Міністерство освіти та науки України
Національний Технічний Університет України
«Київський Політехнічний Інститут»

Фізико – технічний інститут

Лабораторна робота №2

«Метод спряжених градієнтів»

Виконали:
Студенти 4 курсу
Групи ФІ-51
Макаренко Сергій
Скірдін Євгеній
Сітко Дарина
Сімакова Катерина

Перевірив:
Данилов В.Я

Київ, 2018 р.

Теоретичні відомості

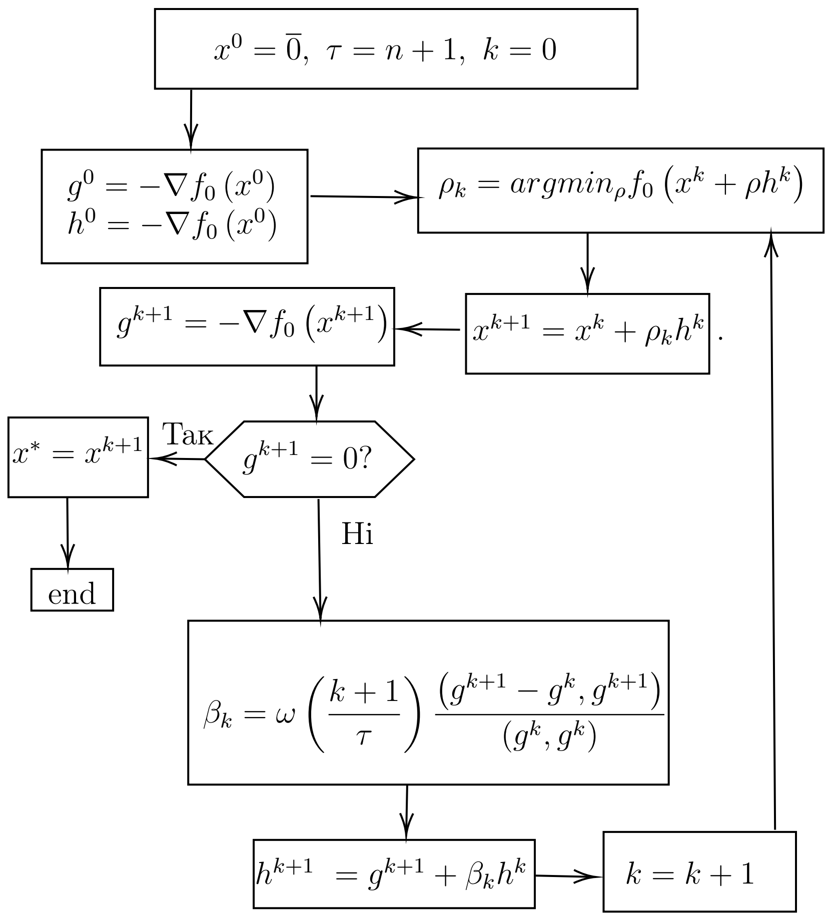
Нехай f(x) – випукла диференційована в усьому просторі функція і треба знайти її точку мінімуму.

Тобто знайти $\operatorname*{argmin}_{x \in R^n} f_0(x)$ для заданої неперервно диференційованої функції $f_0 : R^n \to R^1$

Алгоритм:

1. Вибрати довільне початкове наближення $x^0 \in R^n$, натуральне число $\tau \geqslant n$ (τ – момент відновлення), покласти $k = 0$.
2. Обчислити $\nabla f_0\left(x^0\right)$ і покласти $g^0 = -\nabla f_0\left(x^0\right)$, $h^0 = -\nabla f_0\left(x^0\right)$.
3. Обчислити кроковий множник ρ_k , що задовольняє умову $f_0\left(x^k + \rho_k h^k\right) = \min_{\rho \geqslant 0} f_0\left(x^k + \rho h^k\right)$.
4. Обчислити наближення: $x^{k+1} = x^k + \rho h^k$.
5. Обчислити $\nabla f_0\left(x^{k+1}\right)$ і покласти $g^{k+1} = -\nabla f_0\left(x^{k+1}\right)$
6. Якщо $g^{k+1} = 0$, то покласти $x^* = x^{k+1}$ і завершити обчислення, інакше перейти на крок 7.
7. Обчислити коефіцієнт $\beta_k = \omega\left(\dfrac{k+1}{\tau}\right)\dfrac{\left(g^{k+1} - g^k, g^{k+1}\right)}{\left(g^k, g^k\right)}$, де
$$\omega(t) = \begin{cases} 0, & if \;\; t - integer \\ 1, & else \end{cases}$$
8. Обчислити вектор $h^{k+1} = g^{k+1} + \beta_k h^k$.
9. Покласти $k = k + 1$ і перейти на крок 3.

Блок - схема алгоритму:



```
import numpy as np
teta=3
x = np.array([[0],[0]],float)

print("x", x)
k=0

def f(x):
        return(x[0]*x[0]+2*x[1]*x[1]+np.exp(x[0]*x[0]+x[1]*x[1])-x[0]

def get_grad(x):
    grad_f = np.array([2*x[0]+2*x[0]*np.exp(x[0]*x[0]+x[1]*x[1])-1,4
    return(grad_f)

grad_f=get_grad(x)
print(grad_f)
g=-grad_f
h=-grad_f

while 1:
        rho = 1
        max = 100
        for a in np.arange(0.,1.,0.01):
            s=f(x+a*h)
            if s<max:
                max=s
                rho=a

        x=x+rho*h

        g_old=g
        g=-get_grad(x)

        if np.linalg.norm(g) <0.001:
                break

        beta=np.dot(np.transpose(g-g_old),g)/np.dot(np.transpose(g_o
        if k%(teta-1)==0:
```

```
            beta=0

        h=g+beta*h
        k=k+1
print("x ", x)
```

Результати: $x_1 = -0.5$; $x_2 = 0$; $f\left(\overline{x}\right) = -0.25$