CS164 Assignment 2

Viet Hoang Tran Duong

Fall 2019

Part 1:

$$\min_{x} c^T x$$

Subject to:

$$Ax \le b$$

Where $A \in R^{m\times n}, b \in R^{m\times 1}, c \in R^{n\times 1}$ ($c \neq 0$ or else $c^T x = 0$: no need for optimization), and $x \in R^{n\times 1}$ (*)

Objective function: $f(x) = c^T x$

Constraint: $g(x) = Ax - b \le 0$: This is a set of constraints:

Call M_i: the i_th row of matrix M

$g_i(x) = A_i x - b_i \le 0, i = 1,2,\dots,m.$

The objective function is a linear function, which means it is affine, implying it is a convex function (and concave). Therefore, we can apply the KKT conditions. If a point satisfies the KKT conditions, it satisfies the necessary conditions to be the optimal point of the system.

In this case, we only have the inequality constraints, hence, the KKT conditions for convex problem with inequality constraints state that: If the KKT conditions below hold for x^* and some λ^*, then $x^*$is the optimal point of the initial system:

- $\lambda \in R^{m\times 1}$.
- Stationarity:
  $\nabla_x f(x) + \sum_{i=1}^{m} \lambda_i * \nabla_x g_i(x) = 0$
  $\Leftrightarrow \nabla_x (c^T x) + \sum_{i=1}^{m} \lambda_i * \nabla_x g_i(x) = 0$
  $\Leftrightarrow \nabla_x (c^T x) + \sum_{i=1}^{m} \lambda_i * \nabla_x (A_i x - b_i) = 0$ (K1)
- Inequality constraints & Complementary slackness condition:
  - (complementary slackness): $\lambda_i * g_i(x) = 0 \ \forall\, i = 1,2,\dots,m$ (K2)
  - (dual feasibility): $\lambda \ge 0$ (K3)
  - (primal feasibility): $Ax - b \le 0$ (K4)
- There is no equality constraint in this case.

Assumption 1: Assuming the optimal solution x^* is found in the interior of the feasible region: this means that there is no inequality constraint that are active: (**)

$g_i(x^*) \neq 0 \ \forall\, i = 1,2,\dots,m$ (1)

From (K2) and (1) $\Rightarrow \lambda_i = 0 \ \forall i = 1,2,\dots,m$ (2)
(because $\lambda_i * g_i(x) = 0$ and $g_i(x^*) \neq 0$)

From (2), plugging $\lambda_i = 0 \ \forall i = 1,2,\dots,m$ into (K1):

$\nabla_x(c^T x) + \sum_{i=1}^{m} \lambda_i * \nabla_x(A_i x - b_i) = 0$

$\Leftrightarrow\ \nabla_x(c^T x) + \sum_{i=1}^{m} 0 * \nabla_x(A_i x - b_i) = 0$

$\Leftrightarrow\ \nabla_x(c^T x) = 0$

$\Leftrightarrow\ c^T = 0$: violate (*).

Hence, assumption 1 (**) is incorrect. Hence, x^* cannot be found in the interior of the feasible region $\Rightarrow$ if the optimal solution x^* exists, x^* is on the intersection of the active constraints: either vertex, line, or facets of the feasible polytope.

Geometry intuition:

If the feasible set is empty, then no optimal solution. If the feasible set is nonempty and unbounded, the LP may or may not attain an optimal solution, depending on the cost direction c.

If the feasible set is nonempty and bounded, then LP attains an optimal solution x^* such that $c^T x \geq c^T x^*\ \forall x \in D$. The objective can be improved if exist $x \in D$: $c^T(x - x^*) < 0$. Geometrically, this improvement means that if there exists a point x_0 in the intersection of D and the open half-space $\{x: c^T(x - x^*) < 0\}$, that we can move in the descent direction c while maintaining feasibility. As x^* is the optimal, hence, there is no feasible descent direction at x^*, which implies x^* is at the boundaries of the feasible set, which is either vertex, line, or facets of the feasible polytope. In particular, the optimal solution is unique if the optimal cost hyperplane $\{x: c^T x = p^*\}$ intersects the feasible polytope only at a vertex.

