Aprendizaje Automático (2018-2019)
GRADO EN INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE GRANADA

Práctica 1

UNIVERSIDAD
DE GRANADA

Antonio Jesús Heredia Castillo

16 de marzo de 2019

Índice

Índice de figuras

1. Gradiente descendente

1.1. Implementar el algoritmo de gradiente descendente

He querido implementar el algoritmo del gradiente descendente de la forma mas general que he podido. Para ello, he creado una función que lo calcula. Esta función recibe los siguientes parámetros:

- lr (float): Tasa de aprendizaje
- punto (array[x,y]): Punto desde el cual se quiere aplicar el algoritmo
- función (float): Función a la que se le quiere aplicar el gradiente descendente
- iteraciones (int): Numero máximo de iteraciones que aplicara el algoritmo

Esta subrutina se encarga de hacer las derivadas parciales de la función y de calcular de forma iterativa el punto mínimo de la función. En esta primera versión del algoritmo, la única condición de parada va a ser el numero de iteraciones que le pasa el usuario. Podemos ver el código del algoritmo aquí.

```
_iter = 0 #Contador de iteraciones
_dx = sp.diff(funcion, x) #Derivadas parciales
_dy = sp.diff(funcion, y)
#Lambdifyco las derivadas parciales para calcular mas rapido el valor
_lam_dx = sp.lambdify((x,y),_dx)
_lam_dy = sp.lambdify((x,y),_dy)
while True:
        #creo una copia del punto, para que a la hora de evuluar el x,
        #no lo pierda para y
        _punto_copia = np.copy(_punto)
        #si hay mas iteraicones de las que yo quiero paro
        if _iter > _iteraciones:
                break
        #calculo el gradiente
        _punto[0] = _punto[0]-_lr*_lam_dx(_punto_copia[0],_punto_copia[1])
        _punto[1] = _punto[1]-_lr*_lam_dy(_punto_copia[0],_punto_copia[1])
        _iter =_iter+1
return _punto
```

1.2. Considerar la Función $E(u, v) = (u^2e^v - 2v^2e^{-u})^2$. Usar gradiente descendente para encontrar un mínimo de esta función, comenzando desde el punto $(u, v) = (1, 1)$ y usando una tasa de aprendizaje $\eta = 0,01$.

1.2.1. Calcular analíticamente y mostrar la expresión del gradiente de la función $E(u, v)$

Lo primero que tenemos que saber es que para calcular el gradiente, necesitamos las derivadas parciales de $E(u, v)$.
La derivada respecto de u seria:

$$\frac{\partial E(u,v)}{\partial u} = 2\left(u^2e^v - 2v^2e^{-u}\right)\left(2e^vu + 2v^2e^{-u}\right) \tag{1}$$

La derivada respecto de v seria:

$$\frac{\partial E(u,v)}{\partial v} = 2\left(u^2e^v - 2v^2e^{-u}\right)\left(u^2e^v - 4e^{-u}v\right) \tag{2}$$

Ahora con las derivadas, ya podemos obtener el gradiente. Los nuevos valores de u y v serán:

$$u' = u - 2\eta\left(u^2e^v - 2v^2e^{-u}\right)\left(2e^vu + 2v^2e^{-u}\right) \tag{3}$$

$$v' = v - 2\eta\left(u^2e^v - 2v^2e^{-u}\right)\left(u^2e^v - 4e^{-u}v\right) \tag{4}$$

1.2.2. ¿Cuántas iteraciones tarda el algoritmo en obtener por primera vez un valor de $E(u, v)$ inferior a 10^{-14} . (Usar flotantes de 64 bits)

Para realizar este ejercicio, solo he metido una condición de parada en el algoritmo. La condición de parada en este caso es que el valor de evaluar $E(u, v)$ en un determinada iteración sea $E(u, v) < 10^{-14}$

```
if _iter > _iteraciones or
   _lam_f(_punto_copia[0],_punto_copia[1]) < np.float64(10**-14):
       print ("La funcion ha acabado a las "
       + str(_iter) + " iteraciones")
       break
```

En este caso el numero de iteraciones necesarias ha sido 34, como se puede ver en la Figura1.

