ &\propto (\sigma^2)^{-(n/2)-1} \exp \left\{ -\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2} \right\} \end{aligned}$$

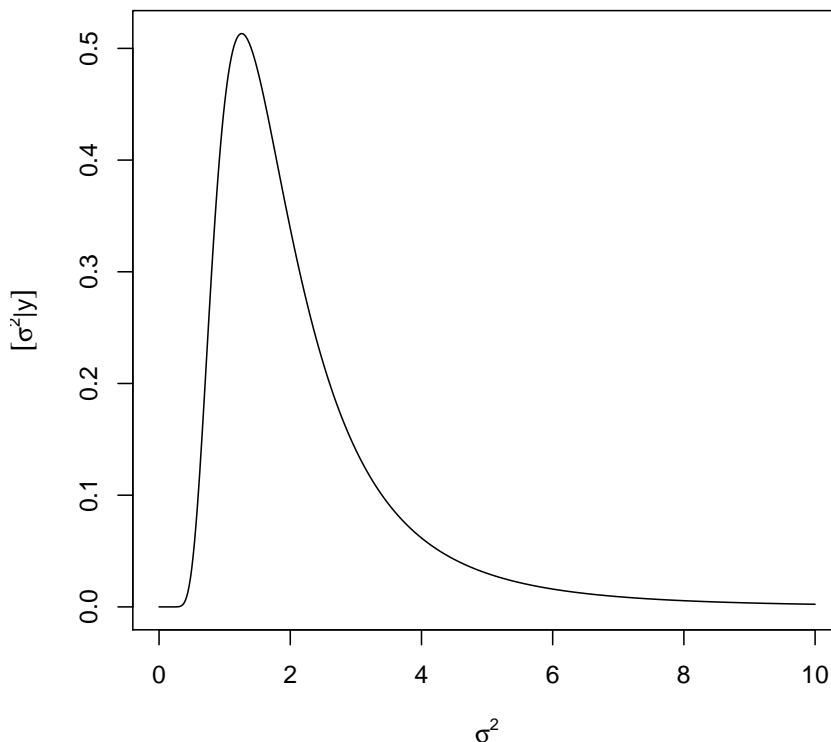
(c) E a expressão anterior, como uma função de σ^2 , corresponde ao núcleo de uma densidade “gama inversa” de parâmetros:

$$\alpha = \frac{n}{2} \quad \text{e} \quad \beta = \frac{2}{\sum_{i=1}^n (y_i - \mu)^2}$$

(d) Para o conjunto de dados temos que

$$\alpha = \frac{n}{2} = \frac{7}{2} = 3,5 \quad \text{e} \quad \beta = \frac{2}{\sum_{i=1}^n (y_i - \mu)^2} = \frac{2}{11,33} = 0,1765.$$

```
> a.post <- length(y)/2
> b.post <- 2/sum((y-10)^2)
> c(a.post, b.post)
[1] 3.5000 0.1765
> dinvgamma <- function(x, a, b, log=FALSE){
+   res <- ifelse(x > 0,
+     - a * log(b) - log(gamma(a)) - (a+1)*log(x) - 1/(b*x), -Inf)
+   if(!log) res <- exp(res)
+   return(res)
+ }
> curve(dinvgamma(x, a=a.post, b=b.post), from=0, to=10, n=501,
+   xlab=expression(sigma^2), ylab=expression(group("[",paste(sigma^2,"|",y),"]")))
```



Se $X \sim \text{Ga}(\alpha, \beta)$ então $Y = 1/X \sim \text{InvGa}(\alpha, \beta)$. Portanto, para simular de uma distribuição gama inversa basta tomar o inverso de valores simulados de uma distribuição gama com os os mesmos parâmetros.

