The Chinese University of Hong Kong, Shenzhen
SDS · School of Data Science
Andre Milzarek · Fall Semester 2024/25

MDS 6106 — Optimization and Modeling

Exercise Sheet Nr.: Assignment 1

Name: Zijin CAI   Student ID: 224040002

In the creation of this solution sheet, I worked together with:

Name: ______ Student ID: ______

Name: ______ Student ID: ______

Name: ______ Student ID: ______

For correction:

Exercise						$\sum$
Grading						

Assignment A1.1

(a) Decision:

production plan of Fruit Salad A (denoted as *FSA: n*) and Fruit Salad B (denoted as *FSB: m*)

Objective:

maximize the profit (net) given the production plan:

$$n \cdot \text{profit\_FSA}(10) + m \cdot \text{profit\_FSB}(20) \tag{1}$$

Constraints:

- fruit resources:

$$\begin{aligned} n \cdot Mango_A + m \cdot Mango_B &\leq 25 \\ n \cdot P\_apple_A + m \cdot P\_apple_B &\leq 10 \\ n \cdot S\_berry_A + m \cdot S\_berry_B &\leq 120 \end{aligned} \tag{2}$$

- complete use of fruit resources (as warehouse is malfunctioning): the resources constraints are now equality such that $\mathbf{g}(\mathbf{x}) \leq 0$ are now $\mathbf{h}(\mathbf{x}) = 0$
- potential integer constraints: $n, m \in \mathbb{Z}^+$

n, m – the amount of fruit salad A and B produced, respectively

profit function: $f\colon = 10 \cdot n + 20 \cdot m$

the optimization problem can be formulated as:

$$\begin{aligned} \text{max.} \quad & f(n, m) = 10 \cdot n + 20 \cdot m \\ \text{s. t.} \quad & g(Mango_i) : n \cdot Mango_A + m \cdot Mango_B - 25 = 0 \\ & g(P\_apple_i) : n \cdot P\_apple_A + m \cdot P\_apple_B - 10 = 0 \\ & g(S\_berry_i) : n \cdot S\_berry_A + m \cdot S\_berry_B - 120 = 0 \\ \text{where:} \quad & n, m \in \mathbb{Z}^+ \end{aligned} \tag{3}$$

apply the fruit resources usage in this case, we have:

$$\begin{aligned} \text{max.} \quad & f(n, m) = 10 \cdot n + 20 \cdot m \\ \text{s. t.} \quad & n + 2m - 100 = 0 \\ & n + 2m - 80 = 0 \\ & 5n + m - 120 = 0 \\ \text{where:} \quad & n, m \in \mathbb{Z}^+ \end{aligned} \tag{4}$$

(b) the feasible set S in this optimization is defined as:

$$S = \{(n, m) \in \mathbb{Z}^+ \mid n + 2m \leq 100, n + 2m \leq 80, 5n + m \leq 120\} \tag{5}$$

or, more specifically:

$$S = \{(n, m) \in \mathbb{Z}^+ \mid n + 2m = 80, 5n + m = 120\} \tag{6}$$

we can solve for the equations as: $n = \frac{160}{9}, m = \frac{280}{9}$, however, which are not integers.

we then substitute the constraints into the objective function, and derive the first-order conditions as:

$$\begin{aligned} \text{FOC:} \quad \frac{\partial f}{\partial n} &= 10n + 20(120 - 5n) = 0 \\ \frac{\partial f}{\partial m} &= 20m + 10(80 - 2m) = 0 \end{aligned} \tag{7}$$

solve for: $\frac{\partial f}{\partial n} = -90$ (negative) and $\frac{\partial f}{\partial m} = 18$ (positive), which indicates that the objective function is maximized at the boundary of the feasible set. apparently, produce more Fruit Salad B (FSB) will yield higher profit

for instance, the potential optimal solution could be:

$$\begin{aligned} f(n,m) = f(0,40) = 20 \cdot 40 = 800 \\ f(n,m) = f(17,31) = 170 + 620 = 790 \end{aligned} \tag{8}$$

the conclusion on feasible set:

if the constraints are relaxed as inequalities (i.e., $\mathbf{g}(x_i) \leq 0$) with remained fruit:

$$S = \{(n,m) \in \mathbb{Z}^+ \mid n \leq 17, m \leq 31\} \tag{9}$$

for the case of complete use of fruit resources(i.e., no $\mathbf{g}(x_i)$ but $\mathbf{h}(x_i) = 0$), the feasible set will be empty

