Numerical Optimization - Assignment 5

David Alvear(187594) - Nouf Farhoud(189731)

In [ ]:
```python
import numpy as np
import numpy.linalg as la
import jax
from scipy.optimize import root
import jax.numpy as jnp
import matplotlib.pyplot as plt
```

1. Exercise in R^3

1. Exercise in R^3. Write and solve the first order optimality conditions of the following problem.

$$\begin{array}{ll} \text{minimize} & x_1x_2 + x_2x_3 \\ \text{subject to} & x_1^2 + x_2^2 - 2 = 0 \\ & x_1^2 + x_3^2 - 2 = 0, \end{array}$$

For the solution of the nonliear system, you may use the code you wrote in Exercise 1 or, alternatively, you may use the `root` routine from the `scipy.optimize` library.

$$L(x_1, x_2, x_3, \lambda_1, \lambda_2) = x_1x_2 + x_2x_3 + \lambda_1(x_1^2 + x_2^2 - 2) + \lambda_2(x_1 + x_3^2 - 2)$$
$$\frac{\partial L}{\partial x_1} = x_2 + 2\lambda_1 x_1 + \lambda_2 = 0$$
$$\frac{\partial L}{\partial x_2} = x_1 + x_3 + 2\lambda_1 x_2 = 0$$
$$\frac{\partial L}{\partial x_3} = x_2 + 2\lambda_2 x_3 = 0$$
$$\frac{\partial L}{\partial \lambda_1} = x_1^2 + x_2^2 - 2 = 0$$
$$\frac{\partial L}{\partial \lambda_2} = x_1 + x_3^2 - 2 = 0$$

In [ ]:
```python
from scipy.optimize import root

def objective(vars):
    x1, x2, x3, lambda1, lambda2 = vars
    eq1 = x2 + 2 * lambda1 * x1 + 2 * lambda2 * x1
    eq2 = x1 + x3 + 2 * lambda1 * x2
    eq3 = x2 + 2 * lambda2 * x3
    eq4 = x1**2 + x2**2 - 2
    eq5 = x1**2 + x3**2 - 2
    return [eq1, eq2, eq3, eq4, eq5]
```

```python
initial_guess = [0.0, 0.0, 0.0, 0.0, 0.0]
result = root(objective, initial_guess)

x1, x2, x3, lambda1, lambda2 = result.x
print(f"Optimal values: x1={x1}, x2={x2}, x3={x3}, lambda1={lambda1}, lambda
```

```
Optimal values: x1=1.3065629648849764, x2=-0.5411961001570583, x3=-0.5411961
001325913, lambda1=0.7071067812139786, lambda2=-0.5000000000261735
```

2. Significance of the Lagrange multipliers

2. Significance of the Lagrange multipliers. Consider the equality-constrained problem:

$$
\begin{array}{ll}
\text{minimize} & f(x) \\
\text{subject to} & h_i(x) = 0, \quad i = 1, \dots, p
\end{array}
$$

Show that the value of a Lagrange multiplier at optimality ν_i^* may be interpreted as the sensitivity of the optimal solution f^* with respect to the right hand side of the i-th constraint. In other words, if the i-the constraint is changed to $h_i(x) = \epsilon$, then: $\Delta f^* \approx -\nu_i^* \epsilon$.

We have the following Problem:

Minimize $f(x)$
Subject to $h_i(x) = 0, i = 0, \dots, P$

If we have a variation in the constraints such that:
$h_i(x) = \epsilon,\ i = \{1, \dots, P\}$

It implies that:
$h_i(x^* + \Delta x) \approx h_i(x^*) + \nabla h_i(x^*)^T \Delta x = -b + a^T \Delta x = 0$
where $b = \epsilon$ and $a = \nabla h_i(x^*)$.

Then, $\nabla h_i(x^*)^T \Delta x = \epsilon$. The implication of this is the shifting of the line when Δx makes $a^T \Delta x = 0$.

Now, generalizing for the P constraints we define the Jacobian of the constraints as $A(x) \in \mathbb{R}^{P \times n}$.

The optimality condition says $\nabla f(x^*) + A(x^*)^T v^* = 0$
$f(x + \Delta x) = f(x) + \nabla f(x) \Delta x$
$f(x + \Delta x) - f(x) = -(\nabla f(x)^T) \Delta x = -(A(x^*)^T \Delta x)^T v$

$A(x)$ is the Jacobian of the constraints composed by $\nabla h_i(x)$ for $i = 1, \dots, P$.

