Hai - Han Sun

ASU ID : 1225435731

ME598/494 Homework 2

1. (20 points) Show that the stationary point (zero gradient) of the function

$$f(x_1, x_2) = 2x_1^2 - 4x_1x_2 + 1.5x_2^2 + x_2$$

is a saddle (with indefinite Hessian).

Find the directions of downslopes away from the saddle. To do this, use Taylor's expansion at the saddle point to show that

$$f(x_1, x_2) = f(1, 1) + (a\partial x_1 - b\partial x_2)(c\partial x_1 - d\partial x_2),$$

with some constants a, b, c, d and $\partial x_i = x_i - 1$ for $i = 1, 2$. Then the directions of downslopes are such $(\partial x_1, \partial x_2)$ that

$$f(x_1, x_2) - f(1, 1) = (a\partial x_1 - b\partial x_2)(c\partial x_1 - d\partial x_2) < 0.$$

2. (a) (10 points) Find the point in the plane $x_1 + 2x_2 + 3x_3 = 1$ in $\mathbb{R}^3$ that is nearest to the point $(-1, 0, 1)^T$. Is this a convex problem? Hint: Convert the problem into an unconstrained problem using $x_1 + 2x_2 + 3x_3 = 1$.

   (b) (40 points) Implement the gradient descent and Newton's algorithm for solving the problem. Attach your codes in the report, along with a short summary of your findings. The summary should include: (1) The initial points tested; (2) corresponding solutions; (3) A log-linear convergence plot.

3. (5 points) Prove that a hyperplane is a convex set. Hint: A hyperplane in $\mathbb{R}^n$ can be expressed as: $\mathbf{a}^T\mathbf{x} = c$ for $\mathbf{x} \in \mathbb{R}^n$, where $\mathbf{a}$ is the normal direction of the hyperplane and c is some constant.

4. (15 points) Consider the following illumination problem:

$$\min_{\mathbf{p}} \quad \max_{k}\{h(\mathbf{a}_k^T\mathbf{p}, I_t)\}$$
$$\text{subject to:} \quad 0 \leq p_i \leq p_{\max},$$

where $\mathbf{p} := [p_1, ..., p_n]^T$ are the power output of the n lamps, $\mathbf{a}_k$ for $k = 1, ..., , m$ are fixed parameters for the m mirrors, I_t the target intensity level. $h(I, I_t)$ is defined as follows:

$$h(I, I_t) = \begin{cases} I_t/I & \text{if } I \leq I_t \\ I/I_t & \text{if } I_t \leq I \end{cases}$$

(a) (5 points) Show that the problem is convex.

(b) (5 points) If we require the overall power output of any of the 10 lamps to be less than p^*, will the problem have a unique solution?

(c) (5 points) If we require no more than 10 lamps to be switched on ($p > 0$), will the problem have a unique solution?

5. (10 points) Let $c(x)$ be the cost of producing x amount of product A and assume that $c(x)$ is differentiable everywhere. Let y be the price set for the product. Assuming that the product is sold out. The total profit is defined as

$$c^*(y) = \max_x \{xy - c(x)\}.$$

Show that $c^*(y)$ is a convex function with respect to y.

1. (20 points) Show that the stationary point (zero gradient) of the function

$$f(x_1, x_2) = 2x_1^2 - 4x_1x_2 + 1.5x_2^2 + x_2$$

is a saddle (with indefinite Hessian).

Find the directions of downslopes away from the saddle. To do this, use Taylor's expansion at the saddle point to show that

$$f(x_1, x_2) = f(1,1) + (a\partial x_1 - b\partial x_2)(c\partial x_1 - d\partial x_2),$$

with some constants a, b, c, d and $\partial x_i = x_i - 1$ for $i = 1, 2$. Then the directions of downslopes are such $(\partial x_1, \partial x_2)$ that

$$f(x_1, x_2) - f(1,1) = (a\partial x_1 - b\partial x_2)(c\partial x_1 - d\partial x_2) < 0.$$

$f(1,1) = 2 - 4 + 1.5 + 1 = 0.5$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 4x_1 - 4x_2 \\ -4x_1 + 3x_2 + 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \therefore x_1 = x_2$$

$x_1 = x_2 = 1$

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} \end{bmatrix} = \begin{bmatrix} 4 & -4 \\ -4 & 3 \end{bmatrix}$$

