Trabajo 2

Luis Suárez Lloréns

13 de abril de 2016

Apartado 1: Modelos lineales

Ejercicio 1

Apartado a:

Lo primero, va a ser obtener el gradiente de la función $E(u,v) = (ue^v - 2ve^{-u})^2$.

$$\nabla E(u,v) = (2(ue^v - 2ve^{-u})(e^v + 2ve^{-u}), 2(ue^v - 2ve^{-u})(ue^v - 2e^{-u}))$$

Una vez tenemos el gradiente, el método del descenso del gradiente nos dice que debemos movernos hacia donde nos indique $\nabla E(u,v)$. Por tanto, el siguiente paso es crear funciones tanto para $E(u,v)$ como para su gradiente.

```
E = function(u,v){
  return( (u*exp(v) -  2*v*exp(-u))**2 )
}

grad.E = function(u,v){
  dx = 2*(u*exp(v) -  2*v*exp(-u))*(exp(v)+2*v*exp(-u))
  dy = 2*(u*exp(v) -  2*v*exp(-u))*(u*exp(v)-2*exp(-u))
  return(c(dx,dy))
}
```

Teniendo preparadas ya todas las funciones, creamos el procedimiento de descenso del gradiente.

```
descenso.gradiente = function(x,y,f,grad.f,tam.paso = 0.1, tolerancia = 10**-5, max.iter = 50){
  valores.f = rep(0,max.iter)
  idx = 1

  valores.f[idx] = f(x,y)

  while(f(x,y)>tolerancia && idx < max.iter){
    grad = grad.f(x,y)
    x = x - tam.paso*grad[1]
    y = y - tam.paso*grad[2]

    idx = idx + 1
    valores.f[idx] = f(x,y)
  }

  return(list(x = x, y = y, iter = idx, error = valores.f[1:idx]))
}

descenso.gradiente(1,1,E,grad.E,tolerancia = 10**-14)
```

```
## $x
## [1] 0.04473629
##
## $y
## [1] 0.02395871
##
## $iter
## [1] 11
##
## $error
##  [1] 3.930397e+00 1.159510e+00 1.007407e+00 9.900912e-02 8.660645e-03
##  [6] 1.817558e-04 1.297240e-06 7.291525e-09 4.009998e-11 2.201683e-13
## [11] 1.208683e-15
```

Apartado b

1)

El primer paso, de nuevo, es encontrar el gradiente de la función $f = x^2 + 2y^2 + 2\sin(2\pi x)\sin(2\pi y)$.

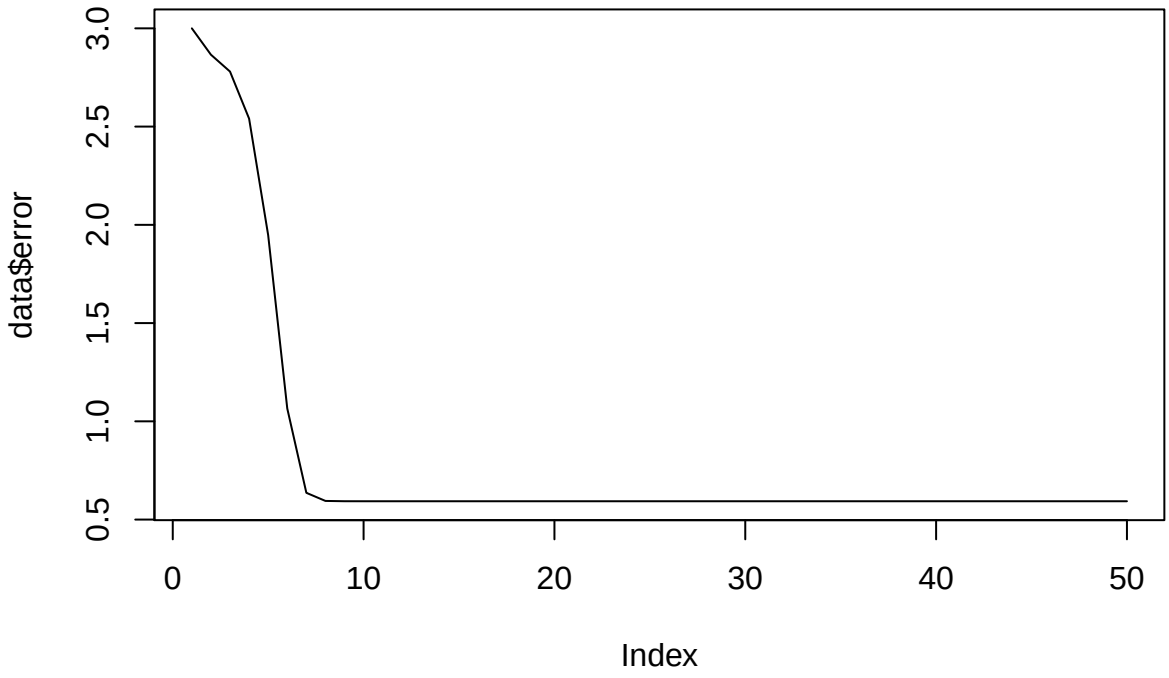
$$\nabla f(x,y) = (2x + 4\pi\sin(2\pi y)\cos(2\pi x), 4y + 4\pi\sin(2\pi x)\cos(2\pi y))$$

