# Мiнiстерство освiти та науки України Нацiональний Технiчний Унiверситет України

«Київський Полiтехнiчний Iнститут» Фiзико – технiчний iнститут

Лабораторна робота №1

«Метод спряжених напрямкiв»

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* , довiльну симет- ричну строго додатно визначену матрицю *H*0 , покласти *k* = 0.

∈

1. Обчислити *Axk* + *b* i покласти *zk* = *Axk* + *b* . Якщо *zk* = 0 , то покласти *x*∗ = *xk* i завершити обчислення, iнакше перейти на крок 3.
2. Якщо *k* = 0 , то перейти на крок 6, iнакше перейти на крок 4.
3. Обчислити вектори: *rk*−1 = *xk* − *xk*−1 , *gk*−1 = *Ark*−1.

*H gk*−1 .*rk*−1Σ*T*

5. Обчислити матрицю *H*

*k*

= *H*

*k*−1

−

0

(*rk*−1*, gk*−1)

1. Обчислити вектор руху *hk* до наближення *xk*+1 *hk* = −*HT zk*

*k*

. Σ

1. Обчислити кроковий множник

*zk, hk*

*ρk* = −(*Ahh, hk* ) .

1. Обчислити наближення *xk*+1 = *xk* + *ρkhk .*
2. Покласти *k* = *k* + 1 i перейти на крок 2.

end

Блок - схема алгоритму:



*zk* = 0?

Так

Нi

Нi

*k* = 0?

Так

*Hk* =

*Hk*−1 −

*H g r*

0

*k*−1 *k*−1

(*rk*−1*, gk*−1)

. Σ

*T*

*.*

*k* = *k* + 1

Σ

*, h* )

*h k*

*zk, hk*

.

(*Ah*

*k*

*ρ* = −

*rk*−1 = *xk* − *xk*−1 *gk*−1 = *Ark*−1

*x*∗ = *xk*

*zk* = *Axk* + *b*

*xk*+1 = *xk* + *ρkhk*

*T k k*

*k*

*h* = −*H z*



*x*0 = 0*, H*0 = *I, k* = 0



**import** numpy as np

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

H = np . array ( [ [ 1 , 0 ] , [ 0 , 1 ] ] , **f loat** )

A = np . array ( [ [ 2 , − 2 ] , [ − 2 , 12 ] ] , **f loat** )

b = np . array ( [ [ 1 ] , [ − 1 ] ] , **f loat** )

z= np . dot (A, x)+b

h=−np . dot ( np . tra ns p o s e (H) , z )

rho = −np . dot ( np . tra ns p o s e ( z ) , h)/ np . dot ( np . tra ns p o s e ( np . dot (A, h ) ) , h) x\_old=x

x=x+rho ∗h

z= np . dot (A, x)+b

H0=H

**while** ( np . l i n a l g . norm ( z ) > 0 . 0001 ) : r=x x\_old

−

g=np . dot (A, r )

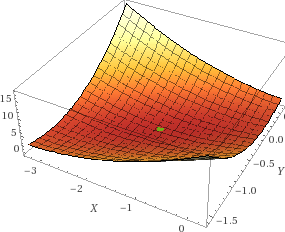
H=H−np . dot ( np . dot ( H0 , g ) , np . tra n s po s e ( r ) ) / np . dot ( np . tra ns po s e h=−np . dot ( np . tra ns p o s e (H) , z )

rho = −np . dot ( np . tra ns p o s e ( z ) , h)/ np . dot ( np . tra ns p o s e ( np . dot (

x\_old=x x=x+rho ∗h

z= np . dot (A, x)+b

**print** ( " r e z u l t " , x )

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