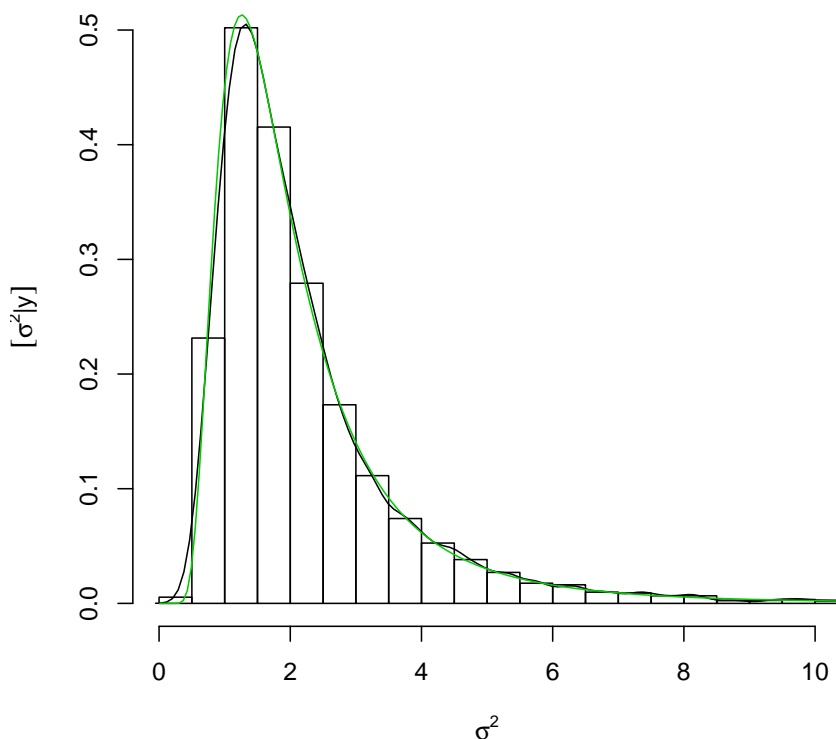
```
> set.seed(123)
> sim <- 1/rgamma(10000, shape=a.post, scale=b.post)
```

```

> hist(sim, prob=TRUE, ylim=c(0,0.5), br=c(seq(0,10, by=0.5), seq(10,60,by=2)),
+      main="amostragem da posteriori", xlab=expression(sigma^2),
+      ylab=expression(group("[",paste(sigma^2,"|",y),"]")), xlim=c(0, 10))
> lines(density(sim))
> curve(dinvgamma(x, a=a.post, b=b.post), from=0.01, to=30, add=T, col=3, n=501)

```

amostragem da posteriori



(e) i. Resumos pontuais

A. Média da posteriori. Neste caso se tem a expressão analítica e que portanto deve ser utilizada. Entretanto, como em muitos casos pode não ser disponível, ilustra-se também a obtenção por integração numérica e por simulação.

- Expressão analítica

$$E[\sigma^2|y] = \frac{1}{\beta(\alpha - 1)} = 2.27$$

```
> (e.post <- 1/(b.post*(a.post-1)))
```

```
[1] 2.266
```

- Integração numérica (a partir da definição de esperança de uma v.a.).

$$E[\sigma^2|y] = \int_0^{\infty} \sigma^2 f(\sigma^2|y) d\sigma^2$$

```
> Epost <- function(par, ...) par * dinvgamma(par, ..., log=FALSE)
```

```
> integrate(Epost, lower=0, upper=50, a=a.post, b=b.post)
```

```
2.263 with absolute error < 6.6e-06
```

- Simulação.

```
> mean(sim)
```

```
[1] 2.28
```

B. Moda da posteriori. Assim como no caso da média a expressão analítica da moda é conhecida mas ilustra-se também a obtenção por otimização numérica e por simulação.

- Expressão analítica.

$$E[\sigma^2|y] = \frac{1}{\beta(\alpha + 1)} = 1.26$$

```
> (mo.post <- 1/(b.post*(a.post+1)))
```

```
[1] 1.259
```

- Otimização numérica, maximizando a densidade (ou, equivalentemente, a log-densidade, o que em geral é mais estável numericamente).

```
> optimize(dinvgamma, lower=0, upper=50, a=a.post, b=b.post, log=FALSE, maximum=TRUE)
$maximum
[1] 1.259
```

```
$objective
[1] 0.5133
```

```
> optimize(dinvgamma, lower=0, upper=50, a=a.post, b=b.post, log=TRUE, maximum=TRUE)
$maximum
[1] 1.259
```

```
$objective
[1] -0.6669
```

- Simulação. Utiliza-se aqui um algoritmo simples tomando-se o ponto de máximo de uma suavização da densidade.¹

```
> sim.den <- density(sim, n=1024)
> sim.den$x[which.max(sim.den$y)]
[1] 1.327
```

C. Mediana da Posteriori

- Expressão analítica: não disponível.
- Otimização numérica.

Pode-se definir uma função que retorne quantis da posteriori gamma inversa.

```
> qinvgamma <- Vectorize(function(p, a, b, ...){
+   q.f <- function(x, a, b)
+     integrate(dinvgamma, lower=0, upper=x, a=a, b=b)$value - p
+   uniroot(q.f, a=a, b=b, ...)$root
+ })
> (md.post <- qinvgamma(0.5, a=a.post, b=b.post, interval=c(0,50)))
[1] 1.785
```

Entretanto, para o cálculo de quantis neste exemplo, a função anterior desnecessária pois pode-se simplesmente tomar inversos dos quantis da distribuição gama.

```
> 1/qgamma(1-0.5, shape=a.post, scale=b.post)
[1] 1.785
```