Assignment A1.2

(a) an (integer) optimization problem, minimizing the differences between each inheritance

Decision:

inheritance partition: $P_{Cass}, \quad P_{Dan}, \quad P_{Bela}$

Objective:

minimize the differences between 3 partitions

$$|P_{Cass} - P_{Dan}| + |P_{Dan} - P_{Bela}| + |P_{Bela} - P_{Cass}| \tag{10}$$

if the equal distribution is achievable, this will be zero

given the birth order, it is ok that: $P_{Bela} \leq P_{Dan} \leq P_{Cass}$

Constraints:

- integer for inheritance items that: $\mathbf{x}_{ij} \in 1, 0, \forall j \in \{inheritances\}$
- dogs not separated: $x_{dogs} = 0.6$ as one share of the inheritance
- the overall inheritance: $P_{Cass} \cup P_{Dan} \cup P_{Bela} = 26.5$,
- while $P_{Cass} \cap P_{Dan} = P_{Cass} \cap P_{Bela} = P_{Dan} \cap P_{Bela} = \emptyset$

define $f \colon = f(\mathbf{P}_i)$ measures the value of the partition $\mathbf{P}_i, i \in \{Cass, Dan, Bela\}$

let $\mathbf{x}_{ij}$ be a binary variable, indicating the inheritance j is allocated to daughter i if $\mathbf{x}_{ij} = 1$

denote w_j as the value weight of the inheritance j such that:

$$f = f(\mathbf{P}_i) = \sum_{i,j} \mathbf{x}_{ij} \cdot w_j, \quad j \in \{inheritances\} \tag{11}$$

then the optimization problem can be formulated as:

$$\begin{aligned} \text{min.}[\text{max.}(&\sum_j x_{Cass,j} \cdot w_j, \quad \sum_j x_{Dan,j} \cdot w_j, \quad \sum_j x_{Bela,j} \cdot w_j) \\ -\text{min.}(&\sum_j x_{Cass,j} \cdot w_j, \quad \sum_j x_{Dan,j} \cdot w_j, \quad \sum_j x_{Bela,j} \cdot w_j)] \end{aligned}$$

$$\begin{aligned} \text{s. t.} \quad & \sum_{i,j} \mathbf{x}_{ij} \cdot w_j = \sum_j w_j = 26.5, \forall i, j \in \{Cass, Dan, Bela\} \& \{inheritance\} \\ & \sum_{i,j} \mathbf{x}_{ij} = 1, \forall i \in \{Cass, Dan, Bela\} \text{for each inheritance -} j \end{aligned} \tag{12}$$

(b) possibility of equal distribution:

denote the equal distribution as $P_{equal} = P_{Cass} = P_{Dan} = P_{Bela} = 26.5/3 \approx 8.83$, which is an infinite decimal

however, the inheritances are valued as finite decimal (i.e, 0.3, 1.2), thus the equal distribution is not achievable

the optimal solution will be the closest to the equal distribution denoted as $\mathbf{P}^* = (P^*_{Cass}, P^*_{Dan}, P^*_{Bela})$

here is the python codes and output:

[2]:
```
import itertools

# iterated weights share of the inheritance item list
inheritance = [8, 0.5, 3.5, 6, 1.2, 1.2, 1.2, 0.3, 0.6, 1, 2, 1]

# the total inheritance value
total_inherit = sum(inheritance)

# the target inheritance plan for each daughter
target_inherit = total_inherit / 3

# indexing the inheritance list
indices = list(range(len(inheritance)))

# initialize the minimum difference and the optimal partition
min_diff = float('inf')
opt_partition = None

# all possible combinations should be generated, considering the inheritance
# the given item list is not too long, the brute-force method is acceptable


def find_partitions():
    global min_diff, opt_partition
    for inheritance1_item in range(1, len(inheritance) - 1):
        for inheritance1_indices in itertools.combinations(indices,␣
↪inheritance1_item):
            inheritance1 = [inheritance[i] for i in inheritance1_indices]
            sum_inheritance1 = sum(inheritance1)
            if abs(sum_inheritance1 - target_inherit) > min_diff:
                continue
            remaining_indices1 = list(set(indices) - set(inheritance1_indices))
            for inheritance2_size in range(1, len(remaining_indices1)):
                for inheritance2_indices in itertools.
↪combinations(remaining_indices1, inheritance2_size):
                    inheritance2 = [inheritance[i] for i in inheritance2_indices]
```

```
                    sum_inheritance2 = sum(inheritance2)
                    inheritance3_indices = list(set(remaining_indices1) -␣
↪set(inheritance2_indices))
                    inheritance3 = [inheritance[i] for i in inheritance3_indices]
                    sum_inheritance3 = sum(inheritance3)
                    sums_inheritance = [sum_inheritance1, sum_inheritance2,␣
↪sum_inheritance3]
                    max_diff = max(sums_inheritance) - min(sums_inheritance)
                    if max_diff < min_diff:
                        min_diff = max_diff
                        opt_partition = (inheritance1, inheritance2,␣
↪inheritance3)
                        if min_diff <= 0.1:
                            return


# output the result
find_partitions()

if opt_partition:
    inheritance1, inheritance2, inheritance3 = opt_partition
    print("the first share of the inheritance is", inheritance1, "total value␣
↪as", sum(inheritance1))
    print("the second share of the inheritance is", inheritance2, "total value␣
↪as", sum(inheritance2))
    print("the third share of the inheritance is", inheritance3, "total value␣
↪as", sum(inheritance3))
    print("maximum difference between the inheritance partitions", min_diff)
else:
    print("no optimal partition found")
```

```
the first share of the inheritance is [8, 0.5, 0.3] total value as 8.8
the second share of the inheritance is [6, 1.2, 0.6, 1] total value as 8.8
the third share of the inheritance is [3.5, 1.2, 1.2, 2, 1] total value as 8.9
maximum difference between the inheritance partitions 0.09999999999999964
```

thus the optimal solution to this inheritance problem will be that: $\mathbf{P}^* = (P^*_{Cass}, P^*_{Dan}, P^*_{Bela}) = (8.9, 8.8, 8.8)$

in words, Cassandra should receive total inheritances valued 8.9 including Yuan dynasty Chinese vase, 2 diamonds, sailing boat and ancient sculpture/Harley-Davidson Motorbike,

while Daniela and Bela should receive total inheritances valued 8.8, either the one including sketch by Mondrian, bust of Alexander the Great, and Louis XV sofa

or the remaining inheritances including the 1 diamond, the dogs, 911 Porsche and ancient sculpture/Harley-Davidson Motorbike, which is left

Assignment A1.3

(A1)

$$\mathbf{A_1} = \begin{bmatrix} 1 & 1 & 2 \\ 1 & 1 & 2 \\ 2 & 2 & 4 \end{bmatrix} \tag{13}$$

as $\mathbf{A_1}$ is symmetric, the eigenvalues can be decomposed for classification

$$det(\mathbf{A_1} - \lambda \cdot \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}) = -\lambda^3 + 6\lambda^2 = 0 \tag{14}$$

solve for the eigenvalues: $\lambda_1 = 0, \lambda_2 = 0, \lambda_3 = 6$

the eigenvalues of $\mathbf{A_1} = (0, 0, 6)$ are all real and non-negative, which indicates that $\mathbf{A_1}$ is positive semi-definite

(A2)

$$\mathbf{A_2} = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 2 & 0 \\ -1 & 0 & 2 \end{bmatrix} \tag{15}$$

$$\mathbf{X^T A_2 X} = (x_1, x_2, x_3) \begin{bmatrix} 0 & 0 & 1 \\ -1 & 2 & 0 \\ -1 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 2x_2^2 + 2x_3^2 - x_1 x_2 \tag{16}$$

