

Please use this link to open the GitHub repository.

<https://github.com/rpatel71/Design-Optimization>

## Solve an Optimization Problem

```
# importing libraries needed
from scipy.optimize import minimize
from scipy.optimize import Bounds
from scipy.optimize import LinearConstraint
```

### Defining the given function

```
# Given function
given_fun = lambda x: ((x[0]-x[1])**2 + (x[1]+x[2]-2)**2 + (x[3]-1)**2 + (x[4]-1)**2)
```

### Define Linear Constraints and Bounds

```
# defining linear constraint
lin_const = ({'type': 'eq', 'fun': lambda x: x[0] + 3*x[1]}, {'type': 'eq', 'fun': lambda x: x[2] - 2}, {'type': 'ineq', 'fun': lambda x: x[3] - 1}, {'type': 'ineq', 'fun': lambda x: x[4] - 1})
# defining the bounds of the variables
bounds = ([-10, 10], [-10, 10], [-10, 10], [-10, 10], [-10, 10])
```

### Taking Initial Guess

```
x0 = [5, 4, 6, 7, 2]
```

### Finding the result

```
optimum_res = minimize(given_fun, x0, method='SLSQP', bounds=bounds, constraints=lin_const)
print(f"minimum values for the initial guess of {x0} are {optimum_res}")

minimum values for the initial guess of [5, 4, 6, 7, 2] are      fun: 4.093023635545365
                jac: array([-2.04545265, -0.1874249 , -2.23287761, -2.23303282, -1.48863679])
                message: 'Optimization terminated successfully'
               nfev: 43
                 nit: 7
                njev: 7
              status: 0
             success: True
                x: array([-0.76704475,  0.25568158,  0.6278796 , -0.11651643,  0.25568158])
```

### Changing initial guess and evaluating the results

```
# taking a different guess
x1 = [-9, 5, 4, 8, -5]
```

```

x2 = [-2, -1, 0, 4, 3]
x3 = [-8, 5, 3, -6, -7]
x4 = [1, 3, 6, -1, -3]
x5 = [9, 6, -4, -2, -1]

# evaluating the results
optimum_res_1 = minimize(given_fun, x1, method='SLSQP', bounds=bounds, constraints=lin_const)
optimum_res_2 = minimize(given_fun, x2, method='SLSQP', bounds=bounds, constraints=lin_const)
optimum_res_3 = minimize(given_fun, x3, method='SLSQP', bounds=bounds, constraints=lin_const)
optimum_res_4 = minimize(given_fun, x4, method='SLSQP', bounds=bounds, constraints=lin_const)
optimum_res_5 = minimize(given_fun, x5, method='SLSQP', bounds=bounds, constraints=lin_const)

# printing the results onto terminal
print(f"minimum values for the initial guess of {x0} are {optimum_res_1}")
print(f"minimum values for the initial guess of {x0} are {optimum_res_2}")
print(f"minimum values for the initial guess of {x0} are {optimum_res_3}")
print(f"minimum values for the initial guess of {x0} are {optimum_res_4}")
print(f"minimum values for the initial guess of {x0} are {optimum_res_5}")

minimum values for the initial guess of [5, 4, 6, 7, 2] are      fun: 4.093023255883389
    jac: array([-2.04651397, -0.18605489, -2.23256886, -2.23254561, -1.48837149])
    message: 'Optimization terminated successfully'
    nfev: 42
    nit: 7
    njev: 7
    status: 0
    success: True
        x: array([-0.76744275,  0.25581425,  0.62790131, -0.11627281,  0.25581425])
minimum values for the initial guess of [5, 4, 6, 7, 2] are      fun: 4.093023358204564
    jac: array([-2.04616207, -0.18687737, -2.2330395 , -2.23233896, -1.48845947])
    message: 'Optimization terminated successfully'
    nfev: 37
    nit: 6
    njev: 6
    status: 0
    success: True
        x: array([-0.76731079,  0.25577026,  0.62770999, -0.11616947,  0.25577026])
minimum values for the initial guess of [5, 4, 6, 7, 2] are      fun: 4.093023255813954
    jac: array([-2.04651159, -0.18604654, -2.23255819, -2.23255807, -1.48837209])
    message: 'Optimization terminated successfully'
    nfev: 37
    nit: 6
    njev: 6
    status: 0
    success: True
        x: array([-0.76744185,  0.25581395,  0.62790696, -0.11627906,  0.25581395])
minimum values for the initial guess of [5, 4, 6, 7, 2] are      fun: 4.09302325589637
    jac: array([-2.04650009, -0.18605363, -2.23255372, -2.23257113, -1.48837495])
    message: 'Optimization terminated successfully'
    nfev: 43
    nit: 7
    njev: 7
    status: 0
    success: True

