**Lab #4**

**PROBLEMA 1**

**Parte 1 – Solución Cerrada**

En la práctica no se suele aplicar una solución cerrada para la resolución del problema de mínimos cuadrados ya que para altos volúmenes de datos el cálculo se vuelve computacionalmente demandante.

**[snapshot interfaz]**

**Parte 2 – GD**

Un step size muy pequeño puede ocasionar un proceso de aprendizaje lento, por lo cual la función objetivo a minimizar requiere de un mayor número de iteraciones para converger, mientras un valor muy grande puede ocasionar que diverja. En este ejercicio el primer valor 0.00005 es muy pequeño por lo cual en la gráfica se observa que su error disminuye más lento que los otros dos. El step size que mejor funciona en esta práctica es el step size mayor con un valor de 0.0007.

**[snapshot graficas]**

Chart, line chart

Description automatically generated

**Parte 3 – SGD**

En el caso de SGD el step size se comporta igual que en lo descrito para GD, sin embargo, es importante tomar en consideración que dado la actualización de los parámetros de forma recurrente que genera este algoritmo, la función objetivo fluctúa y no es monótona decreciente. En general el step size de 0.01 genera valores más bajos que los demás.

**[snapshot graficas]**

Chart, line chart, histogram

Description automatically generated

**Parte 4 – MBGD**

* Un mini batch size muy grande generar mayor error en el conjunto de datos, por lo cual el mini batch de 25 es el que presenta mejor desempeño
* Los mejores resultados se obtienen con un step size de 0.01 y un mini batch size de 25, esto dado que presenta el menor error de todos los valores evaluados.

**[snapshot graficas]**

Chart

Description automatically generated

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

**Parte 5 – Comparación**

|  |  |  |  |
| --- | --- | --- | --- |
| **Método** | **Valor óptimo (f\*)** | **No. Iteraciones** | **Error (x\*, x\_true)** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**PROBLEMA 2**

**Parte 1 – GD con Backtracking Line Search**

|  |  |  |
| --- | --- | --- |
| **X0** | **Error Backtracking** | **Error LR constante** |
| (0, 0) | Inf | 0.69 |
| (0.6, 0.6) | 0.17 | 0.22 |
| (-0.5, 1) | 0.77 | 2.18 |
| (-1.2, 1) | 2.09 | 2.11 |

Usando un step size constante de 0.0005 es posible notar que backtracking funciona mejora para todos los puntos iniciales con excepción del punto (0, 0). Adicionalmente la selección del step size constante fue a través de experimentación y evaluación de resultados pues valores mayores a utilizado divergen, mientras que backtracking determina el step size de forma automática según el algoritmo.

**[snapshot graficas]**

**Parte 2 – Método de Newton con Backtracking Line Search**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X0** | **Error Backtracking** | **Iteraciones Backtracking** | **Error LR constante** | **Iteracions LR constante** |
| (0, 0) | 0 | 343 | 0 | 3 |
| (0.6, 0.6) | 4.03 | 3000 | 0 | 6 |
| (-0.5, 1) | 11.21 | 3000 | 0 | 6 |
| (-1.2, 1) | 53.83 | 3000 | 7 | 0 |

En el caso del método de Newton este algoritmo se beneficia de tener un step size constante en contraste de calcularlo a través de backtracking, pues el número de iteraciones es menor al tener un step size constante de 1 mientras con backtracking solo en el primer punto inicial logra converger a 0.

**[snapshot graficas]**