$\therefore$ Taylor :

$$f(x_1, x_2) \approx f(x_0) + \nabla f(x_0)^T (x - x_0) + \frac{1}{2}(x - x_0)^T H(x_0)(x - x_0)$$

$$x_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\therefore f(x_1, x_2) \approx 0.5 + 0 + \frac{1}{2}\begin{bmatrix} x_1 - 1 & x_2 - 1 \end{bmatrix}\begin{bmatrix} 4 & -4 \\ -4 & 3 \end{bmatrix}\begin{bmatrix} x_1 - 1 \\ x_2 - 1 \end{bmatrix}$$

$$\approx 0.5 + \frac{1}{2}\begin{bmatrix} 4x_1 - 4x_2 & -4x_1 + 3x_2 + 1 \end{bmatrix}\begin{bmatrix} x_1 - 1 \\ x_2 - 1 \end{bmatrix}$$

$$\approx 0.5 + \frac{1}{2}\left[4x_1^2 \cancel{- 4x_1} - 4x_1x_2 + 4x_2 - 4x_1x_2 + 3x_2^2 + x_2 \cancel{+ 4x_1} - 3x_2 - 1\right]$$

$$\approx 0.5 + \frac{1}{2}\left[4x_1^2 + 3x_2^2 - 8x_1x_2 + 2x_2 - 1\right]$$

$$\therefore f(x_1, x_2) - f(1,1) = \frac{1}{2}\left[4x_1^2 + 3x_2^2 - 8x_1x_2 + 2x_2 - 1\right]$$

Find λ

$$|H - \lambda I| = 0$$

$$\begin{vmatrix} 4-\lambda & -4 \\ -4 & 3-\lambda \end{vmatrix} = 0$$

$$\lambda = \frac{\pm\sqrt{65} + 7}{2}$$

$\lambda_{min} = -0.5311$

$$\frac{1}{2}(x - x_0)^T H(x_0)(x - x_0)$$

$$= \frac{1}{2}\Delta x^T H_0 \Delta x + O \cdot \|x - x_0\|^2$$

length, direction

$$\because \Delta x = t \cdot d$$

$$\therefore \frac{1}{2} \cdot t^2 \cdot d^T H_0 d + O \cdot (t^2 \cdot \|x-1\|^2)$$

$$\therefore \frac{1}{2} d^T H_0 d + \frac{O(t^2 \cdot \|x-1\|^2)}{t^2}$$

$$\therefore \lim_{t\to 0} \frac{1}{2} d^T H_0 d + 0 < 0$$

$$\therefore \frac{1}{2}\lambda_{min}(H_0) < 0$$

$$\therefore \lambda_{min}(H_0) < 0$$

#

$$f(x) = f(x^*) + \nabla f(x^*)^T(x - x^*) + \frac{1}{2}(x - x^*)^T H(x^*)(x - x^*)$$

$$+ O \cdot (\|x - x^*\|^2)$$

$\because O(t) \lll t \Rightarrow \lim_{t\to 0} \frac{O(t)}{t} = 0$

①

$$\frac{1}{2}\Delta x^T H^* \Delta x + O \cdot (\|\Delta x\|^2) \geq 0$$

$\Delta x = t \cdot d$ $\quad \|d\|^2 = 1$

length, direction

$$\frac{1}{2} t^2 \cdot d^T H^* d + O \cdot (t^2 \cdot \cancel{\|d\|^2})$$

$$= \frac{1}{2} d^T H^* d + \frac{O(t^2)}{t^2} \geq 0$$

$$\cancel{\lim_{t\to 0}} \frac{1}{2} d^T H^* d + O \geq 0$$

$\exists d \neq 0$ $\quad \|d\|^2 = 1$ $\quad d^T H d_0 \geq \lambda_{min}(H)$

$$\frac{1}{2}\lambda_{min}(H^*) \geq 0$$

$$\therefore \lambda_{min}(H^*) \geq 0$$

#

Find λ $\Rightarrow$ magnitude

$$|H - \lambda I| = 0$$

$$\begin{vmatrix} 4-\lambda & -4 \\ -4 & 3-\lambda \end{vmatrix} = 0$$

$$\lambda = \frac{\pm\sqrt{65} + 7}{2}$$

$\lambda_{min} = -0.5311$

Find Eigenvectors $\Rightarrow$ direction

$$(H - \lambda I) \cdot x = 0$$

① $\lambda = \dfrac{-\sqrt{65} + 7}{2}$

$$\begin{bmatrix} \frac{\sqrt{65}+1}{2} & -4 \\ -4 & \frac{\sqrt{65}-1}{2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

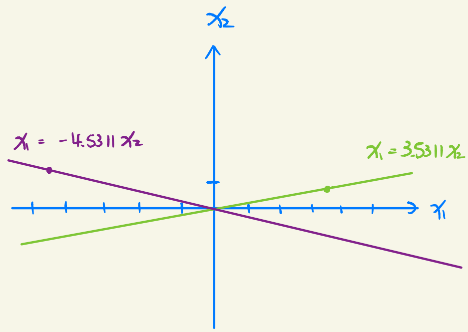
$$x_1 = \frac{\sqrt{65} - 1}{8} x_2$$

$$x_2 = x_2$$

$$\therefore \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \overbrace{\frac{\sqrt{65}-1}{8}}^{3.5311} \\ 1 \end{bmatrix}$$