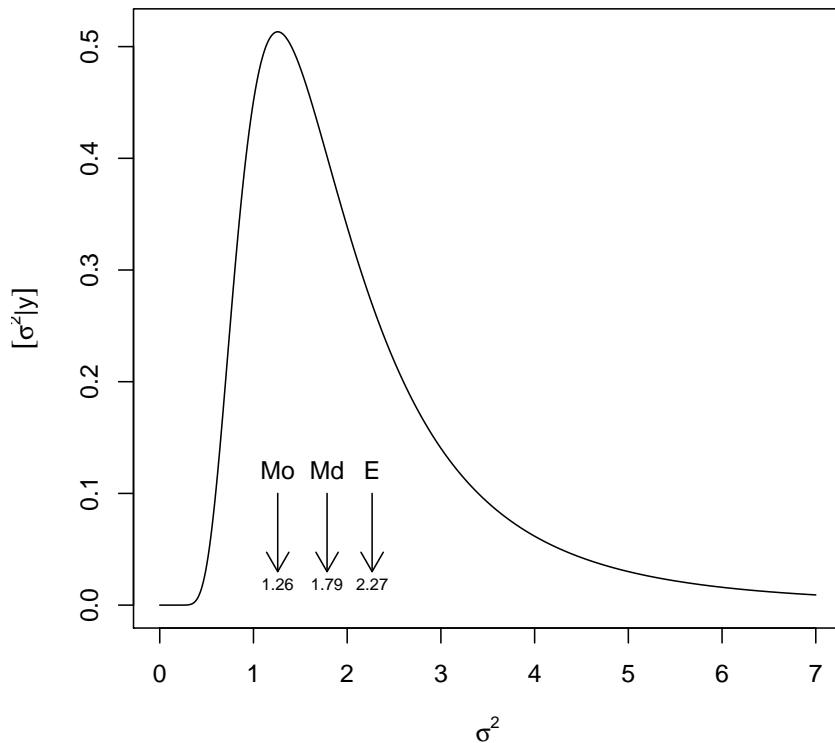
Se desejar ter uma função específica basta fazer:

```
> qinvgamma <- function(p, a, b, ...)
+   1/qgamma(1-p, shape = a, scale=b, ...)
```

- Simulação

```
> median(sim)
[1] 1.794
```

¹veja outro algoritmo mais adiante com o uso da função `hdrcd::hdr()`.



ii. Resumos intervalares (usando $1 - \alpha = 0,95$)

i. Intervalo de credibilidade – quantis

A. Por otimização numérica

```
> (ICq <- qinvgamma(c(0.025, 0.975), a=a.post, b=b.post))
[1] 0.7076 6.7047
```

B. Por simulação

```
> quantile(sim, prob=c(0.025, 0.975))
 2.5% 97.5%
0.7075 6.8346
```

ii. Intervalo de credibilidade – HPD²

A. Otimização numérica. A função a seguir é apenas um exemplo ilustrativo e pode ser escrita de outras formas. Para rotinas mais eficientes vejam as fornecidas por pacotes especializados.

```
> # Função para obter o HPD de uma função (f.d.p.) *unimodal*
> # fç nao totalmente geral devido a valor superior fixado no argumento "interval"
> hpdivgamma <- function(prob, a, b){
+   require(rootSolve)
+   moda <- 1/(b*(a+1))
+   dmax <- dinvgamma(moda, a=a, b=b)
+   #
+   f.int <- function(corte, a, b, prob){
+     f.corte <- function(x) dinvgamma(x, a=a, b=b) - (dmax - corte)
+     int <- uniroot.all(f.corte, interval=c(0,100))
+     p1 <- integrate(dinvgamma, lower=0, upper=int[1], a=a, b=b)$value
+     p2 <- integrate(dinvgamma, lower=0, upper=int[2], a=a, b=b)$value
+     return(((p2-p1) - prob)^2)
+   }
+   corte.int <- optimize(f.int, interval=c(0, dmax), a=a, b=b, prob=prob)
+   f.corte <- function(x)
+     dinvgamma(x, a=a, b=b, log=FALSE) - (dmax - corte.int$minimum)
+   int <- uniroot.all(f.corte, interval=c(0,100))
```

²diversos pacotes do R fornecem algoritmos para cálculos de intervalos HPD. Em geral são fornecidas funções para obtenção de intervalos HPD a partir de um objeto como amostras (simulação) da posteriori, mas alguns pacotes fornecem também funções para cálculo a partir de posterioris na forma de função.

```

+   return(int)
+ }
> (IChpd <- hpdinvgamma(0.95, a=a.post, b=b.post))
[1] 0.4761 5.2770
> integrate(dinvgamma, lower=IChpd[1], upper=IChpd[2], a=a.post, b=b.post)
0.95 with absolute error < 8e-08

```

B. Simulações

```

> require(hdrcde)
> ## baseado nas amostras
> hdr(sim, prob=95)$hdr
      [,1] [,2]
95% 0.3314 5.318
> ## baseado em uma aproximação discreta da densidade
> par.vals <- seq(0, 30, length=2000)
> d.vals <- dinvgamma(par.vals, a=a.post, b=b.post)
> hdr(den=list(x=par.vals, y=d.vals), prob=95)$hdr
      [,1] [,2]
95% 0.4828 5.119

