PEC 5 - Interpolación, derivación e integración numérica (II)

Alan Coila Bustinza

2022-04-04

1. Derivación

1.1 Valores Exactos

Iniciamos declarando los valores solicitados para la obtención de la call.

```
K=99
r=0.01
sigma=0.4
T=0.25
```

Calcularemos ahora los valores exactos de delta(δ) y gamma(Γ), lo haremos segun las ecuaciones del modelo de Black -Scholes:

$$v(S_0, \sigma, T, r, K) = S_0\Phi(d_1) - e^{-rT}K\Phi(d_2)$$

donde:

$$d_1 = \frac{\ln(S_0/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}, d_2 = d_1 - \sigma\sqrt{T}$$

Los valores de δ y Γ se obtienen al derivar v:

$$\delta = \Phi(d_1), \Gamma = \frac{\phi(d_1)}{\sigma\sqrt{T}S_0}$$

Para el cálculo de estos valores usaremos las funciones de distribución y densidad normal estándar $\Phi(x) = \int_{-\infty}^{x} \phi(y)dy$ y $\phi(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$ que aplicaremos en las siguientes funciones auxiliares .

```
# delta de call (formula exacta)
BSdeltacall=function(S)
{
  d1=(log(S/K)+(r+0.5*sigma^2)*T)/(sigma*sqrt(T))
  # funciones de distribucion y densidad
  deltacall=integrate(function(y) 1/sqrt(2*pi)*exp(-0.5*y^2), lower = -Inf, upper = d1)
  return(deltacall)
}
```

```
#gamma de la call (formula exacta)
BSgammacall=function(S)
{
  d1=(log(S/K)+(r+0.5*sigma^2)*T)/(sigma*sqrt(T))
  gamma=(1/sqrt(2*pi)*exp(-0.5*d1^2))/(sigma*sqrt(T)*S)
  return(gamma)
}
```

Ahora aplicamos las funciones para un S_0 de 99

```
# valor exacto de delta
# ya que esta funcion es producto de una integracion añadimos el subconjunto
# "value" para obtener el valor numérico directamente sin el error
vd=BSdeltacall(99)$value

# valor exacto de gamma
vg=BSgammacall(99)
```

1.2 Derivación numérica

Primero definiremos la función call para poder ejecutar la derivación:

```
#precio de la call
BScall=function(S)
{
  d1=(log(S/K)+(r+0.5*sigma^2)*T)/(sigma*sqrt(T))
  d2=d1-sigma*sqrt(T)
  # definimos la funcion anidada phi para no redundar al definir la call, usaremos el
  # subconjunto "value" para poder realizar cálculos con este valor
  phi= function(x) {
    a = (integrate(function(y) 1/sqrt(2*pi)*exp(-0.5*y^2), lower = -Inf, upper = x))$value
    return(a)
  }
  call=S*phi(d1)-exp(-r*T)*K*phi(d2)
  return(call)
}
```

Ahora crearemos las funciones auxiliares correspondientes a la primera y la segunda derivada:

```
# primera derivada numerica (delta)
# esta funcion recibira como primer argumento la funcion a derivar,
# el valor de x y la variacion de x
findiff <- function (f, x, h) {
  return ((f(x + h) - f(x)) / h)
}
# segunda derivada numerica (gamma)
# forma similar a la primera derivada
findiff2 <- function (f, x, h) {
  return ((f(x + h) - 2 * f(x) + f(x - h)) / h^2)
}
```

Con las funciones definidas podemos calcular los errores: a. error absoluto: mediante la resta con el valor real calculado previamente (vd/vg) b. error relativo: la división entre el error asbsoluto y el valor real expresado en porcentaje

```
#valor aproximado de la delta para h=0.1 y errores
vdapprox=findiff(BScall,99,0.1)
abserrdh1=abs(vd-vdapprox)
relerrdh1=abserrdh1/vdapprox*100

#valor aproximado de la delta para h=0.001 y errores
vdapprox=findiff(BScall,99,0.001)
abserrdh2=abs(vd-vdapprox)
relerrdh2=abserrdh2/vdapprox*100
```