② $\lambda = \dfrac{\sqrt{65} + 7}{2}$

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \overbrace{\frac{-\sqrt{65}-1}{8}}^{-4.5311} \\ 1 \end{bmatrix}$$



2. (a) (10 points) Find the point in the plane $x_1 + 2x_2 + 3x_3 = 1$ in $\mathbb{R}^3$ that is nearest to the point $(-1, 0, 1)^T$. Is this a convex problem? Hint: Convert the problem into an unconstrained problem using $x_1 + 2x_2 + 3x_3 = 1$.

(b) (40 points) Implement the gradient descent and Newton's algorithm for solving the problem. Attach your codes in the report, along with a short summary of your findings. The summary should include: (1) The initial points tested; (2) corresponding solutions; (3) A log-linear convergence plot.

(a)
(b)

$$\min_{x \in R} \; f(x) = \left[(x_1+1)^2 + x_2^2 + (x_3-1)^2 \right]^{\frac{1}{2}}$$

[I] $x_1 = 1 - 2x_2 - 3x_3$

$$\text{s.t.} \quad x_1 + 2x_2 + 3x_3 - 1 = 0$$

$$f = \left((2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2 \right)^{\frac{1}{2}}$$

$$f = \left((2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2 \right)^{\frac{1}{2}}$$

$(f \pm g)' = f' \pm g'$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right\}^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-2) + 2x_2\right] \\ \frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right\}^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-3) + 2\cdot(x_3-1)\right] \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 \partial x_3} \\ \frac{\partial^2 f}{\partial x_3 \partial x_2} & \frac{\partial^2 f}{\partial x_3^2} \end{bmatrix} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$$

$(f \cdot g)' = f'g + fg'$

$$a = \frac{d}{dx_2}\left[\frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right\}^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-2) + 2x_2\right]\right]$$

$$= -\frac{1}{4}\left[(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right]^{-\frac{3}{2}} \cdot \left[2(2-2x_2-3x_3)\cdot(-2) + 2x_2\right]^2 +$$

$$\frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right\}^{-\frac{1}{2}} \cdot 10$$

$$b = \frac{d}{dx_3}\left[\frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right\}^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-2) + 2x_2\right]\right]$$

$$= -\frac{1}{4}\left[(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right]^{-\frac{3}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-3) + 2(x_3-1)\cdot 1\right] \cdot$$

$$\left[2\cdot(2-2x_2-3x_3)\cdot(-2) + 2x_2\right] +$$

$$\frac{1}{2}\left[(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right]^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-3) + 2(x_3-1)\cdot 1\right]$$

$$c = \frac{d}{dx_3} \frac{1}{2} \left\{ (2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2 \right\}^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-3) + 2\cdot(x_3-1) \right]$$

$$= -\frac{1}{4} \left\{ (2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2 \right\}^{-\frac{3}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-3) + 2\cdot(x_3-1) \right]^2 +$$

$$\frac{1}{2} \left\{ (2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2 \right\}^{-\frac{1}{2}} \cdot 20$$

$$H_0 = \begin{bmatrix} a_0 & b_0 \\ b_0 & c_0 \end{bmatrix} = \begin{bmatrix} 0.805 & -5.635 \\ -5.635 & 0.088 \end{bmatrix}$$

$$a_0 = -\frac{1}{4} \cdot (5)^{-\frac{3}{2}} \cdot 64 + \frac{1}{2} \cdot (5)^{-\frac{1}{2}} \cdot 10$$

$$= 0.805$$

$$b_0 = -\frac{1}{4} \cdot (5)^{-\frac{3}{2}} \cdot (-14) \cdot (-8) + \frac{1}{2} \cdot (5)^{-\frac{1}{2}} \cdot (-14)$$

$$= -5.635$$

$$c_0 = -\frac{1}{4} \cdot (5)^{-\frac{3}{2}} \cdot (-14)^2 + \frac{1}{2} \cdot 5^{-\frac{1}{2}} \cdot 20$$

$$= 0.089$$

$$\nabla f = \begin{bmatrix} \dfrac{\partial f}{\partial x_2} \\[2ex] \dfrac{\partial f}{\partial x_3} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right\}^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-2) + 2x_2\right] \\[2ex] \frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right\}^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-3) + 2\cdot(x_3-1)\right] \end{bmatrix}$$

$$\nabla f_0 = \begin{bmatrix} \frac{1}{2} \cdot (5)^{-\frac{1}{2}} \cdot (-8) \\[2ex] \frac{1}{2} \cdot (5)^{-\frac{1}{2}} \cdot (-14) \end{bmatrix} = \begin{bmatrix} -1.789 \\[2ex] -3.130 \end{bmatrix}$$

