Funcion Q

J. Abellán

27 de enero de 2016

La definición de Q

La definición de la función Q es muy intuitiva: nos da la *distancia* de una curva teórica a unos datos:

$$Q = Q(\theta) = \Sigma_1^n (y_i - y(x_i; \theta))^2$$

donde y_i son los datos medidos, $y(x_i; \theta)$ el valor que predice la teoría para ese x_i concreto y θ el parámetro o parámetros de la teoría que queremos inferir a partir de los datos $D = \{x_i, y_i, i = 1, 2, \cdots, N\}$ medidos.

El cálculo de Q

El cálculo de la función Q con **R** es muy sencillo:

Un parámetro Q = Q(g)

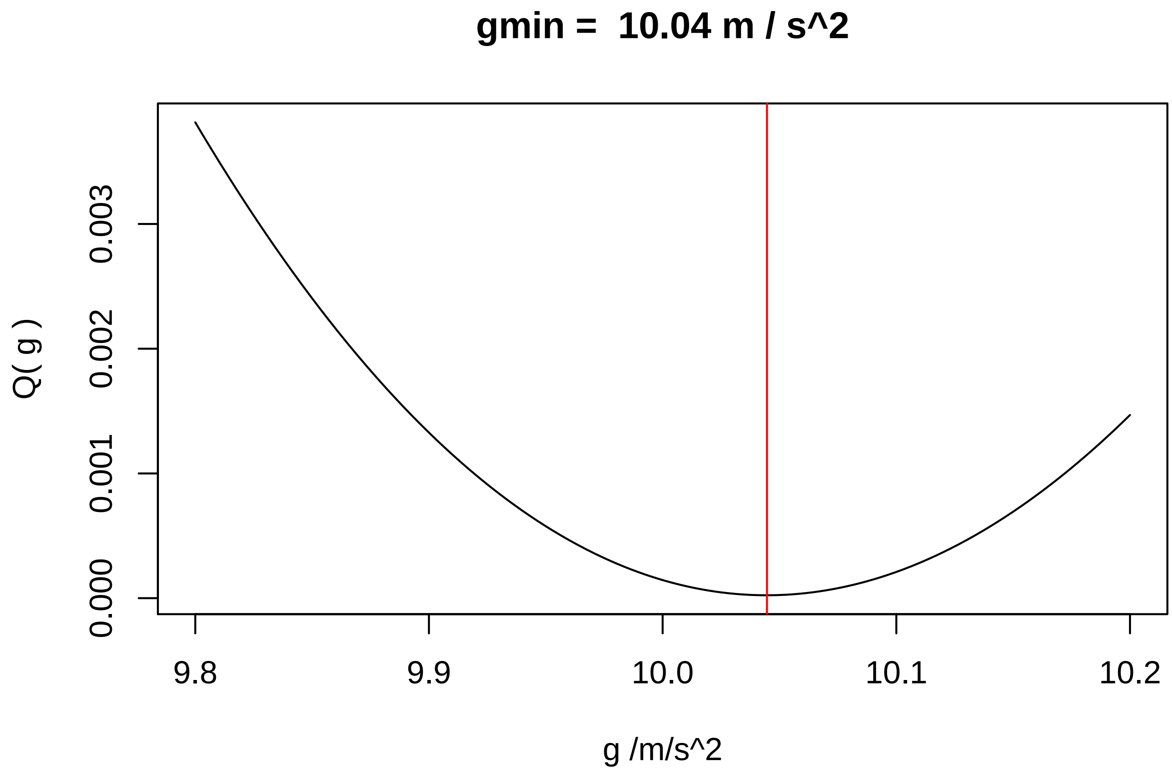
Como ejemplo utilizaremos el péndulo simple. La teoría es bien conocida:

$$T = T(g) = 2\pi\sqrt{\frac{l}{g}}$$

donde T es el periodo (variable respuesta), l la longitud del péndulo (variable de control) y g, la gravedad, el parámetro que queremos inferir.

```
#Función teórica.
#Péndulo simple: recta que pasa por el origen
y.teo <- function( x, g ) { 2 * pi * sqrt( x / g ) }

# Los datos
# variable de control: longitud del péndulo (cm)
lexpcm <- c( 178.1, 162.2, 152.2, 135.2 )

# tiempos de Nperiodos medidos (en segundos)
tNperiodos <- c( 132.09, 126.34, 122.31, 115.36 )

# número de datos
Ndatos <- length( lexpcm )

# longitud del péndulo (en metros)
X.exp <- lexpcm / 100

#Número de oscilaciones medidas
Nperiodos <- 50
```

```r
#Variable respuesta: periodo (en segundos)
Y.exp <- tNperiodos / Nperiodos

#Función Q, calidad del ajuste

#Rango del parámetro g
#divisones del rango
n <- 1000

gmin <- 9.8; gmax <- 10.2

#Rango del parámetro
G <- seq( gmin, gmax, len = n )

dG <- ( gmax - gmin ) / ( n - 1 )

# Declaración de Q
Q <- rep( 0, n )

# Cálculo de Q
for (i in seq(along = G) ){

    Q[ i ] <- sum( ( Y.exp - y.teo( X.exp, G[ i ] ) )^2 )

}

# Valor mínimo de Q
Qmin <- min( Q )

# Valor de g para el que se obtiene el mínimo de Q
Gmin <- G[ which.min( Q ) ]

# Dibujamos la función Q
plot( G, Q,

      type = "l",

      xlab = "g /m/s^2",

      ylab = "Q( g )",

      main = paste("gmin = ", round( Gmin, 2), "m / s^2" )

    )

abline( v = Gmin, col = 2 )
```



El teorema de Bayes

De acuerdo con el teorema de *Bayes* (véase los apuntes 'Lab2016-Bayes.pdf'):

$$p(\theta|DI) = p(\theta|I)\frac{p(D|\theta I)}{p(D|I)}$$

donde $p(\theta|DI)$ es la función de distribución de los parámetros una vez conocidos los datos o *posterior*, θ es el vector de los parámetros (de la teoría más el ruido) $\theta = (\theta_1, \theta_2, ..., \theta_p, \sigma)$, D son los datos, I toda la información relevante sobre el problema, $p(\theta|I)$ la función de distribución de los parámetros antes de adquirir los datos o *prior*, $p(D|\theta_1\theta_2...\theta_p\sigma I)$ es la función de verosimilitud de los datos suponiendo conocidos los parámetros y $p(D|I)$ es una constante de normalización.