```

Alguns exemplos de funções disponíveis em outros pacotes.

```

> ## outras funções/pacotes
> require(TeachingDemos)
> emp.hpd(sim)
> hpd(qinvgamma, a=a.post, b=b.post)
> detach(package:TeachingDemos)
> ##
> require(BEST)
> hdi(sim, credMass=0.95)
> hdi(qinvgamma, credMass=0.95, a=a.post, b=b.post)
> detach(package:BEST)
> ##
> require(LaplacesDemon)
> p.interval(sim)
> detach(package:LaplacesDemon)
> ##
> require(emdbook)
> tcredint("invgamma", list(a=a.post, b=b.post)) # problema aqui
> detach(package:emdbook)
> #
> require(coda)
> HPDinterval(as.mcmc(sim))
> detach(package:coda)

```

Pode-se verificar que o IC-HPD é mais curto.

```

> diff(ICq)
[1] 5.997
> diff(IChpd)
[1] 4.801

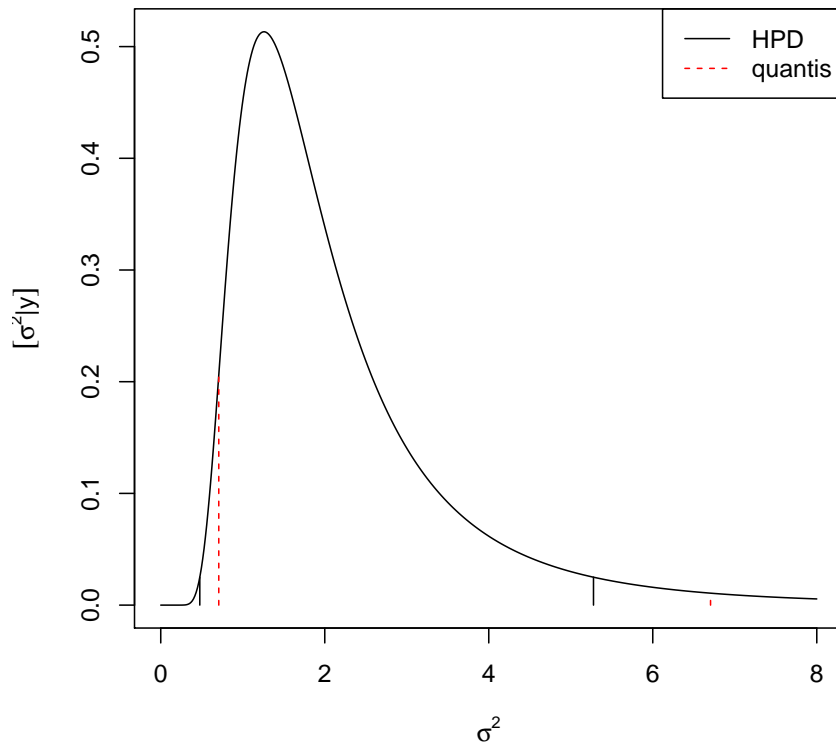
```

O gráfico da função com os dois tipos de intervalos mostra que eles são bem distintos neste caso.

```

> curve(dinvgamma(x, a=a.post, b=b.post), from=0, to=8, n=501,
+       xlab=expression(sigma^2), ylab=expression(group("[",paste(sigma^2,"|",y),"]")))
> segments(ICq, 0, ICq, dinvgamma(ICq, a=a.post, b=b.post), col=2, lty=2)
> segments(IChpd, 0, IChpd, dinvgamma(IChpd, a=a.post, b=b.post))
> legend("topright", c("HPD", "quantis"), col=c(1,2), lty=c(1,2))

```

Gráficos da priori, posteriori e verossimilhança.

Para incluir o gráfico da verossimilhança na mesma escala da priori e posteriori é necessário utilizar a função padronizada (de forma a integrar 1). Há duas formas que serão ilustradas a seguir: (i) reconhecendo o núcleo de alguma distribuição conhecida (quando possível), (ii) integrando explicitamente a verossimilhança (analítica ou numericamente).

