**Липецкий государственный технический университет**

Факультет автоматизации и информатики
Кафедра прикладной математики

Отчет по лабораторной работе № 1
по дисциплине «Компьютерные технологии математических исследований»
на тему «Исследование эффективности неклассических процедур оптимизации»

| | | |
|---|---|---|
| Студент | ____________ <br> подпись, дата | Комолых Т.О. <br> фамилия, инициалы |
| Студент | ____________ <br> подпись, дата | Кобзев А.А. <br> фамилия, инициалы |

Группа ПМ-18

| | | |
|---|---|---|
| Руководитель <br> к.т.н., доцент <br> ученая степень, ученое звание | ____________ <br> подпись, дата | Сысоев А.С. <br> фамилия, инициалы |

Липецк 2021 г.

**Задание кафедры**

1. Реализовать функцию, которая:
   - ▷ Принимает на вход целевую функцию для оптимизации (минимизации или максимизации). Указать выбранный алгоритм оптимизации;
   - ▷ Возвращает найденное оптимальное значение целевой функции и значения аргументов, которые его доставляют;
   - ▷ Указать время выполнения оптимизации;
2. Сравнить эффективность трех алгоритмов оптимизации на двух тестовых функциях.
3. Сделать выводы.

## Оглавление

**Выполнение работы**

Теоретический материал

Оптимизация — это задача нахождения экстремума (минимума или максимума) целевой функции в некоторой области конечномерного векторного пространства, ограниченной набором линейных и/или нелинейных равенств и/или неравенств.

Стандартная математическая задача оптимизации формулируется таким образом. Среди элементов $\chi$, образующих множества $\chi$, найти такой элемент $\chi*$, который доставляет минимальное значение $f(\chi*)$ заданной функции $f(\chi)$. Для того, чтобы корректно поставить задачу оптимизации, необходимо задать:

1. Допустимое множество — множество $X = \{\vec{x}|\ g_i(\vec{x}) \leq 0,\ i = 1, \ldots, m\} \subset R^n$;
2. Целевую функцию — отображение $f:\ X \to Rf$ :;
3. Критерий поиска (max или min).

Тогда решить задачу $f(x) \to \min\limits_{\vec{x}\in \mathrm{X}}$ означает одно из:

1. Показать, что $X =$.
2. Показать, что целевая функция $f(\vec{x})$ не ограничена снизу.
3. Найти $\vec{x}^* \in X:\ f(\vec{x}^*) = \min\limits_{\vec{x}\in X} f(\vec{x})$.
4. Если $\vec{x}^*$, то найти $\inf\limits_{\vec{x}\in X} f(\vec{x})$.

Если минимизируемая функция не является выпуклой, то часто ограничиваются поиском локальных минимумов и максимумов: точек $x_0$ таких, что всюду в некоторой их окрестности $f(x) \geq f(x_0)$ для минимума и $\leq f(x_0)$ для максимума.

Если допустимое множество $X = R^n$, то такая задача называется задачей безусловной оптимизации, в противном случае — задачей условной оптимизации.

Практическая часть

Целевые функции

1. В качестве первой целевой функции использовалась функция 1.