Es fácil demostrar que en muchos casos el cálculo de la función de distribución conjunta se simplifica mucho:

$$p(\theta\sigma|DI) \propto \frac{1}{\sigma}p(D|\theta\sigma I)$$

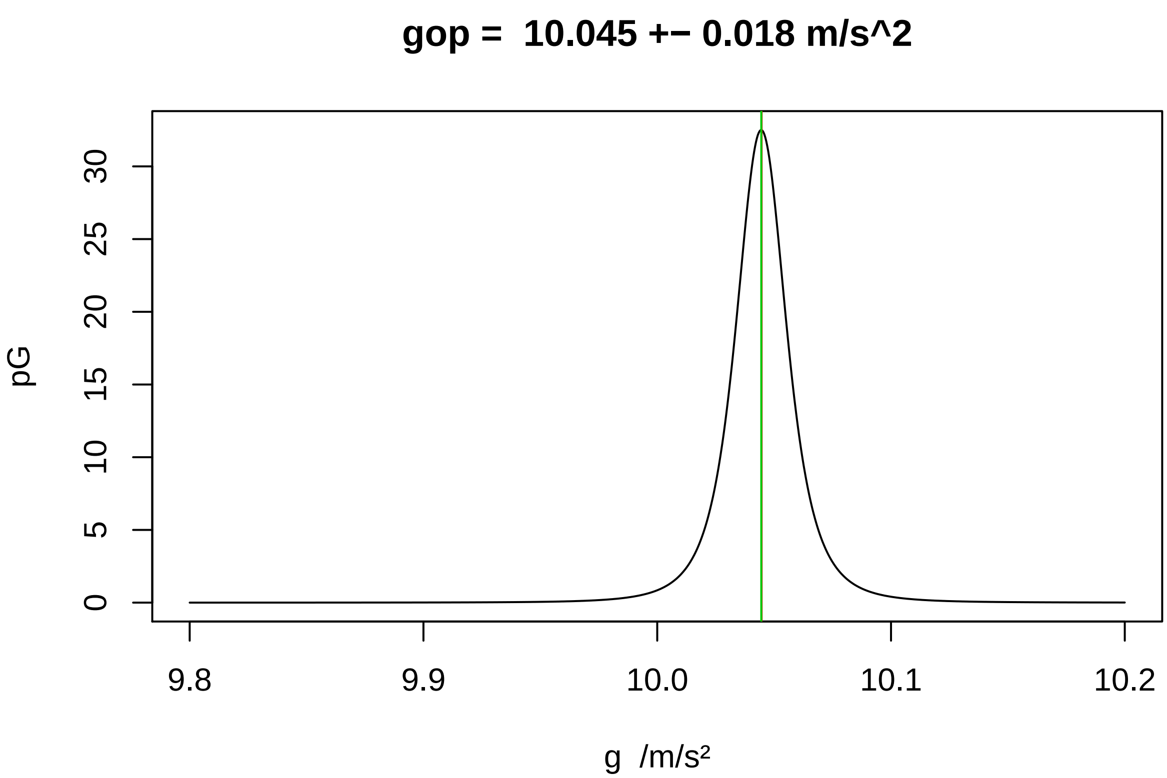
'Eliminando' σ mediante marginalización:

$$p(\theta|DI) \propto Q(\theta)^{-N/2}$$

¡De aquí viene la importancia de saber calcular la función $Q(\theta)$!

```
#Teorema de Bayes
#Función de distribución del parámetro g
pG <- Q^( - Ndatos / 2 )
```

```r
#Normalización
pG <- pG / sum( pG * dG )

#Estadísticos
#valor más probable: moda
g.mp <- G[ which.max( pG ) ]

#media
g.media <- sum( G * pG * dG )

#varianza
varG <- sum( ( G - g.media )^2 * pG * dG )

#desviación estándar (incertidumbre)
deG <- sqrt( varG )

#Dibujamos la función de distribución
plot( G, pG,

      type = "l",

      xlab = "g  /m/s²",

      main = paste("gop = ", round( g.mp, 3 ), "+-", round( deG, 3 ), "m/s^2" )

    )

abline( v = c( Gmin, g.media ), col = c( 2, 3 ) )
```



```r
# El ajuste
# Rango de la variable de control
l1 <- 0.9 * min( X.exp )

l2 <- 1.2 * max( X.exp )

n <- 1000

l.teo <- seq( l1, l2, len = n )

# La curva teórica más probable
T.teo <- y.teo( l.teo, g.mp )

# Superposición de los datos y la curva teórica
plot( X.exp, Y.exp,

      xlab = expression( sqrt( l ) / sqrt( m ) ),

      ylab = "T  /s",

      main = paste("g = ", round( g.mp, 3 ),

                   "+-", round( deG, 3 ), "m/s^2" )

    )

#el mejor ajuste teórico
lines( l.teo, T.teo, col = 2 )
```