Then for the i^{th} constraint we have that $\nabla h_i(x) \Delta x = \epsilon$ and for $i \neq j$ $\nabla h_j(x) \Delta x = 0$ to then:

$\Delta f = -(A(x^*)^T \Delta x)^T v = -(\nabla h_i(x) \Delta x). \lambda_i$

and:
$\Delta f = -\epsilon. \lambda_i$
$\rightarrow$ This implies that $\dfrac{\partial f}{\partial \epsilon} = -\lambda_i$

3. Necessary but not sufficient conditions.

3. Necessary but not sufficient conditions. Verify that the first order optimality conditions of the following problem are satisfied at the points $(-2, 2)$ and $(2, -2)$ but yet neither is a solution.

$$\begin{array}{ll} \text{minimize} & x_1 - x_2 \\ \text{subject to} & x_1 x_2 + 4 = 0 \end{array}$$

$f(x_1, x_2) = x_1 - x_2$

$g(x_1, x_2) = x_1 x_2 + 4 = 0$

① check if the points satisfy the constraint:

$*\ (-2, 2)$

$g(-2,2) = -2(2) + 4 \stackrel{?}{=} 0$

$0 = 0$ ✓

$*\ (2, -2)$

$g(2,-2) = (2)(-2) + 4 \stackrel{?}{=} 0$

$0 = 0$ ✓

Both satisfy the constraint

② 1st order derivation

$L(x_1, x_2, \lambda) = f(x_1, x_2) - \lambda g(x_1, x_2)$

$= x_1 - x_2 - \lambda(x_1 x_2 + 4)$

$\frac{\partial L}{\partial x_1} = 1 - \lambda x_2 = 0$

$\frac{\partial L}{\partial x_2} = -1 - \lambda x_1 = 0$

$\frac{\partial L}{\partial \lambda} = x_1 x_2 + 4 = 0$

③ • $\lambda = \frac{1}{x_2}$ $\qquad \lambda = \frac{1}{2}$

• $-1 - \frac{x_1}{x_2} = 0$

$x_1 = -x_2$ $\qquad x_1 = -2$ $\qquad x_1 = 2$

• $-x_2(x_2) + 4 = 0$

$x_2 = +2$ $\qquad x_2 = -2$

$(-2, 2)$ $\qquad (2, -2)$

First order optimality is satisfied.