[II] $\nabla f = 0$

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_2} \\ \frac{\partial f}{\partial x_3} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right]^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-2) + 2x_2\right] \\ \frac{1}{2}\left\{(2-2x_2-3x_3)^2 + x_2^2 + (x_3-1)^2\right]^{-\frac{1}{2}} \cdot \left[2\cdot(2-2x_2-3x_3)\cdot(-3) + 2\cdot(x_3-1)\right] \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\therefore \quad (-4)\cdot(2-2x_2-3x_3) + 2x_2 = 0$$

$$10x_2 + 12x_3 - 8 = 0$$

$$(-6)\cdot(2-2x_2-3x_3) + 2x_3 - 2 = 0$$

$$12x_2 + 20x_3 - 14 = 0$$

$$\begin{cases} 5x_2 + 6x_3 = 4 \\ 6x_2 + 10x_3 = 7 \end{cases}$$

$$x_2 = -0.1429$$

$$x_3 = 0.7857$$

$$x_1 = 1 + 2\cdot 0.1429 - 3\cdot 0.7857$$

$$= -1.0713$$

$$d = 0.2673$$

[III] Check H is p.d. or p.s.d $\Rightarrow$ Import x_1, x_2, x_3 into H

$a = -13.096 \cdot (3.6\times10^{-7}) + 18.708 = 18.708$

$b = -13.096 \cdot (-8\times10^{-4}) \cdot (-6\times10^{-4}) + 1.871 \cdot (-8\times10^{-4}) = -1.503\times10^{-3}$

$c = -13.096 \cdot (6.4\times10^{-7}) + 37.417 = 37.417$

H is p.d. $\Rightarrow$ strictly convex

$|H| > 0$, $f_{xx} > 0$

$\therefore$ f is convex problem and has an unique sol.

First Principal minor :

$|H| = f_{xx}$

Second - Order conditions for local maximum :

$f_{xx} < 0$, $|H| > 0$

Second - Order conditions for local minimum :

$f_{xx} > 0$, $|H| > 0$

Saddle point (Neither local max. or local min.) :

$|H| < 0$

- Convex . Problem

$$\min_x \; f(x)$$

$$\text{s.t.} \quad x \in \mathcal{X}$$

feasible domain

Convex if and only if $f(x)$ is convex func.
$\mathcal{X}$ is convex set
} Exist infinite many sols, equally good

Strictly convex if and only if $f(x)$ is strictly convex func.
$\mathcal{X}$ is convex set
} only 1 sol.

- Newton's Method

$$f(x) \approx f(x_0) + g_0^T (x - x_0) + \frac{1}{2}(x - x_0)^T H_0 (x - x_0)$$

$$\Delta x = x - x_0$$

$$\min_{\Delta x} \quad F(\Delta x) = f(x_0) + g_0^T \Delta x + \frac{1}{2} \Delta x^T H_0 \Delta x$$

$$\nabla_{\Delta x} F = 0 + g_0 + H_0 \cdot \Delta x = 0 \Rightarrow \underline{\Delta x = -H_0^{-1} g_0}$$

Newton's method

$$\because H_0 \cdot \Delta x = -g_0$$

$$\Rightarrow \Delta x = -\alpha \cdot H_0^{-1} \cdot g_0$$

Newton's step

Check :

$$f(x) = f(x_0) + g_0^T(-\alpha H_0^{-1} g_0) + \frac{1}{2}(-\alpha H_0^{-1} g_0)^T H_0 (-\alpha H_0^{-1} g_0)$$

$$= f(x_0) - \alpha g_0^T H^{-1} g_0 + \frac{1}{2}\alpha^2 g_0^T H_0^{-1} g_0$$

$$= f(x_0) - \frac{1}{2} g_0^T H_0^{-1} g_0 \quad (\alpha = 1)$$

$$\frac{1}{2} g_0^T H_0^{-1} g_0 > 0 \ ?$$

$\Rightarrow$ Depends on H_0^{-1}

- x_0 , $k = 0$, $\varepsilon = 10^{-3}$

- Compute g_k , H_k
- α = linesearch (f, g, H)
- $x_{k+1} = x_k - \alpha H_k^{-1} g_k$

learning rate (Newton step)

- $k = k+1$
- if $\|g_k\| < \varepsilon$

  break

  otherwise

  $k = k+1$

※ line search , start with $\underline{\alpha = 1}$

Gradient Descent

- x_0 , $k=0$, $\varepsilon = 10^{-3}$
- compute gradient $g(x_k) := \nabla f(x_k)$
- update $x_{k+1} = x_k - g(x_k)$
- terminate $\|g(x_k)\|^2 < \varepsilon$