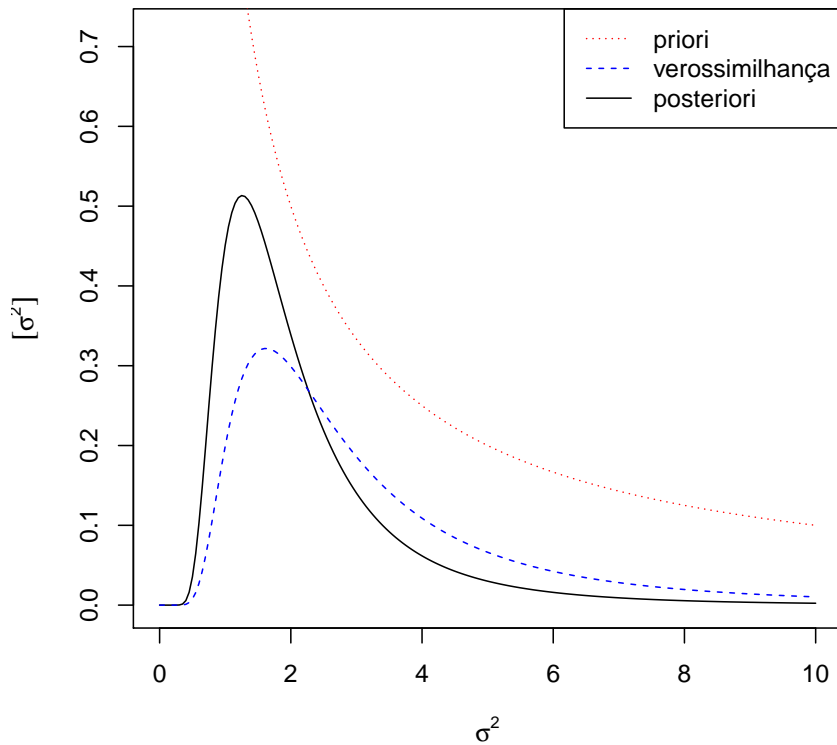
Neste exemplo a opção (i) é possível pois, para a verossimilhança, tem-se que:

$$L(\sigma^2|y) \propto IG(a = (n/2) - 1, b = 2/\sum_{i=1}^n (y_i - \mu)^2).$$

```
> post <- function(dados, mu, plot=TRUE, from, to){
+   a <- length(dados)/2
+   SQ <- sum((dados-mu)^2)
+   b <- 2/SQ
+   MLE <- SQ/length(dados)
+   if(plot){
+     par.seq <- seq(from, to, length=201)
+     vero.seq <- dinvgamma(par.seq, a=a-1, b=b)
+     post.seq <- dinvgamma(par.seq, a=a, b=b)
+     max.seq <- max(c(max(vero.seq), max(post.seq)))
+     plot(par.seq, post.seq, type="l", ylim=c(0, 1.4*max.seq),
+          xlab=expression(sigma^2), ylab=expression(group("[",sigma^2,"]")))
+     lines(par.seq, vero.seq, lty=2, col=4)
+     lines(par.seq, 1/par.seq, lty=3, col=2)
+     legend("topright", c("priori","verossimilhança","posteriori"),
+           lty=c(3,2,1), col=c(2,4,1))
+   }
+   return(c(a=a, b=b, MLE=MLE, media=1/(b*(a-1)), moda=1/(b*(a+1))))
+ }
```

```
> post(dados=y, mu=10, from=0, to=10)
```

a	b	MLE	media	moda
3.5000	0.1765	1.6186	2.2660	1.2589



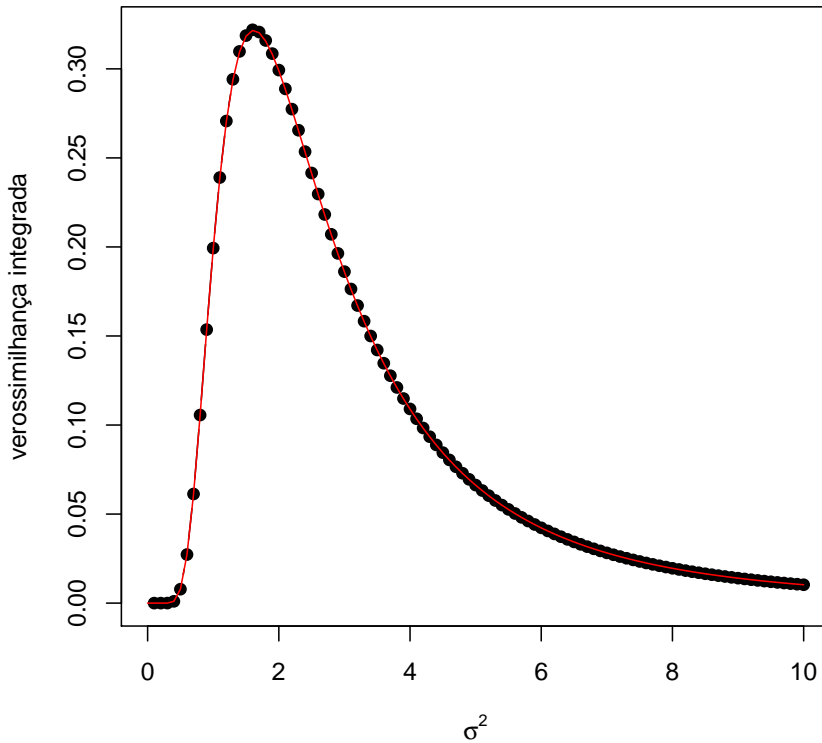
Para (ii) a verossimilhança integrada (numericamente) pode ser obtida da forma a seguir, que pode ser usada mesmo que a expressão da verossimilhança não tenha uma forma proporcional à de uma distribuição de probabilidades conhecida.

```
> vero.int <- function(par, ...){
+   C <- integrate(vero, low=0, up=50, ..., log=FALSE)$value
+   res <- vero(par, ..., log=FALSE)/C
+   attr(res, "C") <- C
+   return(res)
+ }
> ## verificando...se integra 1
> integrate(vero.int, mu=10, dados=y, lower=0, upper=50)

1 with absolute error < 2.9e-06
```

