Міністерство освіти і науки України НТУУ «Київський політехнічний інститут ім. Ігоря Сікорського»

Фізико-технічний інститут

**Розрахункова робота**

з методів оптимізації  
на тему: «Метод проекції градієнту»

Виконали:

студенти 4 курсу

ФТІ групи ФІ-52

Бурлака Марія

Єршов Степан

Кратт Ярослав

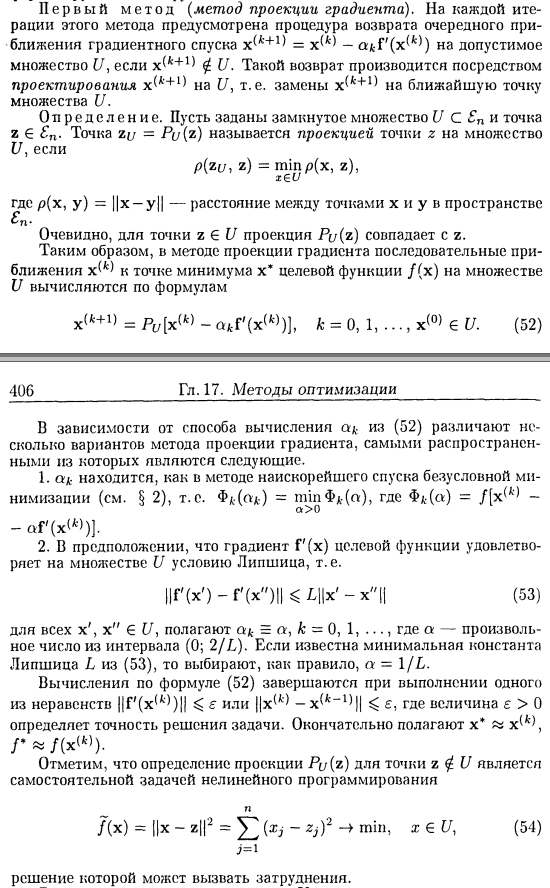
Овчарова Марина

Перевірив:

Данилов В.Я.

Київ   
2018

Метод проекції градієнту



**Завдання:** розв’язати задачу нелінійного програмування методом проекції градієнту.

**Умови:**

1.  
1.1.

**Розв’язання:**

public class GradientProjectionMethod {  
 double x1 = 2, x2 = 2, am = 1, bord = 4;  
  
 public static double function(double x1, double x2) {  
 return 7\*x1\*x1-8\*x1+5\*x2\*x2;  
 }  
  
 public static double derivativeFuncX1(double x1, double x2) {  
 return 14\*x1-8;  
 }  
  
 public static double derivativeFuncX2(double x1, double x2) {  
 return 10\*x2;  
 }  
  
 public static double functionIter(double x1, double x2, double a) {  
 return function(x1 - a\*derivativeFuncX1(x1, x2), x2 - a\*derivativeFuncX2(x1, x2));  
 }  
  
 public double functionBound() {  
 return x1\*x1 + (x2 - 4)\*(x2 - 4);  
 }  
  
 public boolean lessOrEqual() {  
 boolean b = false;  
 if (functionBound() <= bord) {b = true;}  
 return b;  
 }  
  
 public boolean equal() {  
 boolean b = false;  
 if (functionBound() == bord) {b = true;}  
 return b;  
 }  
  
 public boolean moreOrEqual() {  
 boolean b = false;  
 if (functionBound() >= bord) {b = true;}  
 return b;  
 }  
  
 public void proc(boolean ind) {  
 double min = 100;  
 while ((Math.abs(derivativeFuncX1(x1, x2)) > 0.01)&&(Math.abs(derivativeFuncX2(x1, x2)) > 0.01)) {  
 for (double a = 0; a < 1; a = a + 0.01) {  
 double s = functionIter(x1 ,x2, a);  
 if (min > s) {  
 min = s;  
 am = a;  
 }  
 }  
 //System.out.println("min function(a) = " + min + " for a = " + am + ", ");  
 double temp = x1;  
  
 if (ind == true) {  
 x1 = x1 - am\*derivativeFuncX1(x1, x2);  
 x2 = x2 - am\*derivativeFuncX2(temp, x2);  
 } else {  
 x1 = (x1 / (Math.sqrt(x1\*x1 + x2\*x2))\*(x1 - am\*derivativeFuncX1(x1, x2)));  
 x2 = (x2 / (Math.sqrt(temp\*temp + x2\*x2))\*(x2 - am\*derivativeFuncX2(temp, x2)));  
 }  
  
 //System.out.println("x = (" + x1 + ", " + x2 + ")\n");  
 }  
 System.out.println("\_\_\_\_\_\_\_\_\_\_\nmin function(a) = " + min + " for a = " + am + ", x = (" + x1 + ", " + (int)x2 + ")\n");  
 }  
  
 public static void main(String[] args) {  
 GradientProjectionMethod lessOrEqual = new GradientProjectionMethod();  
 GradientProjectionMethod equal = new GradientProjectionMethod();  
 GradientProjectionMethod moreOrEqual = new GradientProjectionMethod();  
 lessOrEqual.proc(lessOrEqual.lessOrEqual());  
 equal.proc(equal.equal());  
 moreOrEqual.proc(moreOrEqual.moreOrEqual());  
 }  
}