Part 2:

2.1. $\boldsymbol{l_1}$ norm:

Problem: $\min_{\Theta} \left|\left|Y - X\Theta\right|\right|_1 = \min_{\Theta} 1^T |Y - X\Theta|$

We introduce a slack variable $s, s \in R^{n\times 1}$ such that: $|Y - X\Theta| \leq s$

$\Leftrightarrow -s \leq Y - X\Theta \leq s$
$\Leftrightarrow\ -X\Theta - s \leq\ -Y$ and $X\Theta - s \leq Y$
$\Leftrightarrow \begin{bmatrix} -X & -I \\ X & -I \end{bmatrix} \begin{bmatrix} \Theta \\ s \end{bmatrix} \leq \begin{bmatrix} -Y \\ Y \end{bmatrix}$

Then, we can convert the problem into
$\min_{s,\Theta} 1^T s$ subject to $Ax \leq b$ where $A = \begin{bmatrix} -X & -I \\ X & -I \end{bmatrix},\ x = \begin{bmatrix} \Theta \\ s \end{bmatrix},\ b = \begin{bmatrix} -Y \\ Y \end{bmatrix}$

Or we can rewrite as $\min_{x} c^T x$ subject to $Ax \leq b$, where:
$A = \begin{bmatrix} -X & -I \\ X & -I \end{bmatrix},\ x = \begin{bmatrix} \Theta \\ s \end{bmatrix},\ b = \begin{bmatrix} -Y \\ Y \end{bmatrix},\ c = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ (2 values of 0 at the beginning and n values of 1)

2.2. $\boldsymbol{l_\infty}$ norm

Problem: $\min\limits_{\Theta}||Y - X\Theta||_\infty = \min\limits_{\Theta}(\max|Y - X\Theta|)$

We introduce a slack variable $s \in R$: such that: $\max|Y - X\Theta| \le s$

$\Leftrightarrow |Y_i - X_i\Theta| \le\ s\ \forall i = 1,2,\dots,n$

$\Leftrightarrow -s \le\ Y_i - X_i\Theta \le\ s\ \forall i = 1,2,\dots,n$

$\Leftrightarrow\ -X_i\Theta - s \le\ -Y_i$ and $X_i\Theta - s \le Y_i\ \ \forall i = 1,2,\dots,n$

$\Leftrightarrow \begin{bmatrix} -X & -I \\ X & -I \end{bmatrix}\begin{bmatrix} \Theta \\ 1 * s \end{bmatrix} \le \begin{bmatrix} -Y \\ Y \end{bmatrix}$ where 1 in $1 * s$ is the column vector of 1 with length n.

Then, we can convert the problem into

$\min\limits_{s,\Theta} s$ subject to $Ax \le b$ where $A = \begin{bmatrix} -X & -I \\ X & -I \end{bmatrix}$, $x = \begin{bmatrix} \Theta \\ 1 * s \end{bmatrix}$, $b = \begin{bmatrix} -Y \\ Y \end{bmatrix}$

where 1 in $1*\ s$ in variable x is the column vector of 1 with length n.

Or we can rewrite as $\min\limits_{x} c^T x$ subject to $Ax \le b$, where:

$A = \begin{bmatrix} -X & -I \\ X & -I \end{bmatrix}$, $x = \begin{bmatrix} \Theta \\ 1 * s \end{bmatrix}$, $b = \begin{bmatrix} -Y \\ Y \end{bmatrix}$, $c = \begin{bmatrix} 0 \\ \frac{1}{n} \end{bmatrix}$ (2 values of 0 at the beginning and n values of $\frac{1}{n}$)

where 1 in $1*\ s$ in variable x is the column vector of 1 with length n.

The reason for having $\frac{1}{n}$ is because if not, then the objective functions will be ns, which dependent on n: number of data.