$2x_2^2 + 2x_3^2$ is non-negative, while $-x_1 x_2$ can be either positive or negative, thus $\mathbf{A_2}$ is indefinite

$$det(\mathbf{A_2} - \lambda \mathbf{I}) = -\lambda^3 + 4\lambda^2 - 5\lambda + 2 = 0 \tag{17}$$

solve for the eigenvalues: $\lambda_1 = 1, \lambda_2 = 1, \lambda_3 = 2$

the eigenvalues of $\mathbf{A_2} = (1, 1, 2)$

(A3)

$$\mathbf{A_3} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ -1 & 1 & 0 \end{bmatrix} \tag{18}$$

$$\mathbf{X^T A_3 X} = (x_1, x_2, x_3) \begin{bmatrix} 1 & 0 & 1 \\ 0 & -1 & -1 \\ -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = x_1^2 - x_2^2 \tag{19}$$

$x_1^2 - x_2^2$ is indefinite, thus $\mathbf{A_3}$ is indefinite

$$det(\mathbf{A_3} - \lambda \mathbf{I}) = -\lambda^3 - \lambda = 0 \tag{20}$$

solve for the eigenvalues: $\lambda_1 = 0, \lambda_2 = i, \lambda_3 = -i$

there is only one eigenvalue for $\mathbf{A_3}$ that $\lambda = 0$

Assignment A1.4

(a) $\mathbf{y}^i \in \mathbb{R}^2 \quad \forall \quad i = 1, \ldots, k$ be given with k different

given set of points $(i, \mathbf{y}^i) \in \{(1, \mathbf{y}^1), (2, \mathbf{y}^2), \ldots, (k, \mathbf{y}^k)\}$, the convex hull is the smallest convex polygon that contains all the points

let $\mathbf{H} = \{h_1, h_2, \ldots, h_k\}$ be the vertices of the convex hull in counter-clockwise order

denote each edge of the convex hull as: $\mathbf{V}_i \forall i \in \{1, \ldots, k\}$

and for each edge $\mathbf{V}_i$, compute the perpendicular direction vector $\mathbf{H}_i$

then rotate calipers around the convex hull, maintaining contacts with the convex hull at all time, with the supporting indices $\mathbf{I} = \{b, r, t, l\}$ (as the bottom, right, top, left of the convex hull, respectively)

let function $f\colon =$ looping selection of the successor vertices i.e., $\mathbf{V}_i^{'} = f(\mathbf{V}_i)$

perpendicular operation :

$$perp(i, y^i) = (y^i, -i) \tag{21}$$

supporting vertices (multiset):

$$\mathbf{S} = \{V_{i0}, V_{i1}, V_{i2}, V_{i3}\} \tag{22}$$

supporting indices:

$$\mathbf{I} = \{i_0, i_1, i_2, i_3\} \tag{23}$$

denote the initial axis direction:

$$\mathbf{U}_0 = \frac{(\mathbf{V}_0 - \mathbf{V}_{n-1})}{|\mathbf{V}_0 - \mathbf{V}_{n-1}|} \quad \mathbf{U}_1 = -perp(\mathbf{U}_0) \tag{24}$$

the width and the height are denoted as $w \& h$ respectively:

$$w = \mathbf{U}_0(\mathbf{V}_r - \mathbf{V}_l) \quad h = \mathbf{U}_1(\mathbf{V}_t - \mathbf{V}_b) \tag{25}$$

let $\mathbf{W}_0 = \mathbf{V}_{ji} - \mathbf{V}_{j0} \quad \& \quad \mathbf{W}_1 = -perp(\mathbf{W}_0)$, the unit-length axis is $\mathbf{U}_i = \mathbf{W}_i / |\mathbf{W}_0|$

and the area is then intuitively: $a = w \cdot h$:

$$a = w \cdot h = \frac{(\mathbf{W}_0 \cdot (\mathbf{V}_r - \mathbf{V}_l)) \cdot (\mathbf{W}_1 \cdot (\mathbf{V}_t - \mathbf{V}_b))}{|\mathbf{W}_0|^2} \tag{26}$$