E o gráfico a seguir mostra que (i) e (ii) são equivalentes.

```
> curve(vero.int(x, mu=10, dados=y), from=0, to=10, type="b", pch=19,
+       xlab=expression(sigma^2), ylab="verossimilhança integrada")
> curve(dinvgamma(x, a=a.post-1, b=b.post), from=0, to=10, col=2, add=T)
```



Aproximação normal da posteriori Sob certas condições a posteriori pode ser aproximada por uma distribuição normal com média igual a moda da posteriori e variância igual ao negativo do inverso do Hessiano. A aproximação é boa quando (i) o parâmetro é interior ao espaço paramétrico, para valores do parâmetro na proximidade da moda e para grandes tamanhos de amostra.

$$\begin{aligned}
 [\sigma^2|y] &\propto (\sigma^2)^{-(n/2)-1} \exp\left\{-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2}\right\} \\
 \log[\sigma^2|y] &\propto -(n/2) - 1 \log(\sigma^2) - \frac{\sum_{i=1}^n (y_i - \mu)^2}{2} (\sigma^2)^{-1} \\
 \frac{d \log[\sigma^2|y]}{d\sigma^2} &\propto -(n/2) - 1 (\sigma^2)^{-1} + \frac{\sum_{i=1}^n (y_i - \mu)^2}{2} (\sigma^2)^{-2} \\
 H(\sigma^2) = \frac{d^2 \log[\sigma^2|y]}{d(\sigma^2)^2} &\propto -(-(n/2) - 1) (\sigma^2)^{-2} - \sum_{i=1}^n (y_i - \mu)^2 (\sigma^2)^{-3}
 \end{aligned}$$

E a variância para a aproximação normal é dada por $H(\hat{\sigma}^2) = -2.84$, em que $\hat{\sigma}^2 = 1/(\beta(\alpha + 1)) = 1.26$ é a moda da posteriori. Portanto a variância a aproximação normal é da posteriori:

$$[\sigma^2|y] \sim N(\hat{\sigma}^2, -H^{-1}(\hat{\sigma}^2)) \sim N(1.26, 0.352)$$

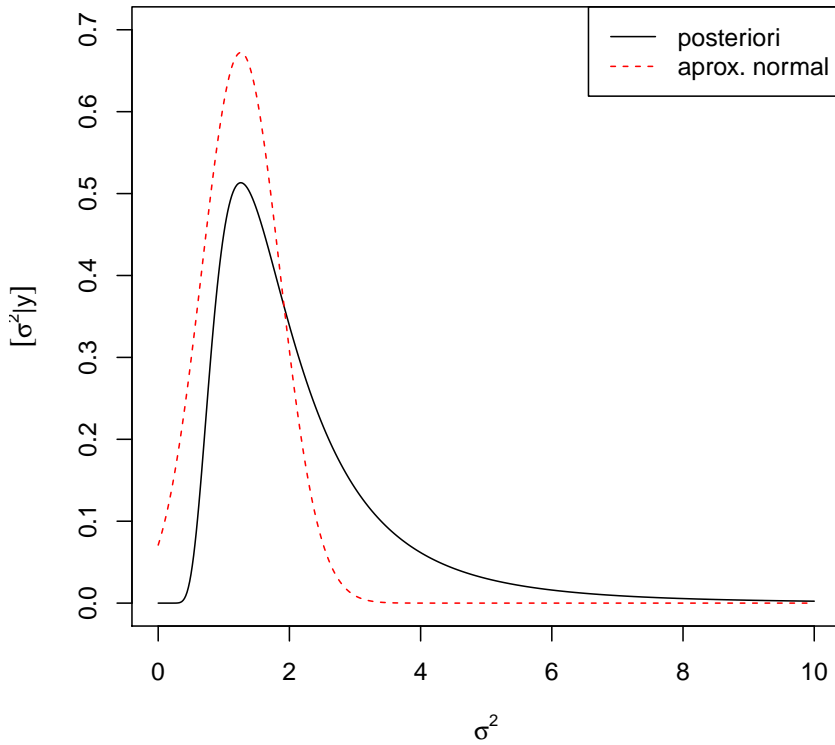
Neste caso ambos, $\hat{\sigma}^2$ e $-H^{-1}(\hat{\sigma}^2)$ puderam ser obtidos analiticamente. Caso isto não fosse possível aproximação obtidas numericamente podem ser obtidas pela maximização da log-posteriori (ou minimização do negativo da log-posteriori).

```
> neglogpost <- function(par, y, mu)
+   (0.5*length(y)+1) * log(par) + 0.5 * sum((y-mu)^2)/par
> optim(1, neglogpost, y=y, mu=10, method="L-BFGS-B", lower=0, hessian=TRUE)[c("par", "hessian")]
```

```
$par
[1] 1.259
```

```
$hessian
[,1]
[1,] 2.839
```

```
> curve(dinvgamma(x, a=a.post, b=b.post), from=0, to=10, n=501, ylim=c(0, 0.7),
+       xlab=expression(sigma^2), ylab=expression(group("[",paste(sigma^2,"|",y),"]")))
> curve(dnorm(x, m=mo.post, sd=sqrt(-1/H)), from=0, to=10, n=501, col=2, add=T, lty=2)
> legend("topright", c("posteriori","aprox. normal"), lty=c(1,2), col=c(1,2))
```