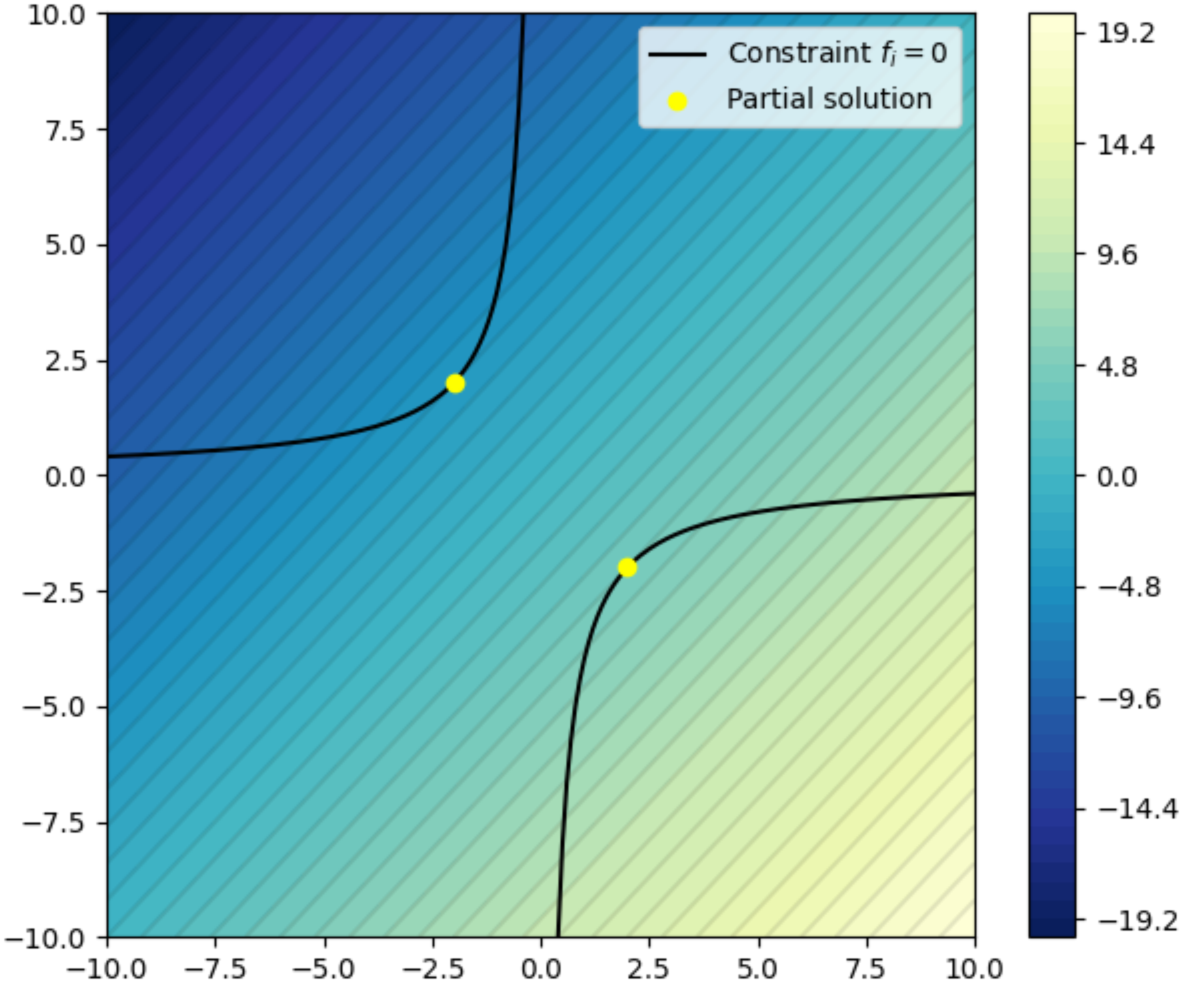
Both $(2,-2)$ and $(-2,2)$ satisfy constraint and the first order optimality. Therefore, they are

Solutions.

In [ ]:
```
from scipy.optimize import fsolve

def eqs2(x):
    x1 = x[0]
    x2 = x[1]
    lam = x[2]

    return(np.array([1 + lam*x2,
                     -1 + lam*x1,
                     x1*x2+4]))

fsolve(eqs2, (0,0,0))
```

Out[ ]:
```
array([-2. ,  2. , -0.5])
```

In [ ]:
```
plt.figure(figsize=(7, 6))
xx, yy = np.meshgrid(np.linspace(-10, 10, 100), np.linspace(-10, 10, 100))
obj = xx - yy

_x = np.linspace(-100, -1E-4, 1000)
__x = np.linspace(1E-4, 100,  1000)

plt.plot(_x, -4/_x, 'k', label='Constraint $f_i = 0$')
plt.plot(__x, -4/__x, 'k')
im = plt.contourf(xx, yy, obj, 50, cmap='YlGnBu_r')
plt.contour(xx, yy, obj, 50, colors='black', linestyles='solid', alpha=0.1)
plt.scatter(-2, 2, zorder=10, c='yellow', label='Partial solution')
plt.scatter(2, -2, zorder=10, c='yellow')
plt.xlim(-10, 10)
plt.ylim(-10, 10)
plt.legend(loc=1)
plt.colorbar(im);
```



4. Healthy snack

4. Healthy snack. Consider the problem of purchasing afternoon snacks. Health conscious buyers need at least 6 total ounces of chocolate, 10 ounces of sugar, and 8 ounces of cream cheese. There are 2 choices of snacks: brownies and cheescakes whose ingredients are listed below. Brownies cost 50 cents and mini-cheesecakes cost 80 cents.