Taylor :

$$f(x_{k+1}) = f(x_k) + \nabla f(x_k)^T (x_{k+1} - x_k)$$

$$= f(x_k) + g_k^T(-g_k)$$

$$= f(x_k) - g_k^T g_k$$

$$a^T a = a_1^2 + a_2^2 + \cdots$$

$$= \|a\|^2 \geq 0$$

$$\Rightarrow f(x_{k+1}) - f(x_k) = -g_k^T g_k < 0$$

Issue : $x_{k+1} \cong x_k$ good

$x_{k+1} \ggg x_k$ bad

Solve :

When $x_{k+1} \ggg x_k$: We add a step size (or) learning rate

α is a small number

- update $x_{k+1} = x_k - \alpha g(x_k)$

3. (5 points) Prove that a hyperplane is a convex set. Hint: A hyperplane in $\mathbb{R}^n$ can be expressed as: $\mathbf{a}^T\mathbf{x} = c$ for $\mathbf{x} \in \mathbb{R}^n$, where $\mathbf{a}$ is the normal direction of the hyperplane and c is some constant.

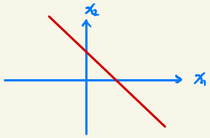
$$S = \{ x \in R^n \mid a^T x = c \;,\; a \in R^n \;,\; c \in R \}$$

① Convex set ?

$a^T x = c$

$\therefore \underbrace{a_1 x_1 + a_2 x_2}_{} + \cdots + a_k x_k = \underbrace{c}_{}$

$\Downarrow$



$\Rightarrow$ All points on the line $\in S$

② $\forall \lambda \in [0, 1]$?

$\bar{x} = \lambda x_1 + (1-\lambda) x_2$

$a^T \bar{x} = c \;\; \in S$

$\therefore$ It's convex set.

4. (15 points) Consider the following illumination problem:

$$\min_{\mathbf{p}} \quad \max_k \{h(\mathbf{a}_k^T \mathbf{p}, I_t)\}$$
$$\text{subject to: } \quad 0 \leq p_i \leq p_{\max},$$

where $\mathbf{p} := [p_1, ..., p_n]^T$ are the power output of the n lamps, $\mathbf{a}_k$ for $k = 1, ...,, m$ are fixed parameters for the m mirrors, I_t the target intensity level. $h(I, I_t)$ is defined as follows:

$$h(I, I_t) = \begin{cases} I_t/I & \text{if } I \leq I_t \\ I/I_t & \text{if } I_t \leq I \end{cases}$$

(a) (5 points) Show that the problem is convex.

(b) (5 points) If we require the overall power output of any of the 10 lamps to be less than p^*, will the problem have a unique solution?

(c) (5 points) If we require no more than 10 lamps to be switched on ($p > 0$), will the problem have a unique solution?

- Convex Problem

(a)

$\min_x \quad f(x)$

st. $\quad x \in \mathcal{X}$ ← feasible domain

Convex if and only if $f(x)$ is convex func.
$\mathcal{X}$ is convex set } Exist infinite many sols. equally good

Strictly convex if and only if $f(x)$ is strictly convex func.
$\mathcal{X}$ is convex set } only 1 sol.

① $\forall x_1, x_2 \in S$

$\lambda \in [0,1]$

$\lambda x_1 + (1-\lambda) x_2 \in S$

if $a_1 p_1 \leq I_t \Rightarrow \frac{I_t}{a_1 p_1} : x_1$ ← scalar

if $a_2 p_2 \leq I_t \Rightarrow \frac{I_t}{a_2 p_2} : x_2$

$\lambda \cdot \frac{I_t}{a_1 p_1} + (1-\lambda) \frac{I_t}{a_2 p_2} \in S$

$\therefore \ x \in \mathcal{X}$ (Convex)

② $f(x)$ is convex func. ?

$\forall x_1, x_2 \in S$

$f(\lambda x_1 + (1-\lambda) x_2) \leq f(x_1) \cdot \lambda + (1-\lambda) f(x_2)$

$\lambda \in [0,1]$

[I] Find H

$f(x) = a^T p = \sum_i a_i p_i$

$\nabla f = a$

$H = I \cdot n$

$\therefore \ f(x)$ is convex func.