Amostragem (MCMC) Como já visto anteriormente a posteriori do exemplo possui forma analítica conhecida e amostras da posteriori podem ser obtidas diretamente tomando os inversos dos valores simulados da distribuição gamma correspondente. Entretanto, a título de ilustração será exemplificado a seguir um algoritmo de *Cadeias de Markov via Monte Carlo* (MCMC) para obter amostras da posteriori.

O algoritmo de Metropolis-Hastings para amostrar de uma distribuição $\pi(\theta)$ segue os seguintes passos:

1. Toma-se um valor inicial θ_0 (arbitrário) para o parâmetro.
2. Escolhe-se uma função $q(\theta_*|\theta_{t-1})$ para sortear um novo valor θ_* “proposto” para o parâmetro.
3. Geram-se valores em uma “cadeia” (*suficientemente grande*) segundo as regras:

(a) para cada valor proposto calcula-se $\alpha = \min \left\{ 1, \frac{\pi(\theta_*)q(\theta_{t-1}|\theta_*)}{\pi(\theta_{t-1})q(\theta_*|\theta_{t-1})} \right\}$,

(b) aceita-se que a cadeia mova-se para o valor proposto com probabilidade α , ou seja, sorteia-se $U \sim U(0, 1)$ e

$$\theta_t = \begin{cases} \theta_* & \text{se } u < \alpha \\ \theta_{t-1} & \text{se } u \geq \alpha \end{cases} .$$

Uma versão simplificada deste algoritmo (chamada de algoritmo de Metrópolis) é obtida quando $q(\cdot)$ é simétrica. Neste caso $q(\theta_*|\theta_{t-1}) = q(\theta_{t-1}|\theta_*)$ e o cálculo de α se reduz a

$$\alpha = \min \left\{ 1, \frac{\pi(\theta_*)}{\pi(\theta_{t-1})} \right\} .$$

A distribuição $\pi(\theta)$ e a distribuição objetivo, da qual se deseja simular, o caso a posteriori $[\sigma^2|y]$. É importante notar que a função não precisa ser completamente especificada, ou seja pode ser conhecida somente a uma constante de proporcionalidade uma vez que tal constante é cancelada no cálculo de α . Portanto para simular de uma posteriori precisamos somente definir em $\pi(\cdot)$ a função obtida por

$$[\sigma^2|y] \propto [\sigma^2] \cdot L(\sigma^2; y).$$

Na definição da função usaremos agora os argumentos \mathbf{n} e \mathbf{SQ} para evitar recalculando quantidades fixas a cada iteração.³ Usa-se ainda o fato que $\frac{\pi(\theta_*)}{\pi(\theta_{t-1})} = \exp[\log(\pi(\theta_*) - \pi(\theta_{t-1}))]$ ⁴.

³Funções definidas anteriormente como a de verossimilhança também poderiam ser definidas desta forma para maior eficiência.

