# Мiнiстерство освiти та науки України Нацiональний Технiчний Унiверситет України

«Київський Полiтехнiчний Iнститут» Фiзико – технiчний iнститут

Лабораторна робота №2

«Метод спряжених градiєнтiв»

Виконали: Студенти 4 курсу Групи ФI-51 Макаренко Сергiй Скiрдiн Євгенiй Сiтко Дарина Сiмакова Катерина

Перевiрив: Данилов В.Я

Київ, 2018 р.

Теоретичнi вiдомостi

Нехай f(x) – випукла диференцiйована в усьому просторi функцiя i треба знайти її точку мiнiмуму.

Тобто знайти argmin *f*0(*x*) для заданої неперервно диференцiйо-

*x*∈*Rn*

ваної функцiї *f*0 : *Rn* → *R*1

Алгоритм:

1. Вибрати довiльне початкове наближення *x*0 *Rn* , натуральне число *τ* “ *n* ( *τ* – момент вiдновлення), покласти *k* = 0.

∈

. Σ . Σ . Σ

1. Обчислити ∇*f*0 *x*0 i покласти *g*0 = −∇*f*0 *x*0 , *h*0 = −∇*f*0 *x*0 .
2. Обчислити кроковий множник *ρk* , що задовольняє умову

. Σ . Σ0 *k* 0

*f xk* + *ρ hk* = min *f xk* + *ρhk* .

*ρ*“0

1. Обчислити наближення: *xk*+1 = *xk* + *ρhk*.

. Σ . Σ

1. Обчислити ∇*f*0 *xk*+1 i покласти *gk*+1 = −∇*f*0 *xk*+1
2. Якщо *gk*+1 = 0 , то покласти *x*∗ = *xk*+1 i завершити обчислення, iнакше перейти на крок 7.
3. Обчислити коефiцiєнт

.

.*k* + 1 Σ .*gk*+1 − *gk, gk*+1Σ

, де

*βk* = *ω τ*

*ω*(*t*) = 0*, if t* − *integer*

1*, else*

(*gk, gk*)

1. Обчислити вектор *hk*+1 = *gk*+1 + *βkhk .*
2. Покласти *k* = *k* + 1 i перейти на крок 3.



*gk*+1 = −∇*f*0 .*xk*+1Σ

Так

*gk*+1 = 0?

Нi

*β* = *ω*

*k*

.

*k* + 1

*τ*

*hk*+1

= *gk*+1 + *βkhk*

end

*x*∗ = *xk*+1



Блок - схема алгоритму:



*h*0 = −∇*f*0 (*x*0)

*g*0 = −∇*f*0 (*x*0)



*x*0 = 0*, τ* = *n* + 1*, k* = 0



|  |  |  |  |
| --- | --- | --- | --- |
| *ρk* = *argminρf*0 .*xk* + *ρhk*Σ | | | |
| *xk*+1 = *xk* + *ρkhk .*  Σ .*gk*+1 − *gk, gk*+1Σ  (*gk, gk*) | |  | |
|  | *k* = *k* + 1 | |  |

**import** numpy as np te ta=3

x = np . array ( [ [ 0 ] , [ 0 ] ] , **f loat** )

**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 . 0 1 ) : s=f ( x+a h)

∗

**i f** s<**max**:

**max**=s rho=a

x=x+rho ∗h

g\_old=g g=−get\_grad ( x )

**i f** np . l i n a l g . norm ( g ) <0 . 001:

**break**

beta=np . dot ( np . tra n s po s e ( g−g\_old ) , g )/ np . dot ( np . tra ns po s e ( g\_o

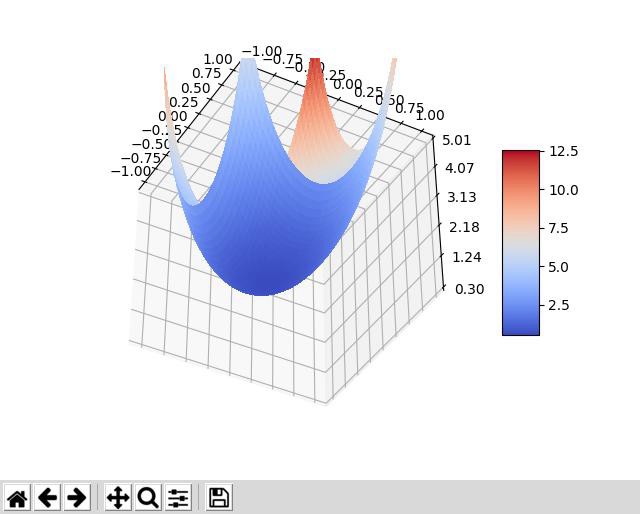
**i f** k%( teta −1)==0:

beta=0

h=g+beta h k=k+1

∗

**print** ( "x¸" , x )



Результати: *x*1 = −0*.*5; *x*2 = 0; *f* (*x*) = −0*.*25