Part 3: for np.random.seed(2021)

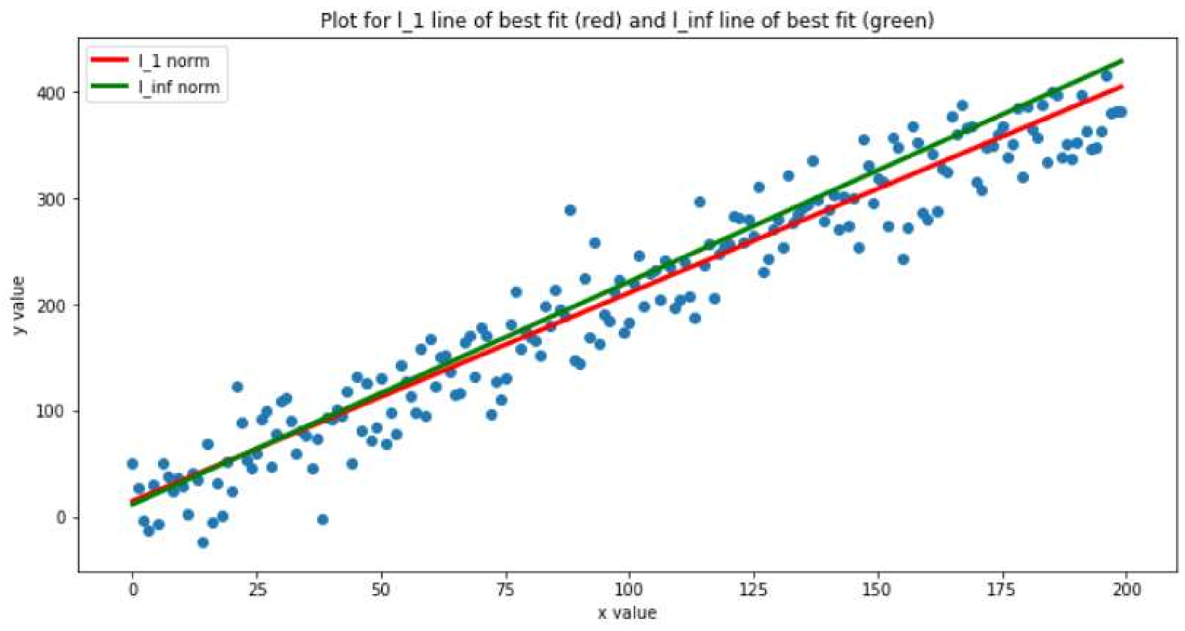


Figure 1: The plot for l_1 line of best fit (red) and l_∞ line of best fit (green)

For l_1 norm

```
Results for l_1 norm:
Status: optimal
Optimal value: 4809.433318703739
Achieve at: theta = [[ 1.96249763], [14.64189308]]
Function: y = 1.9624976288329208x + 14.641893083017035
```

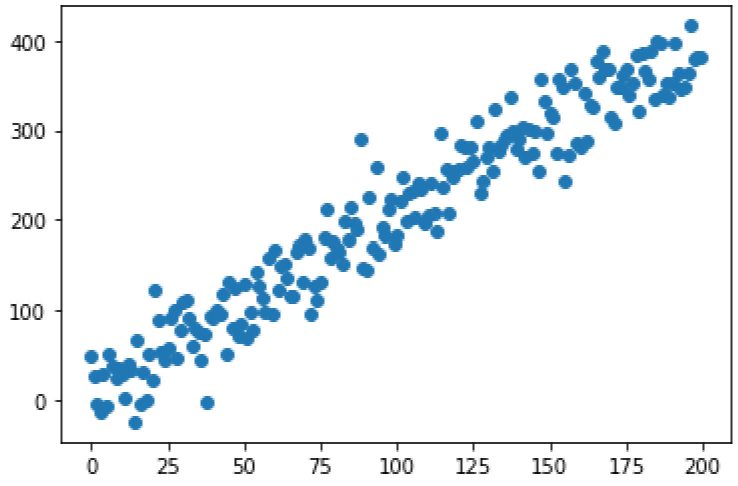
For l_∞ norm

```
Results for l_inf norm:
Status: optimal
Optimal value: 93.9827687411251
Achieve at: theta = [[ 2.10004311], [11.3752135]]
Function: y = 2.100043107358015x + 11.375213502763392
```