⁴Outras simplificações poderiam ser obtidas para este exemplo considerando a razão para o cálculo de α .

```

> dpost <- function(par, n, SQ)
+   ifelse(par > 0, exp(-(0.5*n+1) * log(par) - 0.5 * SQ/par), 0)
> logdpost <- function(par, n, SQ)
+   ifelse(par > 0, -(0.5*n+1) * log(par) - 0.5 * SQ/par, -Inf)

```

O algoritmo a seguir utiliza uma $q(\cdot)$ simétrica uniforme ao redor do valor atual do parâmetro tal que $\theta_* \sim U[\theta_{t-1} - a, \theta_{t-1} + a]$.

```

> MCMC <- function(N, init, n, SQ, a){
+   sim <- numeric(N)
+   sim[1] <- init
+   aceita <- 0
+   for(i in 2:N){
+     nv <- runif(1, sim[i-1]-a, sim[i-1]+a)
+     alpha <- min(1, exp(logdpost(nv, n=n, SQ=SQ)-logdpost(sim[i-1], n=n, SQ=SQ)))
+     if(runif(1) < alpha) {sim[i] <- nv; aceita <- aceita+1}
+     else sim[i] <- sim[i-1]
+   }
+   attr(sim, "taxa aceitação") <- aceita/(N-1)
+   return(sim)
+ }

```

E rodando o algoritmo para o exemplo temos os seguintes resultados.

```

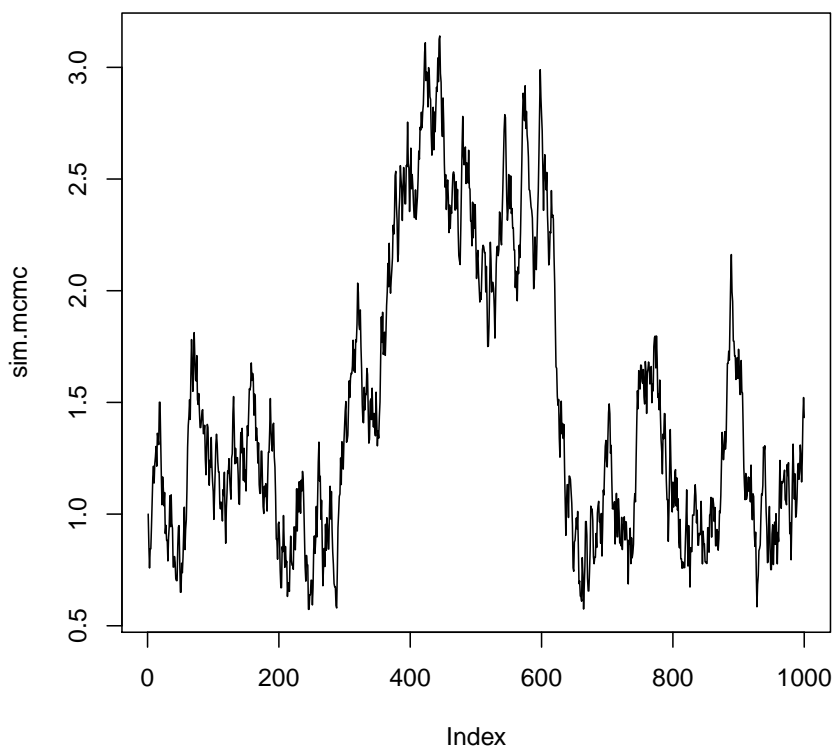
> n.y <- length(y)
> SQ.y <- sum((y-10)^2)
> sim.mcmc <- MCMC(1000, init=1, n=n.y, SQ=SQ.y, a=0.2)
> plot(sim.mcmc, ty="l")
> attributes(sim.mcmc)

```

```

$`taxa aceitação`
[1] 0.9439

```



A figura que sobrepõe a densidade empírica mostra que esta amostra é claramente inadequada!

```

> curve(dinvgamma(x, a=a.post, b=b.post), from=0, to=10, n=501, ylim=c(0, 0.7),
+   xlab=expression(sigma^2), ylab=expression(group("[",paste(sigma^2,"|",y),"]")))
> hist(sim.mcmc, prob=T, add=T)
> lines(density(sim.mcmc), col=2)

```

O algoritmo MCMC requer cuidados. Entre eles destacamos: (i) é necessário que a distribuição estacionária seja atingida o que em geral não é válido para valores iniciais da cadeia que devem ser descartados sendo considerados um período de “aquecimento” da cadeia (*burn-in*); (ii) a distribuição $q(\cdot|\cdot)$ deve ser calibrada (no exemplo pela escolha do valor argumento \mathbf{a}) para fornecer uma taxa de aceitação compatível com o algoritmo (no exemplo seria por volta de 30%) (iii) as amostras podem ser muito correlacionadas o que requer um maior número de valores a serem gerados (N). Além disto, pode-se armazenar apenas um a cada tantos valores gerados (por exemplo 1 a cada 10) o que reduz a memória computacional utilizada para armazenamento e reduz a autocorrelação dos valores a serem armazenados e a este procedimento denomina-se raleamento (*thinning*). Com estes pontos em mente o algoritmo acima deve ser adaptado e calibrado (por tentativa e erro na especificação de \mathbf{a} para atingir a taxa aceitação recomendada).

Existem ainda na literatura diversos “testes convergência” que visam verificar se há evidências de que a cadeia atingiu seu estado estacionário.