4. (15 points) Consider the following illumination problem:

$$\min_{\mathbf{p}} \quad \max_{k}\{h(\mathbf{a}_k^T\mathbf{p}, I_t)\}$$
$$\text{subject to:} \quad 0 \le p_i \le p_{\max},$$

where $\mathbf{p} := [p_1, ..., p_n]^T$ are the power output of the n lamps, $\mathbf{a}_k$ for $k = 1, ..., , m$ are fixed parameters for the m mirrors, I_t the target intensity level. $h(I, I_t)$ is defined as follows:

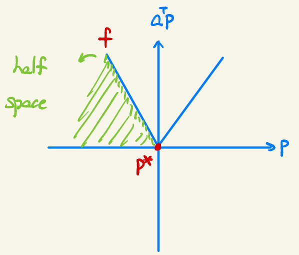
$$h(I, I_t) = \begin{cases} I_t/I & \text{if } I \le I_t \\ I/I_t & \text{if } I_t \le I \end{cases}$$

(a) (5 points) Show that the problem is convex.

(b) (5 points) If we require the overall power output of any of the 10 lamps to be less than p^*, will the problem have a unique solution?

(c) (5 points) If we require no more than 10 lamps to be switched on ($p > 0$), will the problem have a unique solution?

(b)



$\therefore$ No, it will not have unique sol.

All points on $f \in S$

$n \ge 10$

half space $\rightarrow$ $P_1 + \cdots + P_{10} \le p^*$

$\rightarrow$ $P_2 + \cdots + P_{11} \le p^*$

$\Rightarrow \sum_{i=1}^{10} P_i \le p^*$

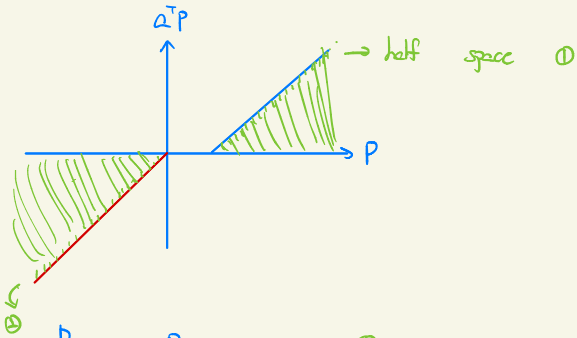
show convex set

$\therefore \sum_{i=k}^{k+9} a_i P_i \le a^T p^*$

(c)

$n \leq 10$ $\sum_{i=1}^{10} a_i p_i$

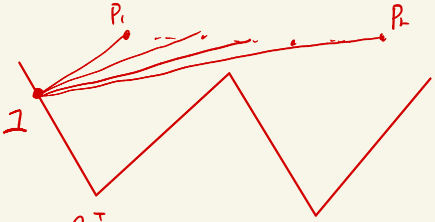
$p > 0$ others $p \leq 0$



$p_1 + \cdots + p_{10} > 0$ — ①

$p_{11} + \cdots p_k \leq 0$ — ②

$\therefore$ Not Convex set.



$\underline{a_i^T p}$

$h(a_i^T p, I_t)$

$$\min_p \underbrace{\max_k \{h(a_k^T p, I_k)\}}_{\text{worst case}}$$

$$\max_k \{h_1, h_2, \cdots, h_m\}$$

① show each convex

② show $h(a_i^T p, I)$ convex w.r.t p

(b) $n \geq 10$

half space $\rightarrow$ $p_1 + \cdots + p_{10} \leq p^*$

$\rightarrow$ $p_2 + \cdots + p_{11} \leq p^*$

show convex set

(c) No convex set

longer

5. (10 points) Let $c(x)$ be the cost of producing x amount of product A and assume that $c(x)$ is differentiable everywhere. Let y be the price set for the product. Assuming that the product is sold out. The total profit is defined as

$$c^*(y) = \max_x \{xy - c(x)\}.$$

Show that $c^*(y)$ is a convex function with respect to y.

① x : production quantity , $C(x)$ production cost

y : price of product

max profit : $\max_x \{x \cdot y - C(x)\}$

convex conjugate ← $C^*(y) = \max_x \{x \cdot y - C(x)\} = \max\{\underline{x_1 y - C(x_1)}, x_2 y - C(x_2), \cdots, x_m y - C(x_m)\}$