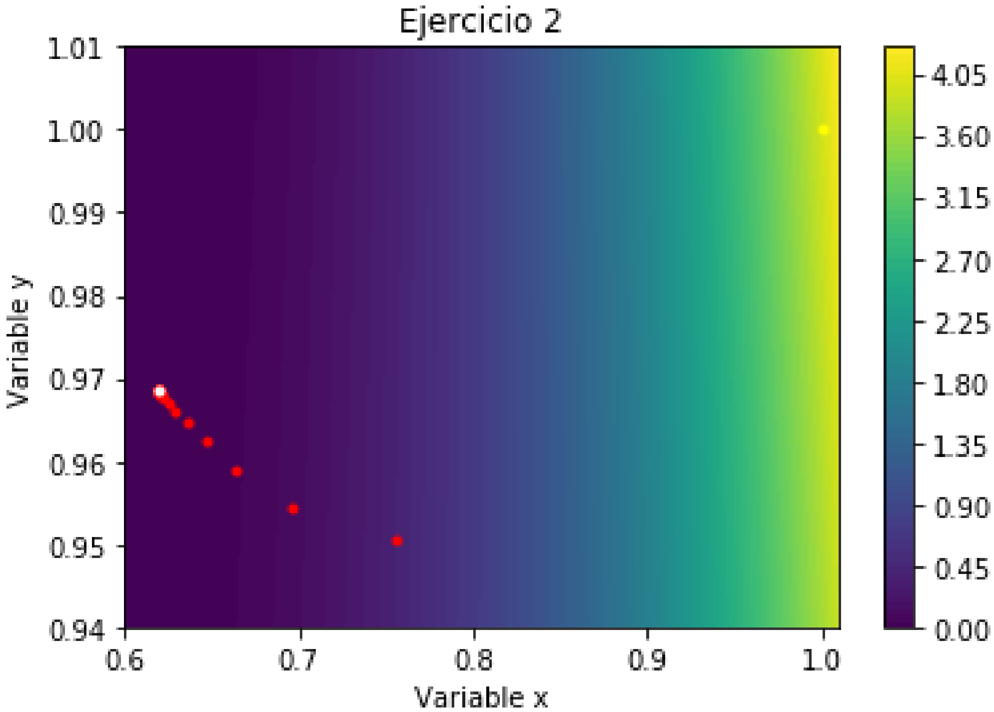

Figura 1: Representación de como avanza el gradiente en la función $E(u, v)$. Punto en amarillo inicio y punto final en blanco.

1.2.3. ¿En qué coordenadas (u, v) se alcanzó por primera vez un valor igual o menor a 10^{-14} en el apartado anterior.

Como el algoritmo del gradiente me devuelve el punto en el que acaba, lo único que tengo que hacer es mostrarlo por pantalla. En este caso el punto en el que se alcanza 10^{-14} es $(0{,}61920768, 0{,}96844827)$

1.3. Considerar ahora la función $f(x,y) = x^2 + 2y^2 + 2\sin(2\pi x)\sin(2\pi y)$

1.3.1. Usar gradiente descendente para minimizar esta función. Usar como punto inicial $(x_0 = 0{,}1, y_0 = 0{,}1)$, (tasa de aprendizaje $\eta = 0,01$ y un máximo de 50 iteraciones. Generar un gráfico de cómo desciende el valor de la función con las iteraciones. Repetir el experimento pero usando $\eta = 0{,}11$, comentar las diferencias y su dependencia de η.

Usando un $\eta = 0{,}01$, podemos gráficamente Figura 2 como el gradiente desciende de forma adecuada. En este caso acaba en una altura de $-1{,}8200785415471563$