Definimos las funciones:

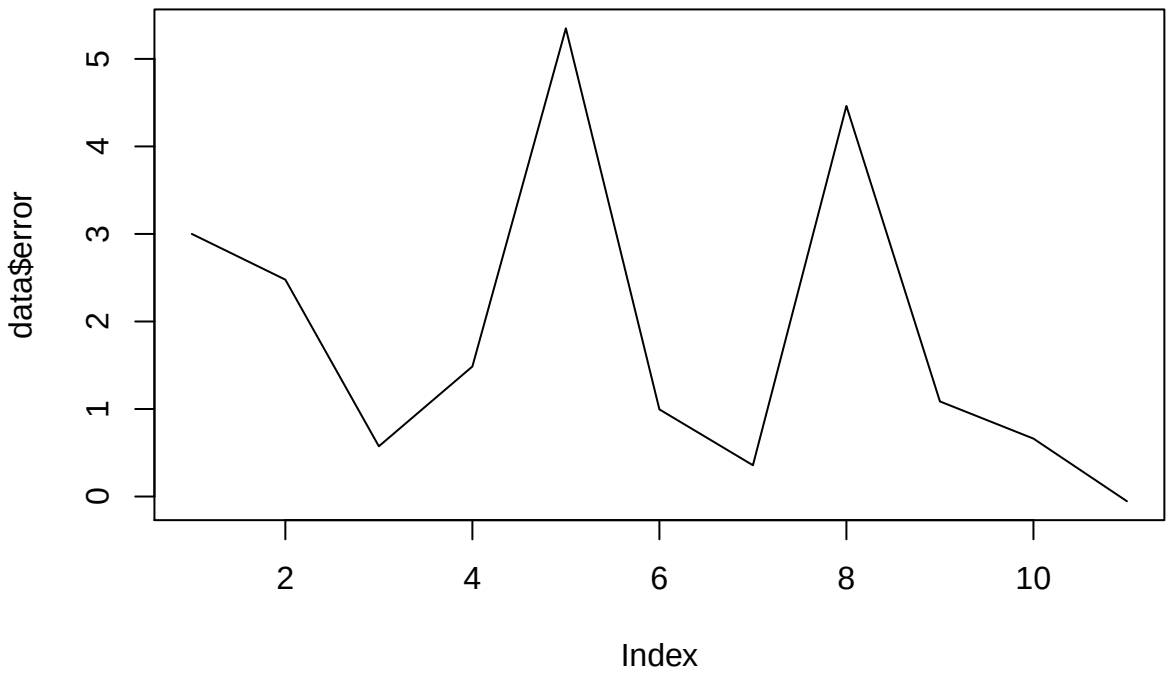
```
f = function(x,y){
  return(x**2+2*y**2+2*sin(2*pi*x)*sin(2*pi*y))
}
grad.f = function(x,y){
  dx = 2*x + 4*pi*sin(2*pi*y)*cos(2*pi*x)
  dy = 4*y + 4*pi*sin(2*pi*x)*cos(2*pi*y)
  return(c(dx,dy))
}
```

Volvemos a calcular el proceso del descenso del gradiente

```
data = descenso.gradiente(1,1,f,grad.f,tam.paso = 0.01, tolerancia = 10**-14)
plot(data$error,type = "l")
```




```
data = descenso.gradiente(1,1,f,grad.f,tam.paso = 0.1, tolerancia = 10**-14)
plot(data$error,type = "l")
```



Del primero, vemos como se queda atascado en un mínimo local, pues el valor se estabiliza en 0.5, y como podemos ver en la segunda gráfica, la función llega a 0. La segunda, pese a acercarse al 0 de la función, se comporta de una manera errática, pues la tasa de aprendizaje es demasiado grande.