Code available at:
https://colab.research.google.com/drive/1TpPdxZ5xEyRZgcN6PXADLKboIIiSiwGs

Data generating

```
# l_1 and l_infinity regression using cvxpy
import numpy as np
import cvxpy as cvx
import matplotlib.pyplot as plt

# set seed for consistent results -> Minerva Class of 2021
np.random.seed(2021)

# generate a synthetic dataset
# actual parameter values
theta1_act = 2
theta2_act = 5

# Number of points in dataset
N = 200

# Noise magnitude
mag = 30

# datapoints
x = np.arange(0,N)
y = theta1_act * x + theta2_act *np.ones([1,N]) + np.random.normal(0,mag,N)

plt.figure()
# Scatter plot of data
plt.scatter(x,y)
plt.show()

```



```
# initialize the X = [x_i,1]: stacking x and a vector of 1
# it will have the dimension (2,N) -> transpose to have (N,2) shape
X = np.array([x, np.ones(N)]).T

# y has shape (1,N) -> transpose to have (N,1)
Y = np.array(y.T)
```

L_1 norm (l1)

```
# Define the variables
theta_l1 = cvx.Variable((2,1))
s_l1 = cvx.Variable((N, 1))

# Define the objective function
obj_l1 = cvx.Minimize(cvx.norm(s_l1, 1))

# Define the constraints
con_l1 = [Y - X*theta_l1 <= s_l1, Y - X*theta_l1 >= -s_l1]

# Solve the optimization problem
prob_l1 = cvx.Problem(obj_l1,con_l1)
prob_l1.solve(solver=cvx.ECOS)
```

```
prob_l1.solve(solver=cvx.ECOS)
print("Results for l_1 norm:")
print("Status:", prob_l1.status)
print("Optimal value:", prob_l1.value)
print("Achieve at: theta =", theta_l1.value)
print("Function: y = {}x + {}".format(theta_l1.value[0][0],theta_l1.value[1][0]))
```

```
Results for l_1 norm:
Status: optimal
Optimal value: 4809.433318703739
Achieve at: theta = [[ 1.96249763]
 [14.64189308]]
Function: y = 1.9624976288329208x + 14.641893083017035
```