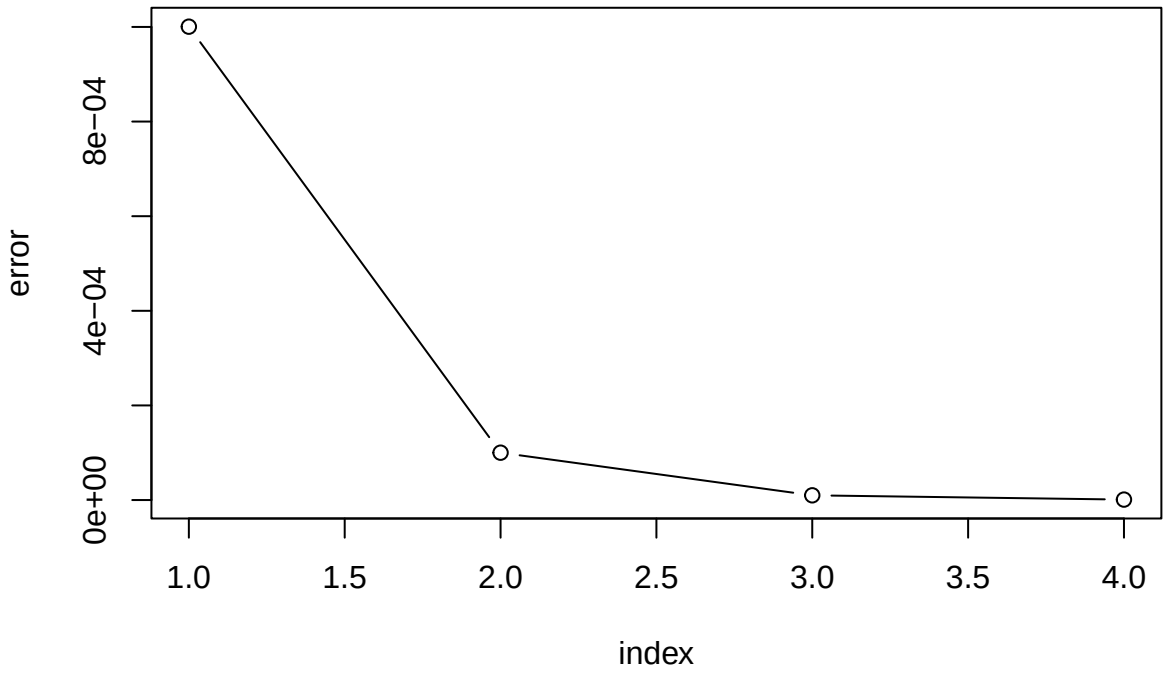
```
#valor aproximado de la gamma para h=0.1 y errores
vgapprox=findiff2(BScall,99,0.1)
abserrgh1=abs(vg-vgapprox)
relerrgh1=abserrgh1/vgapprox*100

#valor aproximado de la gamma para h=0.01 y errores
vgapprox=findiff2(BScall,99,0.01)
abserrgh2=abs(vg-vgapprox)
relerrgh2=abserrgh2/vgapprox*100
```

Tabla de resultados obtenidos:

h	Error absoluto(δ)	Error relativo(δ)	Error absoluto(Γ)	Error relativo(Γ)
0.1	0.0010005	0.1833212	3.6291358×10^{-8}	1.8126226×10^{-4}
0.001	1.0007316×10^{-5}	0.0018369	1.1102553×10^{-9}	5.5453155×10^{-6}

Ahora representaremos el error mediante un ploteo, con el podremos observar que cada para estos cuatro valores de h, cad vez que este se hace mas pequeño, el error tambien disminuye, sin embargo lo hace forma logaritmica por lo que el error tiende a 0 cada vez que h se hace mas pequeña.

```
#representacion error absoluto para delta
vec_h=c(0.1,0.01,0.001,0.0001)
index=c(1,2,3,4)
# realizamos un bucle para utilizar cada valor de h solicitado
# iniciamos el vector que recibira los valores del bucle
vdapprox_vech=c()
for(i in vec_h){
    a= findiff(BScall,99,i)
    vdapprox_vech=c(vdapprox_vech,a)
}
# realizamos el calculo del error
error=abs(vd-vdapprox_vech)
# graficamos
plot(index,error, type="b")
```


error
0e+00
4e−04
8e−04
1.0
1.5
2.0
2.5
3.0
3.5
4.0
index