Convex Optimization Hw1

Created	@April 23, 2025 10:30 PM
Class	Convex Optiization
Type	Hw
Reviewed	☐

學號：B10901052

姓名：聶臣璟

Problem 1

1(a)

First consider the convexity of function $f(x) = \log(1 + \exp(-x))$,

$dom\ f = R,\ f$ is diffrentiable on its domain.

$f'(x) = \frac{-\exp(-x)}{(1+\exp(-x))}$

$f''(x) = \frac{\exp(-x)}{(1+\exp(-x))^2} > 0$

Since $f(x)$ is twice differentiable with $dom\ f = R$, and its second derivative f''(x) is non-negative, $f(x) = \log(1 + \exp(-x))$ is convex.

Since $y_i x_i^T \beta$ is an affine function,

Using the fact that "**The composition of a convex function with an affine function remains convex",**

We can conclude that $f(\beta) = \log(1 + \exp(-y_i x_i^T \beta))$ is convex.

QED.

1(b)

We know that:

1. If f is convex and $\alpha > 0$, then αf is also convex.

2. If both f_1 and f_2 are convex, then $f_1 + f_2$ is also convex.
3. Every norm on R^n is convex.
4. If g is convex and non-negative and p≥1, then $g(x)^p$ is convex.

By 1(a), we know that $\log(1+\exp(-y_i x_i^T \beta))$ is convex.

This means that by 2 above,

$\sum_{i=1}^m \log(1+\exp(-y_i x_i^T \beta))$ is convex.

And thus by 1,

$\frac{1}{m}\sum_{i=1}^m \log(1+\exp(-y_i x_i^T \beta))$ is convex.

Knowing that by 3,

$||\beta||_2$ is convex and non-negative, so that by 4,

$||\beta||_2^2$ is convex.

Lastly, since $\lambda > 0$, by 1 and 2,

$f(\beta) = \frac{1}{m}\sum_{i=1}^m \log(1+\exp(-y_i x_i^T \beta)) + \lambda||\beta||_2^2$ is convex.

QED.

1(c)

optimal value $f(\beta) = 0.6817473135293762$

optimal point

$\beta =$
$[-0.04759088,\ -0.06531756,\ -0.03539397,\ -0.01375628,\ -0.05236341]$

1(d)

optimal value $f(\beta) = 0.6802176408190813$

X, y, and optimal β is in X.csv, y.csv, and beta.csv respectively.

Problem 2

2(a)

Optimal value = 1.0

Optimal x1 = 1.0000000000220999

Optimal x2 = 0.9999999999779001

2(b)

The optimal solution is not unique.

Any feasible solution that satisfied the constraint $x_1 + x_2 = 2$ makes the objective $(x_1 + x_2 - 1)^2 = 1$. x_1 can be any real number between 1 and 2, which means, $x_1 \in [1, 2]$, and $x_2 = 2 - x_1$.

The solver might be designed to return only one single feasible solution that satisfies the constraints and minimizes the objective function, even if there are multiple optimal solutions. The selection of the solution may be based on its internal algorithms and numerical precision.

2(c)

optimal $\{x_1, x_2\} = \{x_1, x_2 | x_1 + x_2 = 2, x_1 >= 1, x_2 >= 0\}$

parametric expression:

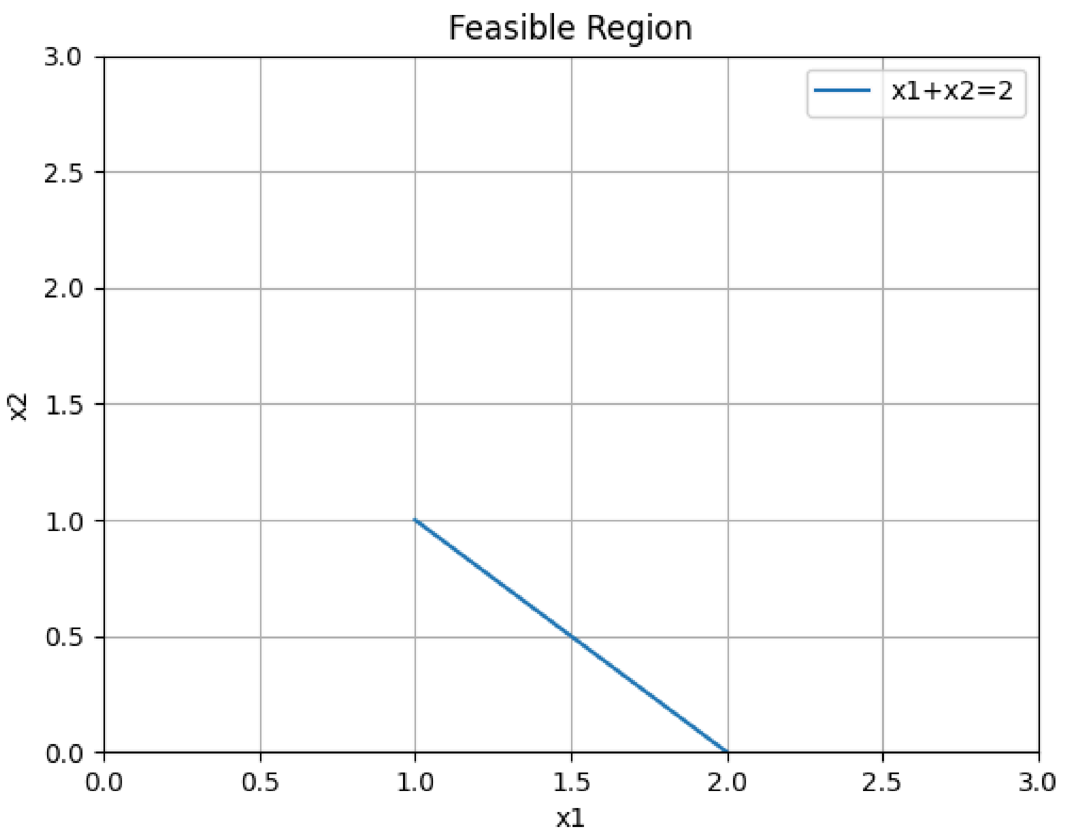
$t \in [1, 2]$

$x_1 = t$

$x_2 = 2 - t$

2(d)

In this problem, the feasible set is the same as the optimal set, which is shown in 2(c).



2(e)

The optimal solution should be unique, which should be $x_1 = 2, x_2 = 0$. Since $x_1 + x_2 = 2$ is fixed to satisfy the constraint, to minimize the objective function, we need to minimize x_2^2. As a result, $x_2 = 0$, and thus $x_1 = 2$, is the optimal solution.

$\epsilon = 1$

Optimal value = 1.0
Optimal x1 =
2.0
Optimal x2 =
$-7.287175973681554 \times 10^{-23}$

$\epsilon = 10^{-4}$

Optimal value = 1.0
Optimal x1 =
1.9999999903894259

Optimal x2 =
$9.610574160533399 \times 10^{-09}$

$\epsilon = 10^{-10}$

Optimal value = 1.00000000009992
Optimal x1 =
1.0003998998469619
Optimal x2 =
0.9996001001530382

The optimal solutions derived from python program shows that when ϵ is bigger, the derived solution is closer to the actual optimal solution. This problem is caused by the precision of the computer. When calculating ϵx_2^2, if ϵ is too small, the total number will be too small for the computer to correctly record, leading to a deviation to the actual optimal solution. As shown in the data above, the solution is pretty close to the actual solution when $\epsilon = 1$ and $\epsilon = 10^{-4}$. However, when $\epsilon = 10^{-10}$, the term ϵx_2^2 will be too small, leading to an obvious deviation to the actual solution and get a solution much more similar to 2(a).