2)

Ahora, vamos a ver los datos para diferentes puntos de partida:

```
data = descenso.gradiente(0.1,0.1,f,grad.f,tam.paso = 0.01, tolerancia = 10**-14)
cat(paste0("x:",data$x,"\ny:",data$y,"\nvalor mínimo:",data$error[data$iter]))
```

```
## x:0.00526609481014761
## y:-0.00239206934812207
## valor mínimo:-0.000955213854398881
```

```
data = descenso.gradiente(1,1,f,grad.f,tam.paso = 0.01, tolerancia = 10**-14)
cat(paste0("x:",data$x,"\ny:",data$y,"\nvalor mínimo:",data$error[data$iter]))
```

```
## x:1.21807030131108
## y:0.712811950601778
## valor mínimo:0.593269374325836
```

```
data = descenso.gradiente(-0.5,-0.5,f,grad.f,tam.paso = 0.01, tolerancia = 10**-14)
cat(paste0("x:",data$x,"\ny:",data$y,"\nvalor mínimo:",data$error[data$iter]))
```

```
## x:-0.580323406173378
## y:-0.383991863552131
## valor mínimo:-0.012440020728642
```

```
data = descenso.gradiente(-1,-1,f,grad.f,tam.paso = 0.01, tolerancia = 10**-14)
cat(paste0("x:",data$x,"\ny:",data$y,"\nvalor mínimo:",data$error[data$iter]))
```

```
## x:-1.21807030131108
## y:-0.712811950601778
## valor mínimo:0.593269374325836
```

Ejercicio 2

Vamos a definir la función de **Coordenada Descendente**, que realiza un proceso similar al gradiente descendente, pero avanzando sólo en una dirección cada vez.

```
coordenada.descendente = function(x,y,f,grad.f,tam.paso = 0.1, tolerancia = 10**-5, max.iter = 50){
  valores.f = rep(0,max.iter)
  idx = 1

  valores.f[idx] = f(x,y)

  while(f(x,y)>tolerancia && idx < max.iter){
    grad = grad.f(x,y)
    x = x - tam.paso*grad[1]
    grad = grad.f(x,y)
    y = y - tam.paso*grad[2]

    idx = idx + 1
    valores.f[idx] = f(x,y)
  }

  return(list(x = x, y = y, iter = idx, error = valores.f[1:idx]))
}
```

Ahora, vamos a probarlo con las funciones del ejercicio 1.1.

```
coordenada.descendente(1,1,E,grad.E,tolerancia = 10**-14)
```

```
## $x
## [1] 6.235043
##
## $y
## [1] -3.377273
##
## $iter
## [1] 50
##
## $error
##  [1]  3.93039723 34.29016311  0.53414259  0.43266083  0.36503974
##  [6]  0.31646808  0.27976342  0.25098631  0.22778330  0.20865670
## [11]  0.19260566  0.17893475  0.16714505  0.15686899  0.14782952
```

```
## [16]  0.13981379  0.13265544  0.12622253  0.12040902  0.11512872
## [21]  0.11031077  0.10589645  0.10183660  0.09808978  0.09462081
## [26]  0.09139961  0.08840030  0.08560053  0.08298084  0.08052425
## [31]  0.07821585  0.07604251  0.07399261  0.07205582  0.07022296
## [36]  0.06848580  0.06683698  0.06526986  0.06377846  0.06235738
## [41]  0.06100171  0.05970700  0.05846918  0.05728457  0.05614976
## [46]  0.05506165  0.05401739  0.05301436  0.05205014  0.05112250
```

Como podemos ver, el descenso del gradiente si llega al mínimo mientras que coordenada descendente no. Si nos fijamos en los resultados, este método nos lleva a otro mínimo, lo cual impide encontrar el mínimo que sí habíamos encontrado antes. Esto no hace que no pueda encontrar un mínimo, pero nos hace pensar que el descenso del gradiente es más sólido que la coordenada descendente.

Aquí podemos ver un ejemplo, cambiando la tasa de aprendizaje, de como es capaz de encontrar un mínimo.

```
coordenada.descendente(1,1,E,grad.E,tam.paso = 0.05,tolerancia = 10**-14)
```

```
## $x
## [1] 0.4626781
##
## $y
## [1] 0.9519494
##
## $iter
## [1] 21
##
## $error
##  [1] 3.930397e+00 3.363428e-01 1.041846e-01 1.609311e-02 3.347256e-03
##  [6] 6.112302e-04 1.181141e-04 2.226773e-05 4.243474e-06 8.048796e-07
## [11] 1.529775e-07 2.904943e-08 5.518439e-09 1.048145e-09 1.990941e-10
## [16] 3.781652e-11 7.183082e-12 1.364386e-12 2.591583e-13 4.922575e-14
## [21] 9.350177e-15
```

Ejercicio 3