Programming part

I. Problem 5

A. Closest point projection implementation

Firstly, to be able to use Projected Gradient Descent (PGD), we needed to implement projections onto the domains of interest. We implemented projections onto three different domains, as follows:

- A circle with center at $(0,0)$ and radius $\sqrt{1.5}$
- A square: $[-2,2] \times [-2,2]$
- A triangle with vertices $(-1,-1)$, $(1.5,-1)$, and $(-1,1.5)$

Figure 1 shows the projections of various points onto these domains.

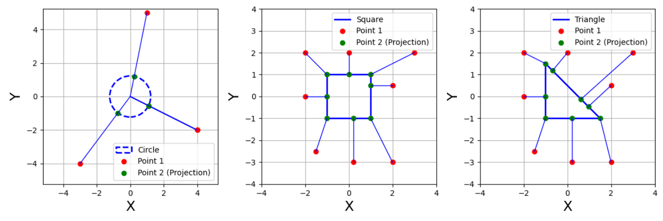

Figure 1. Closest point projection to different domains.

B. Using PGD on the function

We applied the Projected Gradient Descent (PGD) method to the function:

$$f(x,y) = x^2 + e^x + y^2 - xy$$

with the starting point $x_1 = (1,1)$. This function is α-strongly convex, β-smooth, and L-Lipschitz continuous on the three domains used. The gradient of the function is given by:

$$\nabla f = (2x + e^x - y, 2y - x)$$

For each domain, we performed PGD using three different learning rates, denoted by γ:

- $\gamma_1 = \frac{\|x_1 - x^*\|}{L\sqrt{T}}$
- $\gamma_2 = \frac{1}{\beta}$
- $\gamma_3 = \frac{2}{\alpha(k+1)}$, where k is the iteration number

We conducted 10 steps of gradient descent for each combination of learning rate γ and domain. The final positions after these steps are presented in Table I.

The approximate position of the minimum is $x^* = (-0.43, -0.22)$, and as shown in the table, the closest approach to this minimum is achieved with γ_3 on the circular domain.

Additionally, the PGD steps for all combinations of domains and learning rates are shown in Figure 2. We observe that for γ_3, the learning rate is initially too large, but it adapts and decreases with each iteration, eventually leading to the closest convergence to the minimum.

	Circle	**Square**	**Triangle**
γ_1	$[-0.4799,\ 0.3517]$	$[-0.5271,\ 0.4092]$	$[-0.5271,\ 0.4092]$
γ_2	$[-0.3496,\ -0.0682]$	$[-0.3484,\ -0.0602]$	$[-0.3484,\ -0.0602]$
γ_3	$[-0.4345,\ -0.2190]$	$[-0.4263,\ -0.2077]$	$[-0.4236,\ -0.2039]$

Table I
Convergence results for circle, square, and triangle domains for different learning rates.

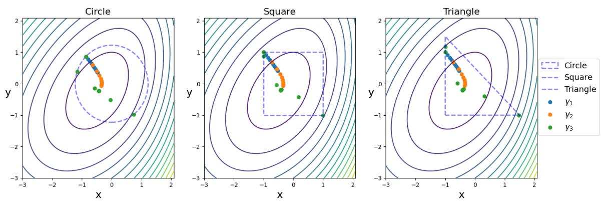

Figure 2. PGD for different learning rates and domains

We also checked the theoretical guarantees from Theorem 3.3 in the notes. The results are presented in Table II. As observed, all the guarantees are easily satisfied. Additionally, we note that the results do slightly vary with different domains.

Condition	**Circle**	**Square**	**Triangle**
Cond 1	$0.8554 \le 6.2477$	$1.1567 \le 5.4298$	$1.1567 \le 6.2477$
Cond 2	$0.0188 \le 5.1256$	$0.0206 \le 6.7972$	$0.0206 \le 6.7972$
Cond 3	$0.0188 \le 0.7704$	$0.0206 \le 1.0200$	$0.0206 \le 1.0200$
Cond 4	$6.49 \cdot 10^{-5} \le 36.7375$	$0.0046 \le 36.7375$	$0.0101 \le 36.7375$

Table II
Convergence guarantees for circle, square, and triangle domains.

II. Problem 6

In this problem we tried to find the global minimum of the function:

$$f(z) = -\sum_{i=1}^{4} c_i \cdot e^{-\sum_{j=1}^{3} a_{ij}(z-p_{ij})^2} \tag{1}$$

using gradient descent. The partial derivative w.r.t. z_k is calculated as follows:

$$\frac{\partial f}{\partial z_k} = 2\sum_{i=1}^{4} c_i \cdot a_{ik}(z_k - p_{ik}) \cdot e^{-\sum_{j=1}^{3} a_{ij}(z-p_{ij})^2} \tag{2}$$

The gradient is then

$$\nabla f(z) = (z_1, z_2, z_3) \tag{3}$$

When experimenting with different starting points and learning rates for gradient descent (GD), we observed that the procedure often gets stuck at local minima. This led us to believe that the function has multiple local minima. To address this, we performed a grid search over our domain, which is $[0,1]^3$. The search evaluated the function values at different points, with a step size of 0.1 (resulting in 1000 steps in total) across the three variables.

From this grid search, we identified the point with the smallest function value, located at $(0.1, 0.6, 0.9)$, as our starting point for GD descent. According to the instructions in the Homework, this point was a good choice since it allowed us to get quite close to the reported approximate minimum. The function values and corresponding positions for different learning rates are presented in Table III. We can see that the closes result was achieved for learning rates 0.008 or lower. Note that we stopped the methoed when the gradient step was less than 10^{-15}.

γ	**Iterations**	**Final Point**	**Function Value**
1	1	[0.09302137, -2.096733, -8.01761408]	-0.0
0.1	1	[0.1, 0.6, 0.9]	-3.586922560937449
0.0102	2087	[0.11461434, 0.55564885, 0.85254695]	-3.8627821478207554
0.01	2128	[0.11461434, 0.55564885, 0.85254695]	-3.8627821478207554
0.008	2641	[0.11461434, 0.55564885, 0.85254695]	-3.862782147820755
0.005	4156	[0.11461434, 0.55564885, 0.85254695]	-3.862782147820755
0.001	19499	[0.11461434, 0.55564885, 0.85254695]	-3.862782147820755
0.0001	176068	[0.11461434, 0.55564885, 0.85254695]	-3.862782147820755

Table III
Results for different learning rates (γ) showing the number of iterations, final point, and function value.