Decision:

determine the iterated supporting indices $\mathbf{I}' = \{b', r', t', l'\}$ that generated from the initial $\mathbf{I} = \{b, r, t, l\}$

Objective:

minimize the area of rectangle such that $a = w \cdot h$, containing the convex hull of all points

Constraints:

maintaining contacts with the hull at all points

the optimization problem can be formulated as:

$$\text{min.} \quad a = w \cdot h = \frac{(\mathbf{W}_0 \cdot (\mathbf{V}_r - \mathbf{V}_l)) \cdot (\mathbf{W}_1 \cdot (\mathbf{V}_t - \mathbf{V}_b))}{|\mathbf{W}_0|^2}$$

$$\text{s. t.} \quad \mathbf{U_0'} = f(\mathbf{U_0}) = \frac{(\mathbf{V}_{i0}' - \mathbf{V}_{i0-1}')}{|\mathbf{V}_{io} - \mathbf{V}_{i0-1}|} \quad \forall i \in \{1, \ldots, k\} \quad (27)$$

it's a Constraint, Non-Linear, Continuous optimization problem

(b) the rectangle with the minimum perimeter problem:

denote the perimeter as $p = 2 \cdot (w + h)$, which is the new objective, with the same decision and constraints

$$\text{min.} \quad a = 2 \cdot (w + h) = \frac{2 \cdot (\mathbf{W}_0 \cdot (\mathbf{V}_r - \mathbf{V}_l)) + 2 \cdot (\mathbf{W}_1 \cdot (\mathbf{V}_t - \mathbf{V}_b))}{|\mathbf{W}_0|^2}$$

$$\text{s. t.} \quad \mathbf{U_0'} = f(\mathbf{U_0}) = \frac{(\mathbf{V}_{i0}' - \mathbf{V}_{i0-1}')}{|\mathbf{V}_{io} - \mathbf{V}_{i0-1}|} \quad \forall i \in \{1, \ldots, k\} \quad (28)$$

by simplify:

$$\begin{aligned} &\text{min.} \quad (\mathbf{V}_{ji} - \mathbf{V}_{j0}) \cdot (\mathbf{V}_r - \mathbf{V}_l) - prep((\mathbf{V}_{ji} - \mathbf{V}_{j0})) \cdot (\mathbf{V}_t - \mathbf{V}_b) \\ &\text{min.} \quad \mathrm{V}_{ji} \cdot (\mathbf{V}_r - \mathbf{V}_l) - prep(\mathbf{V}_{ji}) \cdot (\mathbf{V}_t - \mathbf{V}_b) \end{aligned} \tag{29}$$

while we simplify the minimum area:

$$\text{max.} \quad \mathbf{V}_{ji} \cdot (\mathbf{V}_r - \mathbf{V}_l) \cdot prep(\mathbf{V}_{ji}) \cdot (\mathbf{V}_t - \mathbf{V}_b) \tag{30}$$

these indicate that the minimum area and the minimum perimeter are paradoxical to each other that, the rectangle with minimum area will yield a larger (which is definitely not the minimal) perimeter, and vice versa

Assignment A1.5

(a) the gradient vector:

$$\nabla f = \left(\frac{\partial f}{\partial x_1} f(x), \frac{\partial f}{\partial x_2} f(x), \ldots, \frac{\partial f}{\partial x_n} f(x) \right)^T \tag{31}$$

the Hessian matrix:

$$\nabla^2 \mathbf{f}(\mathbf{x}) = \mathbf{H_f}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1 \partial x_1}(\mathbf{x}) & \frac{\partial f}{\partial x_1 \partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f}{\partial x_1 \partial x_n}(\mathbf{x}) \\ \frac{\partial f}{\partial x_2 \partial x_1}(\mathbf{x}) & \frac{\partial f}{\partial x_2 \partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f}{\partial x_2 \partial x_n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial x_n \partial x_1}(\mathbf{x}) & \frac{\partial f}{\partial x_n \partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f}{\partial x_n \partial x_n(\mathbf{x})} \end{bmatrix} \tag{32}$$

