МИНИМИЗАЦИЯ КВАДРАТИЧНОЙ ФУНКЦИИ

In [1]:

```
import numpy as np
```

In [2]:

```
N = 1
eps = 1e-06
A = np.array([[4, 1, 1], [1, 2 * (3 + 0.1 * N), -1], [1, -1, 2 * (4 + 0.1 * N)]])
b = np.array([1, -2, 3])
a = np.diag(np.diag(A))
x0 = np.linalg.solve(a, -b)
print(x0)
```

```
[-0.25        0.32258065 -0.36585366]
```

In [3]:

```
def diag_pr(a):
    k = range(len(a))
    delta = a[0][0]
    for i in k:
        s = 0
        for j in k:
            if j != i:
                s += abs(a[i][j])
        if (abs(a[i][i]) - s) < delta:
            delta = a[i][i] - s
    return delta
```

In [4]:

```
def f(x):
    return (
        2 * x[0] ** 2 + (3 + 0.1 * N) * x[1] ** 2 + (4 + 0.1 * N) * x[2] ** 2
        + x[0] * x[1] - x[1] * x[2] + x[0] * x[2] + x[0] - 2 * x[1] + 3 * x[2] + N
    )
```

In [5]:

```
def gradf(x):
    return np.dot(A, x) + b
```

In [6]:

```
def mngs(x0):
    iter = 0
    while 1:
        iter = iter + 1
        q = gradf(x0)
        mhy = -((np.linalg.norm(gradf(x0)))**2) / (np.dot(
            np.dot(gradf(x0).transpose(), A), gradf(x0)))
        x = x0 + mhy * q
        delta = diag_pr(A)
        if diag_pr(A) > 0:
            if np.linalg.norm(np.dot(A, x) + b) / delta < eps:
                return x, iter
        else:
            if np.linalg.norm(f(x) - f(x0)) < eps:
                return x, iter
        x0 = x
```

In [7]:

```
def mps(x0):
    iter = 0
    while 1:
        for i in range(len(x0)):
            iter = iter + 1
            q = np.zeros(len(x0))
            q[i] = 1
            mhy = -(np.dot(q.transpose(), gradf(x0))) / (np.dot(
                np.dot(q.transpose(), A), q))
            x = x0 + mhy * q
            delta = diag_pr(A)
            if delta > 0:
                if np.linalg.norm(np.dot(A, x) + b) / delta < eps:
                    return x, iter
            else:
                if np.linalg.norm(f(x) - f(x0)) < eps:
                    return x, iter
            x0 = x
```

In [8]:

```
extr, it = mps(x0)
print("МПС")
print("Минимум функции", extr)
print("Значение функции в минимуме", f(extr))
print("Количество итераций", it)
```

```
МПС
Минимум функции [-0.25491895  0.31591599 -0.29623981]
Значение функции в минимуме 0.11226497595119544
Количество итераций 20
```

In [9]:

```
extr, it = mngs(x0)
print("МНГС")
print("Минимум функции", extr)
print("Значение функции в минимуме", f(extr))
print("Количество итераций", it)
```

```
МНГС
Минимум функции [-0.25491887  0.31591592 -0.2962396 ]
Значение функции в минимуме 0.11226497595116836
Количество итераций 13
```

Протестируем на функции с извесным минимумом

$f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$

$f(1, 3) = 0$ - Минимум функции

In [10]:

```
eps = 1e-06
A = np.array([[10, 8], [8, 10]])
b = np.array([-34, -38])
a = np.diag(np.diag(A))
x0 = np.linalg.solve(a, -b)
print(x0)
```

```
[3.4 3.8]
```

In [11]:

```
def f(x):
    return ((x[0] + 2*x[1] - 7)**2 + (2*x[0] + x[1] - 5)**2)
```

In [12]:

```
extr, it = mps(x0)
print("МПС")
print("Минимум функции", extr)
print("Значение функции в минимуме", f(extr))
print("Количество итераций", it)
```

```
МПС
Минимум функции [0.99920772 3.00099035]
Значение функции в минимуме 1.7654348630768776e-06
Количество итераций 31
```

In [13]:

```
extr, it = mngs(x0)
print("МНГС")
print("Минимум функции", extr)
print("Значение функции в минимуме", f(extr))
print("Количество итераций", it)
```

```
МНГС
Минимум функции [1.00002335 3.00000778]
Значение функции в минимуме 4.482059555986774e-09
Количество итераций 6
```

In [ ]:

```

```