$y \in [0, M]$

Convex set ✓

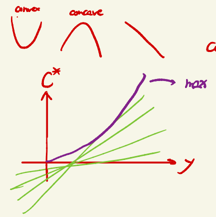
Linear

$\Rightarrow$ P.S.d

$\Downarrow$

convex concave

convex / concave

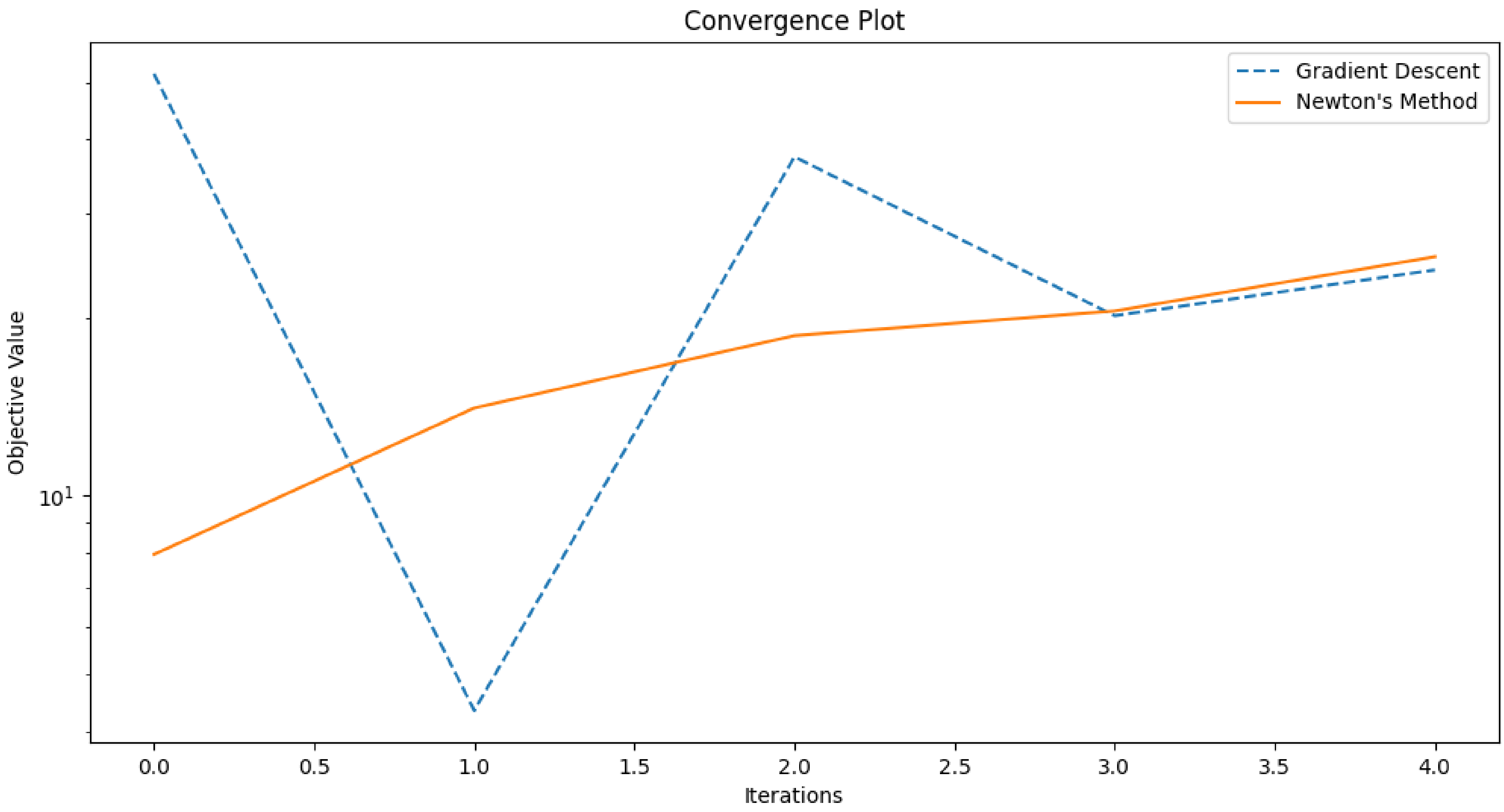


homework 2.py

```
"""
2.
(a) Find the point in the plane x_1 + 2*x_2 + 3*x_3 = 1 in R^3 that is nearest to the point (-1,0,1)^T.
Is this a convex problem?
(b) Implement the gradient descent and Newton's algorithm for solving this problem. Attach a python codes which include:
(1) The initial points tested; (2) corresponding solutions; (3) A log-linear convergence plot
"""
##### To check the commit histories, please visit this site:
##### https://github.com/Hans-Sun825/Hai-Han-Sun-MAE598-Design-Optimization-Homeworks

import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import line_search

### Define the objective function and its gradient
# x[0] = x_2, x[1] = x_3
# x_1 = 1 - 2*x_2 - 3*x_3
def objective(x):
    return ((2 - 2*x[0] -3*x[1])**2 + x[0]**2 + (x[1] - 1)**2)**0.5

"""
>>> a = numpy.array([1, 2, 3, 4], dtype=numpy.float64)
>>> a
array([ 1.,  2.,  3.,  4.])
>>> a.astype(numpy.int64)
array([1, 2, 3, 4])
"""

def gradient(x):
    g = np.array([0.5 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1]-1)**2)**-0.5 * (2 * (2 - 2*x[0] - 3*x[1]) * (-2) + 2*x[0]),
    0.5 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1]-1)**2)**-0.5 * (2 * (2 - 2*x[0] - 3*x[1]) * (-3) + 2 * (x[1]-1))])
    return np.array(g)

def hessian(x):
    h = np.array([[-0.25 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2)**-1.5 * (-4 * (2 - 2*x[0] - 3*x[1] + 2*x[0]))**2 + 5 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2)**-0.5,
        -0.25 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2)**-1.5 * (-6 * (2 - 2*x[0] - 3*x[1]) + 2 * (x[1] - 1) * (-4 * (2 - 2*x[0] - 3*x[1]) + 2*x[0])) + (0.5 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2))**-0.5 * (-6 * (2 - 2*x[0] - 3*x[1]) + 2*(x[1] - 1))],
        [-0.25 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2)**-1.5 * (-6 * (2 - 2*x[0] - 3*x[1]) + 2 * (x[1] - 1) * (-4 * (2 - 2*x[0] - 3*x[1]) + 2*x[0])) + (0.5 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2))**-0.5 * (-6 * (2 - 2*x[0] - 3*x[1]) + 2*(x[1] - 1)),
        -0.25 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2)**-1.5 * (-6 * (2 - 2*x[0] - 3*x[1]) + 2 * (x[1] - 1))**2 + 10 * ((2 - 2*x[0] - 3*x[1])**2 + x[0]**2 + (x[1] - 1)**2)**-0.5]])
    h_inv = np.linalg.inv(np.matrix(h))
    return np.array(h_inv)
#def gradient_zero(x):
```

```python
    return np.transpose([-1.789, -3.130])

#def hessian_zero(x):
    return([0.805, -5.635],
            [-5.635, 0.089])

# Define the necessary parameters
X0 = np.array([[0.0], [0.0]])           # X0 : Initial point