function 1: $f_1(x) = (x_1^2 - x_2)^2 + (x_1 - x_2^2)^2 - (x_1 - 1)^3 + (x_2 - 1)^3$

$$\begin{gathered} \nabla f_1(x) = \begin{bmatrix} 4x_1^3 - 3x_1^2 + 8x_1 - 4x_1x_2 - 2x_2^2 - 3 \\ 4x_2^3 + 3x_2^2 - 4x_1x_x - 2x_1^2 + 3 \end{bmatrix} \\ \nabla^2 f_1(\mathbf{x}) = \begin{bmatrix} 12x_1^2 - 6x_1 + 8 - 4x_2 & -4x_1 - 4x_2 \\ -4x_1 - 4x_2 & 12x_2^2 + 6x_2 - 4x_1 - 4 \end{bmatrix} \end{gathered} \tag{33}$$

function 2: $f_2(x) = \cos(x_1)\sin(x_2) - \frac{x_1}{1+x_2^2}$

$$\begin{gathered} \nabla f_2(x) = \begin{bmatrix} -\sin(x_1)\sin(x_2) - \frac{1}{1+x_2^2} \\ \cos(x_1)\cos(x_2) + \frac{2x_1x_2}{(1+x_2^2)^2} \end{bmatrix} \\ \nabla^2 f_2(\mathbf{x}) = \begin{bmatrix} -\cos(x_1)\sin(x_2) & -\sin(x_1)\cos(x_2) + \frac{2x_2}{(1+x_2^2)^2} \\ -\sin(x_1)\cos(x_2) + \frac{2x_2}{(1+x_2^2)^2} & -\cos(x_1)\sin(x_2) + \frac{2x_1(1-3x_2^2)}{(1+x_2^2)^3} \end{bmatrix} \end{gathered} \tag{34}$$

function 3: $f_3(x) = x_1^4 + 2(x_1 - x_2)x_1^2 + 4x_2^2$

$$\begin{gathered} \nabla f_3(x) = \begin{bmatrix} 4x_1^3 + 6x_1^2 - 4x_1x_2 \\ 8x_2 - 2x_1^2 \end{bmatrix} \\ \nabla^2 f_3(\mathbf{x}) = \begin{bmatrix} 12x_1^2 + 12x_1 - 4x_2 & -4x_1 \\ -4x_1 & 8 \end{bmatrix} \end{gathered} \tag{35}$$

(b)

$$
\begin{aligned}
&\nabla f_3(x^*) = 0 \quad \text{s.t.} \quad \nabla f_3(x) = \begin{bmatrix} 4x_1^3 + 6x_1^2 - 4x_1x_2 \\ 8x_2 - 2x_1^2 \end{bmatrix} = 0 \\
&\text{solve for:} \quad x_1 = 0, x_2 = 0 \quad \text{or} \quad x_1 = -2, x_2 = 1 \\
&\text{s.t.} \quad x^* \in \{(0,0), (-2,1)\}
\end{aligned} \tag{36}
$$

if $x^* = (0,0)$:

$$
\begin{aligned}
&\nabla^2 f_3(x^*) = \begin{bmatrix} 4x_1^3 + 6x_1^2 - 4x_1x_2 \\ 8x_2 - 2x_1^2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 8 \end{bmatrix} \\
&\text{which is symmetric, and the eigenvalues are:} \quad \lambda_1 = 0, \lambda_2 = 8
\end{aligned} \tag{37}
$$

thus in this case, the Hessian matrix is positive semi-definite

if $x^* = (-2,1)$:

$$
\begin{aligned}
&\nabla^2 f_3(x^*) = \begin{bmatrix} 4x_1^3 + 6x_1^2 - 4x_1x_2 \\ 8x_2 - 2x_1^2 \end{bmatrix} = \begin{bmatrix} 20 & 8 \\ 8 & 8 \end{bmatrix} \\
&\text{which is symmetric, and the eigenvalues are:} \quad \lambda_1 = 24, \lambda_2 = 4
\end{aligned} \tag{38}
$$

