HW 4

- Vamos a programar un optimizador (hacerlo en Python):

1. 1-D:
   a. Dado $f(x)$, queremos crear una función en Python para aproximar el mínimo (local).
   b. Elegimos un valor pequeño de Δx para calcular derivadas, una constante α que va a determinar el tamaño del paso, y otra constante pequeña ε que va a determinar cuándo paramos.
   c. Creamos dos listas vacías: X y F que vamos a ir llenando.
   d. Empezamos en un punto arbitrario $x_i, i = 0$, evaluamos $f(x_i)$. Hacemos $X[0] = x_0$ y $F[0] = f(x_0)$
   e. Calculamos numéricamente la derivada en ese punto: $f'(x_i) = \frac{f(x_i+\Delta x)-f(x_i)}{\Delta x}$
   f. Calculamos $x_{i+1} = x_i - \alpha \cdot f'(x_i)$ (por que el signo menos?) y $f(x_{i+1})$. Hacemos $X[i+1] = x_{i+1}$ y $F[i+1] = f(x_{i+1})$
   g. Verificamos si $|f(x_{i+1}) - f(x_i)| < \varepsilon$. Si se cumple la condición, frenamos, sino volvemos al paso 4 con $i = i + 1$ y repetimos hasta que la condición se verifique (como para que no ocurra que nunca pare si la función no tuviera mínimo, podemos además forzar que si $i = i_{max}$ para algún i_{max}, el programa termina).
   h. Los outputs de la función deben los valores i_{ultimo}, y las listas X y F. Los valores $X[i_{ultimo}] = x_{ultimo}$ y $F[i_{ultimo}] = f(x_{ultimo})$ corresponden a una ε-aproximación al mínimo local buscado. Pero además tenemos toda la historia del camino y los valores de la función que la misma fue tomando. Graficar estos valores en un diagrama (x, f).
   i. Aplicar el optimizador que acaban de programar a las funciones $f(x) = x - x^2 + x^4$ y $g(x) = x - 3x^2 + x^4$. Ver esas funciones en el Geogebra colgado en el webcampus y elegir estratégicamente varios puntos de partida x_0. Explicar por qué en el caso de f, si uno elige α lo suficientemente pequeña, siempre converge al mismo valor y en el caso de g no. Comprobar que si α es lo suficientemente grande el algoritmo NO converge, explicar por qué.

HW 4

- Vamos a programar un optimizador (hacerlo en Python):

2. 2-D:

   a. Dado $f(x)$, donde $\boldsymbol{x} = \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \in \mathbb{R}^2$, queremos aproximar el mínimo (local).

   b. Elegimos un valor pequeño de $\Delta x = \Delta y$ para calcular derivadas, una constante α que va a determinar el tamaño del paso, y otra constante pequeña ε que va a determinar cuándo paramos.

   c. Creamos tres listas vacías: X, Y, y F que vamos a ir llenando.

   d. Empezamos en un punto arbitrario $\begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$, $x_i, i = 0,$, $y_i, i = 0$, evaluamos $f(x_i, y_i)$. Hacemos $X[0] = x_0$, $\mathrm{Y}[0] = y_0$ y $F[0] = f(x_0, y_0)$.

   e. Calculamos numéricamente las componentes del gradiente en ese punto: $\nabla_x f(x_i, y_i) = \frac{f(x_i+\Delta x, y_i) - f(x_i, y_i)}{\Delta x}$, $\nabla_y f(x_i, y_i) = \frac{f(x_i, y_i+\Delta y) - f(x_i, y_i)}{\Delta y}$.

   f. Calculamos $x_{i+1} = x_i - \alpha \cdot \nabla_x f(x_i, y_i)$, $y_{i+1} = y_i - \alpha \cdot \nabla_y f(x_i, y_i)$ y $f(x_{i+1}, y_{i+1})$. Hacemos $X[i+1] = x_{i+1}$, $\mathrm{Y}[i+1] = y_{i+1}$ y $F[i+1] = f(x_{i+1}, y_{i+1})$

   g. Verificamos si $|f(x_{i+1}, y_{i+1}) - f(x_i, y_i)| < \varepsilon$. Si se cumple la condición, frenamos, sino volvemos al paso 4 con $i = i + 1$ y repetimos hasta que la condición se verifique (como para que no ocurra que nunca pare si la función no tuviera mínimo, podemos además forzar que si $i = i_{max}$ para algún i_{max}, el programa termina).

   h. Los outputs de la función deben los valores i_{ultimo}, y las listas X, Y y F. Los valores $X[i_{ultimo}] = x_{ultimo}$, $\mathrm{Y}[i_{ultimo}] = y_{ultimo}$ y $F[i_{ultimo}] = f(x_{ultimo})$ corresponden a una ε-aproximación al mínimo local buscado. Pero además tenemos toda la historia del camino y los valores de la función que la misma fue tomando.

   i. Graficar en un mismo grafico en el plano (x, y), para las funciones $f(x, y) = (x$-1)**4 $+$ $(y$-1)**4 - $(x$-1)**2 - $(y$-1)**2 $+$ x $+ y$, y $g(x, y) = (x-1) ** 4 + (y-1) ** 4 - 2(x-1) ** 2 - 3(y-1) ** 2 + x + y$ (ver esas funciones en el JN y en el Geogebra colgados en el webcampus):

      i. Las curvas de nivel.

      ii. El camino $(X[i], Y[i])$ que hizo la maquina para llegar el mínimo. Notar que este camino (si α es lo suficientemente chico) es siempre perpendicular a las curvas de nivel.

      iii. En función de los gráficos de dichas funciones, elegir estratégicamente puntos iniciales y explicar por qué $f(x, y)$ siempre converge al mismo punto mientras que g(x, y) no.

      iv. Comprobar que si α es lo suficientemente grande el algoritmo NO converge, explicar por qué.