	Chocolate	Sugar	Cream Cheese
Brownie	3	2	2
Cheesecake	0	4	5

- Formulate the minimum-cost healthy purchase snack problem as a linear optimization problem, assuming a friendly bakery that allows fractional purchases, and solve it using the `linprog` routine from the `scipy.optimize` library.
- What are the values of the Lagrange multipliers? What is their physical interpretation in this problem? Comment on their values. (*Note* the values of the multipliers may be found in the `ineqlin` field of the result returned by `linprog`)

Problem formulation:
$x_1 \to$ Brownies
$x_2 \to$ Cheesecakes

Minimize $0.50x_1 + 0.80x_2$
Subject to

$$3x_1 \geq 6 \quad \to \quad -3x_1 + 6 \leq 0$$
$$2x_1 + 4x_2 \geq 10 \quad \to \quad -2x_1 - 4x_2 + 10 \leq 0$$
$$2x_1 + 5x_2 \geq 8 \quad \to \quad -2x_1 - 5x_2 + 8 \leq 0$$

$$L(x, v) = 0.50x_1 + 0.80x_2 + [v_1, v_2, v_3] \begin{bmatrix} -3x_1 + 6 \\ -2x_1 - 4x_2 + 10 \\ -2x_1 - 5x_2 + 8 \end{bmatrix}$$

$$\nabla_x L(x, v) = \begin{bmatrix} 0.50 - 3v_1 - 2v_2 - 2v_3 \\ 0.80 - 4v_2 - 5v_3 \end{bmatrix} = \mathbf{0}$$

$$\nabla_v L(x, v) = \begin{bmatrix} -3x_1 + 6 \\ -2x_1 - 4x_2 + 10 \\ -2x_1 - 5x_2 + 8 \end{bmatrix} = \mathbf{0}$$

In [ ]:
```python
from scipy.optimize import linprog
import numpy as np

# Define the coefficients of the objectivefunction
c = np.array([0.50, 0.80])

# Define the coefficients of the constraints
A = np.array([[-3, 0],  # Chocolate
              [-2, -4],  # Sugar
              [-2, -5]]) # Cream Cheese
b = np.array([-6, -10, -8]) # At least ...

# Bounds
x_bounds = [
    (0, None),
    (0, None)
]

# Solve the linear programming problem
result = linprog(c, A_ub=A, b_ub=b, bounds=x_bounds, method='highs')

# Print the result
print(result)
```

```
        message: Optimization terminated successfully. (HiGHS Status 7: Opti
mal)
        success: True
         status: 0
            fun: 2.2
              x: [ 2.000e+00  1.500e+00]
            nit: 1
          lower:  residual: [ 2.000e+00  1.500e+00]
                 marginals: [ 0.000e+00  0.000e+00]
          upper:  residual: [       inf        inf]
                 marginals: [ 0.000e+00  0.000e+00]
          eqlin:  residual: []
                 marginals: []
        ineqlin:  residual: [ 0.000e+00  0.000e+00  3.500e+00]
                 marginals: [-3.333e-02 -2.000e-01 -0.000e+00]
 mip_node_count: 0
 mip_dual_bound: 0.0
        mip_gap: 0.0
```

In [ ]:

```python
# Check the multipliers
print(result.ineqlin)
```

```
 residual: [ 0.000e+00  0.000e+00  3.500e+00]
marginals: [-3.333e-02 -2.000e-01 -0.000e+00]
```

Based on the results the objective function found the minimum price is 2.2 dollars buying 2 browines and 1.5 cheesecakes to fulfill the nutritional requirements.

We can see that the lagrange multipliers in the optimization problem have the values of:

- $\nu_1 = 0.0333$
- $\nu_2 = 0.2000$
- $\nu_3 = 0.0000$

The units of the lagrange multipliers are $\frac{dollars}{ounces}$. This multipliers measures the sensitivity of the function to the constraint. In fact, we can see that the multiplier ν_3 is zero. Which implies that the constraint is not used in the optimization.

In [ ]:

```python
# Check the lagrange optimal conditions
x1 = 2.0
x2 = 1.5
nu1 = 1/30
nu2 = 0.2000
nu3 = 0.0

# Gradient of the lagrange respect to x
print("Gradient of the lagrange respect to x")
print(0.50+nu1*(-3)+nu2*(-2)+nu3*(-2) == 0)
print(0.80+nu2*(-4)+nu3*(-5) == 0)

# Gradient of the lagrange respect to nu
print("Gradient of the lagrange respect to nu")
print(-3*x1 + 6 <= 0)
```

```python
print(-2*x1 - 4*x2 + 10 <= 0)
print(-2*x1 - 5*x2 + 8 <= 0)
```

```
Gradient of the lagrange respect to x
True
True
Gradient of the lagrange respect to nu
True
True
True
```