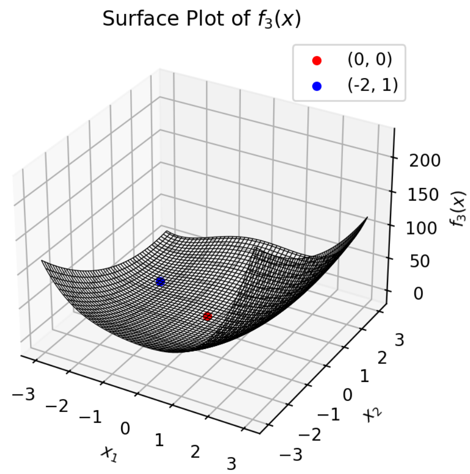
thus the Hessian matrix is positive definite in this case

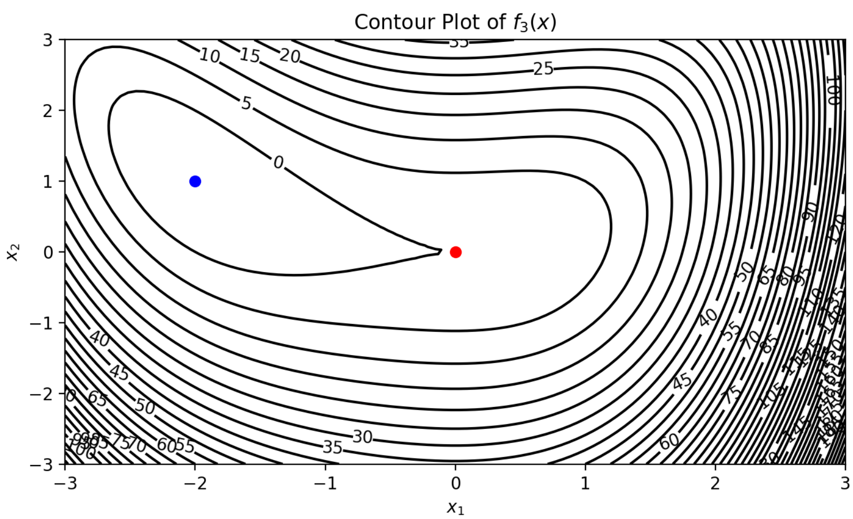
overall, the definiteness of the Hessian $\nabla^2 f_3(x^*)$ is PSD

(c) python code for the surface plot and the contour plot of the function $f_3(x)$:

[3]:
```python
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D


# define the function
def f3(x1, x2):
    return x1 ** 4 + 2 * (x1 - x2) * x1 ** 2 + 4 * x2 ** 2


# create data grid for x1 and x2
x1 = np.linspace(-3, 3, 100)
x2 = np.linspace(-3, 3, 100)
X1, X2 = np.meshgrid(x1, x2)

# calculate the function values
Z = f3(X1, X2)

# create plots
fig = plt.figure(figsize=(14, 6))
```

```python
# create the 3D surface plot
ax1 = fig.add_subplot(211, projection='3d')
ax1.plot_wireframe(X1, X2, Z, color='k', linewidth=0.5)
ax1.set_title('Surface Plot of $f_3(x)$')
ax1.set_xlabel('$x_1$')
ax1.set_ylabel('$x_2$')
ax1.set_zlabel('$f_3(x)$')
ax1.scatter(0, 0, f3(0, 0), color='r', label='(0, 0)')
ax1.scatter(-2, 1, f3(-2, 1), color='b', label='(-2, 1)')
ax1.legend()

# create the contour plot
ax2 = fig.add_subplot(212)
contour = ax2.contour(X1, X2, Z, levels=50, colors='k')
ax2.set_title('Contour Plot of $f_3(x)$')
ax2.set_xlabel('$x_1$')
ax2.set_ylabel('$x_2$')
ax2.scatter(0, 0, color='r')
ax2.scatter(-2, 1, color='b')
ax2.clabel(contour, inline=True, fontsize=10)  #

plt.show()
```


Surface Plot of $f_3(x)$
(0, 0)
(-2, 1)
$f_3(x)$
x_1
x_2


Contour Plot of $f_3(x)$
x_1
x_2