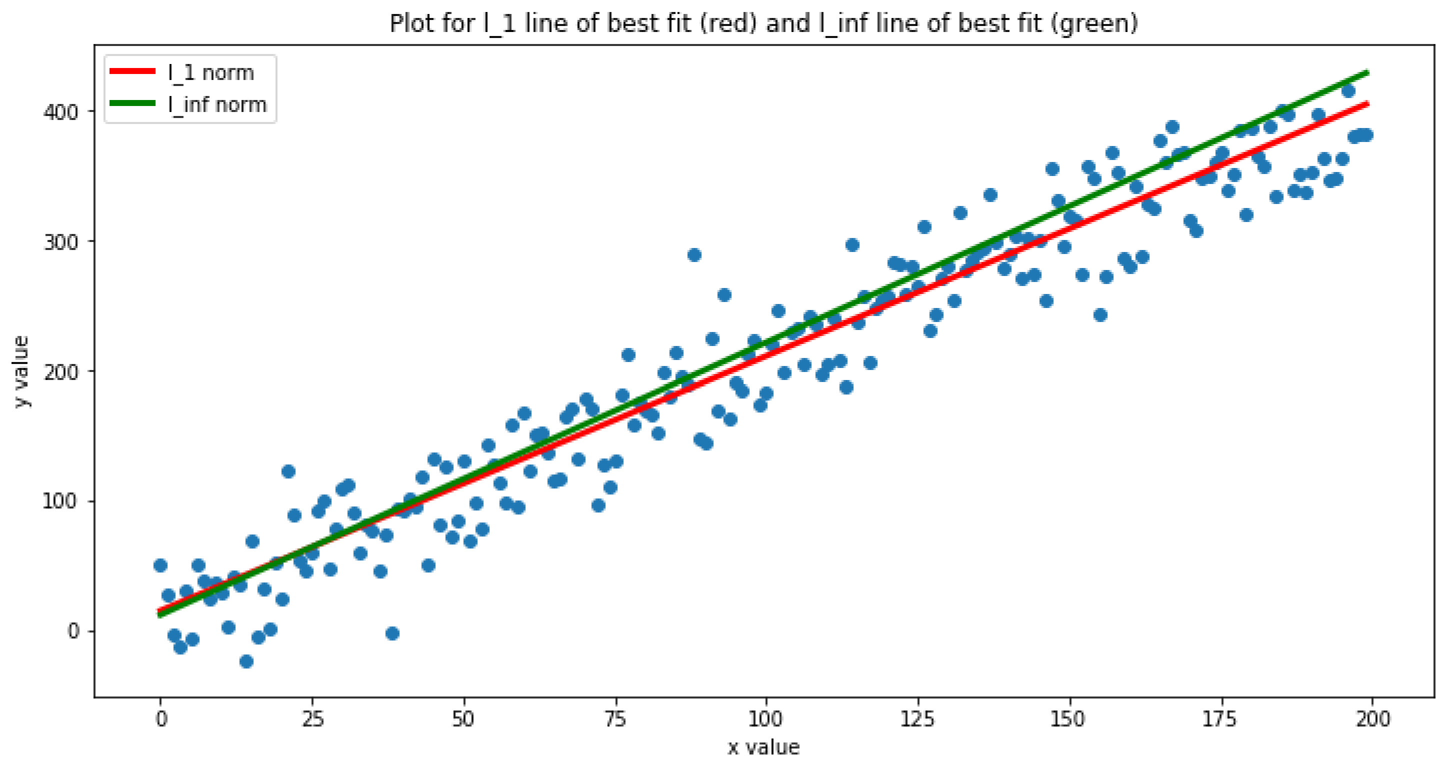
L_Infinity norm (linf)

```
# Define the variables
theta_linf = cvx.Variable((2,1))
s_linf = cvx.Variable((1,1))

# vector of 1s
ones = np.ones((N,1))
# Define the objective function
obj_linf = cvx.Minimize(s_linf)

# Define the constraints
con_linf = [Y - X*theta_linf <= s_linf@ones, Y - X*theta_linf >= -s_linf@ones]

# Solve the optimization problem
prob_linf = cvx.Problem(obj_linf,con_linf)
prob_linf.solve(solver=cvx.ECOS)
print("Results for l_inf norm:")
print("Status:", prob_linf.status)
print("Optimal value:", prob_linf.value)
print("Achieve at: theta =", theta_linf.value)
print("Function: y = {}x + {}".format(theta_linf.value[0][0],theta_linf.value[1][0]))
```

```
Results for l_inf norm:
Status: optimal
Optimal value: 93.9827687411251
Achieve at: theta = [[ 2.10004311]
 [11.3752135 ]]
Function: y = 2.100043107358015x + 11.375213502763392
```

Plot lines of best fit

```python
import matplotlib.pyplot as plt

plt.figure(figsize=(12,6))
# l1 norm
y_l1 = X@theta_l1.value
plt.plot(x, y_l1, color = "red", linewidth = 3, label = "l_1 norm")

# linf norm
y_linf = X@theta_linf.value
plt.plot(x, y_linf, color = 'green', linewidth = 3, label = 'l_inf norm')
plt.legend()

# Scatter plot of data
plt.scatter(x,y)
plt.xlabel("x value")
plt.ylabel("y value")
plt.title("Plot for l_1 line of best fit (red) and l_inf line of best fit (green)")
plt.show()
```


Plot for l_1 line of best fit (red) and l_inf line of best fit (green)
l_1 norm
l_inf norm
400
300
200
100
0
y value
0
25
50
75
100
125
150
175
200
x value