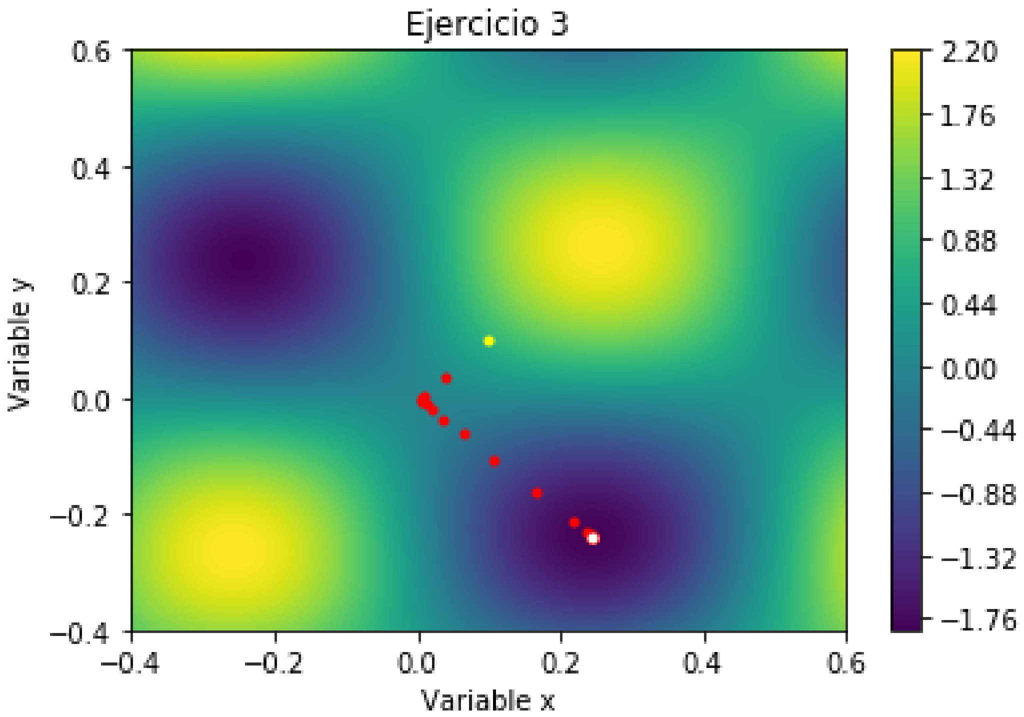

Figura 2: Gradiente de la función $F(u,v)$ con $(x_0 = 0{,}1, y_0 = 0{,}1)$ y $\eta = 0{,}01$. Punto en amarillo inicio y punto final en blanco.

En cambio cuando usamos $\eta = 0{,}1$, podemos ver (Figura 3) como no funciona bien. En este caso acaba en una altura de $2{,}5007003656467446$, que es incluso mas alto que desde donde empezamos $(0{,}7209830056250534)$.

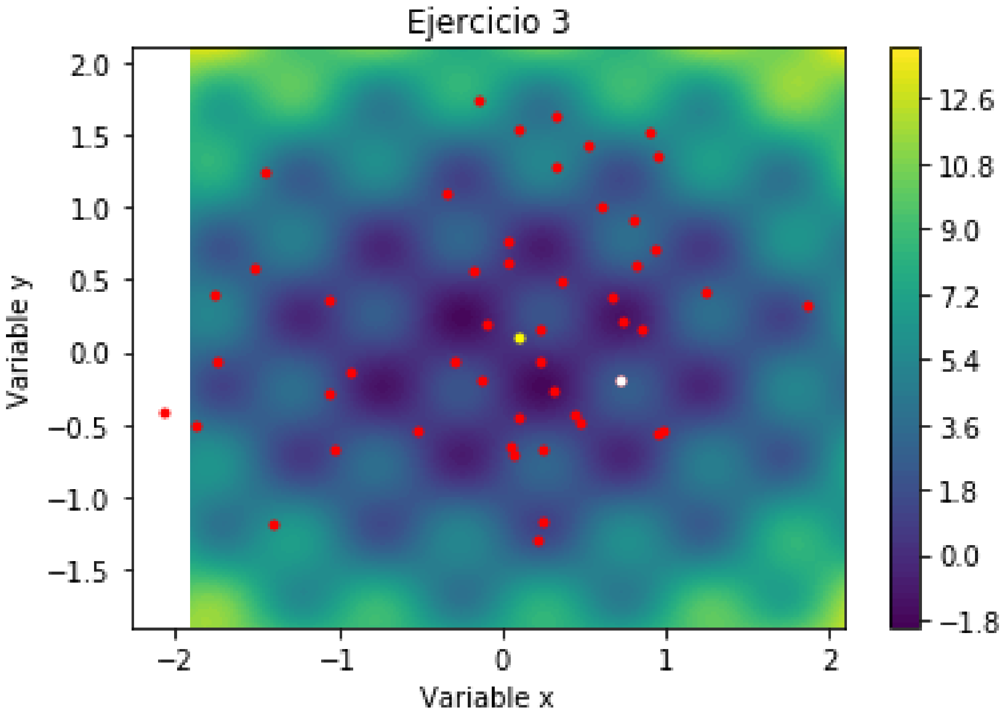

Figura 3: Gradiente de la función $F(u, v)$ con $(x_0 = 0{,}1, y_0 = 0{,}1)$ y $\eta = 0{,}1$. Punto en amarillo inicio y punto final en blanco.