alpha = line_search(f = objective, myfprime = gradient,
                    xk = np.array([1.0, 1.0]), pk = np.array([-0.1, -0.1])) # alpha : learning rate
num_iterations = 5                  # num_iterations : k
epsilon = 0.001                     # Stopping criteria is abs(f(x)) < epsilon.

# Define the gradient descent
def gradient_descent(alpha, num_iterations, epsilon):
    x = np.array([[0.0], [0.0]])         # Initial point
    storage = []                    # To store objective values for convergence plot

    for i in range(num_iterations):
        gradient_x = gradient(x)
        x = x - np.dot(alpha[i], gradient_x)

        if np.linalg.norm(gradient_x)**2 < epsilon:
            break
        else:
            i = i + 1
        storage.append(objective(x))
    return x, storage

# Define the Newton's algorithm
def newton_method(num_iterations, alpha, epsilon):
    x = np.array([[0.0], [0.0]])        # Initial point
    storage = []                    # To store objective values for convergence plot

    for i in range(num_iterations):
        gradient_x = gradient(x)
        hessian_x = hessian(x)
        # Update x using Newton's method
        x = x - alpha[i] * np.dot(hessian_x, gradient_x)

        if np.linalg.norm(gradient_x) < epsilon:
            break
        else:
            i = i + 1
        storage.append(objective(x))
    return x, storage

### Import initial guesses and show the results

# Gradient Descent
for initial_point in X0:
    x_gd, storage_gd = gradient_descent(alpha, num_iterations, epsilon)
    print(f"Gradient Descent: Initial Point = {initial_point}, Solution = {x_gd}")
# Newton's Algorithm
for initial_point in X0:
```

```
    x_newton, stroage_newton = newton_method(num_iterations, alpha, epsilon)
    print(f"Newton's Method: Initial Point = {initial_point}, Solution = {x_newton}")

# Plot convergence
plt.figure(figsize=(12, 6))
plt.plot(range(num_iterations), storage_gd, label="Gradient Descent", linestyle='--')
plt.plot(range(num_iterations), stroage_newton, label="Newton's Method", linestyle='-')
plt.xlabel("Iterations")
plt.ylabel("Objective Value")
plt.yscale("log")
plt.legend()
plt.title("Convergence Plot")
plt.show()

"""
Result:
Gradient Descent: Initial Point = [0.], Solution = [[-3.80036949]
 [-4.51742044]]
Gradient Descent: Initial Point = [0.], Solution = [[-3.80036949]
 [-4.51742044]]
Newton's Method: Initial Point = [0.], Solution = [[-5.43768716]
 [-3.81062982]]
Newton's Method: Initial Point = [0.], Solution = [[-5.43768716]
 [-3.81062982]]
"""

```


Convergence Plot
Gradient Descent
Newton's Method
Objective Value
10^1
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Iterations