```

```
x: array([-0.76743755,  0.25581252,  0.62791061, -0.11628558,  0.25581252])
minimum values for the initial guess of [5, 4, 6, 7, 2] are      fun: 4.093023255813954
jac: array([-2.04651165, -0.18604648, -2.23255819, -2.23255807, -1.48837209])
message: 'Optimization terminated successfully'
nfev: 31
nit: 5
njev: 5
status: 0
success: True
x: array([-0.76744187,  0.25581396,  0.62790695, -0.11627903,  0.25581396])
```



## Do you find different solutions?

Yes, as we can see from the different results that we got from using different initial guesses, we can say that there is a difference in calculation of the results, though method is the same, the precision of the answer is changing. This trend seems to be affecting the digits with the precision of  $10^{-5}$ . So, as it is perceived here, changing the initial guesses does impact the optimization answer.

Rumitkumar Miteshkumar Patel

ASU ID :  
1219783279.

MAE 598: Design Optimization  
HW - I

Problem-2

(a)  $f(x) = b^T x + x^T A x$ .

gradient  $g(x) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$

So, here,  $g(x) = b^T + \frac{\partial f}{\partial x}$

$g(x) = b + 2Ax$

(∴ here, considering matrix system as  
a linear system for ease of use).

Hessian  $H(x) = \frac{\partial^2 f}{\partial x^2} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$

Here,  $H(x) = \frac{\partial^2 f}{\partial x^2}$

$= \frac{\partial}{\partial x} (g(x))$

$\boxed{H(x) = 2A}$

(b) 1<sup>st</sup> order Taylor approximation of  $f(x)$

$$f(x) = f(x_0) + \nabla f \Big|_{x_0}^T (x - x_0)$$

$$\bullet f(x_0) = f(0) = 0$$

$$f(x) = 0 + (b + 2Ax) \Big|_{x_0}^T (x - x_0)$$

$$f(x) = \cancel{b} \quad b^T x \quad \text{--- (1)}$$

2<sup>nd</sup> order Taylor approximation of  $f(x)$

$$f(x) = f(x_0) + \nabla f \Big|_{x_0}^T (x - x_0) + \frac{1}{2} (x - x_0)^T H \Big|_{x_0} (x - x_0)$$

• from eq<sup>n</sup> ①

$$= b^T x + \frac{1}{2} (x - 0)^T (2A) (x - x_0)$$

$$\boxed{f(x) = b^T x + x^T A x}$$

These approximations are not exact. To reduce the error, we need higher-order Taylor's approximations which are being neglected here.

(c) For  $A$  to be positive definite the eigenvalues should be greater than zero. i.e.  $\lambda > 0$   
 ~~$\forall \lambda$  of~~

from the ~~equation~~ given function

$$x^T A x > 0 \text{ for all } x, \text{ where } x \neq 0.$$

And all upper left determinants  $> 0$ .

(d) necessary and sufficient condition for  $A$  to have full rank:

$$|A| \neq 0 \quad \underline{\text{or}} \quad \det(A) \neq 0$$

which means  $A$  should not be singular  
or all columns of  $A$  should be linearly independent from each other.

(e) Given is  $y \neq 0$ , and  $A^T y = 0$ .

$A^T y = 0$  which means  $y$  belongs to the null space of  $A^T$ .

To have a solution of  $x$ ,  $b$  must be a part of column space of  $A$ .  
must be  $b \in C(A)$

Let's have,

$$Ax = b$$

Taking transpose both sides.

$$(Ax)^T = b^T$$

$$(Ax)^T y = b^T y$$

multiplying  $y$  both sides)

$$(x^T A^T) y = b^T y.$$

$$x^T (A^T y) = b^T y.$$

$A^T y = 0$  which is given

So,  $b^T y = 0$

according to  $a^T b = \|a\| \|b\| \cos \theta = 0$ .

$$\cos \theta = 0$$

$$\theta = 90^\circ$$

which means  $a$  and  $b$  are perpendicular  
from that

~~$y$~~   $b$  and  $y$  are also perpendicular.  
and  ~~$y$~~ ,  $y \in N(A)$   
that means  $b$  belongs to column space of  $A$ .

$$\Rightarrow b \in C(A)$$

Problem 3>

N types of food  
which contains M types of Nutrition.

$a_{ij}$  → quantity of nutrition of j type of food type i

$c_i$  → price of food type i

$b_j$  → necessary nutrition of type j for a month.

Let's consider quantity of food type i as  $x_i$

So, minimize grocery cost which is

$$f = \sum_{i=1}^n c_i x_i$$

in other words we need to minimize the quantity of the food bought of type i

$$\min \left\{ x_i \right\}_{i=1}^n$$

It is obvious that  $x_i \geq 0$ , for all  $i \in N$

Also quantity of nutrition should be greater or equal to the min. value for a month

so,

$$\sum_{i=1}^N (a_{ij} x_i) \geq b_j, \forall j \in N$$