$$y = x^2 - 2x + 30 \tag{1}$$

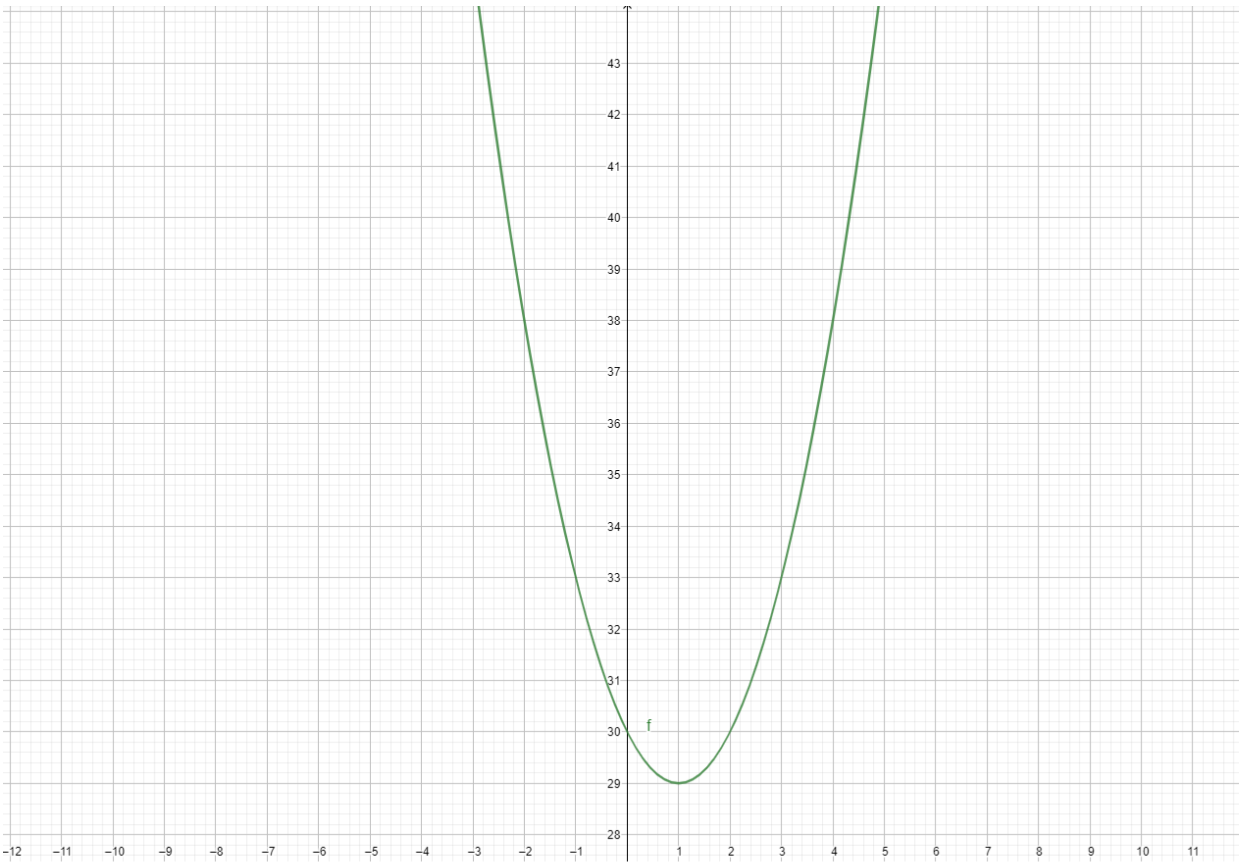


Рисунок 1 – График первой целевой функции

2. В качестве второй целевой функции использовалась функция 2.

$$y = e^{(x+3)} + x^3 \tag{2}$$

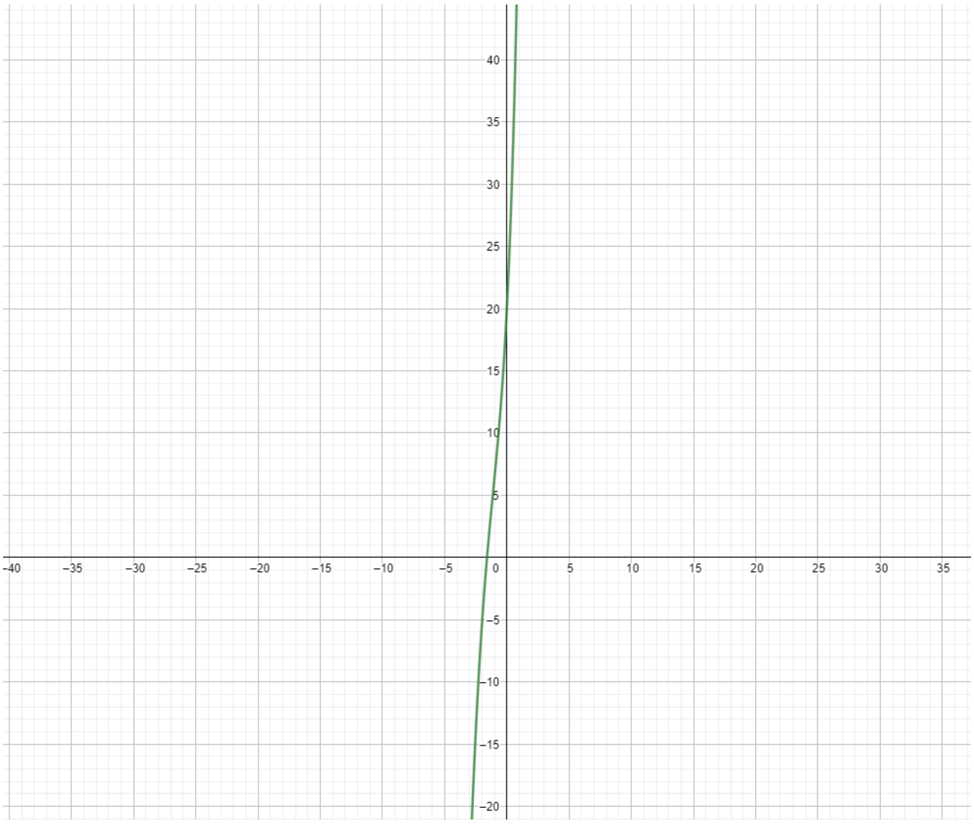


Рисунок 2 – График второй целевой функции

Реализация методов

## 1. Метод имитации отжига

```
hchange=function(par,lower,upper,dist,round=TRUE,...)
{
  D=length(par)
  step=dist(D,...)
  if(round)
    step=round(step)
  par1=par+step
  return(ifelse(par1<lower,lower,ifelse(par1>upper,upper,par1)))
}

bchange=function(par)
{
  D=length(par)
  hchange(par,lower=rep(0,D),upper=rep(1,D),rnorm,mean=0,sd=1)
}
D=8
```

## 2. Метод Монте-Карло

```
mcsearch=function(N,lower,upper,FUN,type="min",...)
{
  D=length(lower)
  s=matrix(nrow=N,ncol=D)
  for(i in 1:N)
    s[i,]=runif(D,lower,upper)
  fsearch(s,FUN,type,...)
}

fsearch=function(search,FUN,type="min",...)
{
    x=apply(search,1,FUN,...)
    ib=switch(type,min=which.min(x),max=which.max(x))
    return(list(index=ib,sol=search[ib,],eval=x[ib]))
}
```

## 3. Метод восхождения к вершине

```
hclimbing=function(par,fn,change,lower,upper,control, type="min",...)
{
    fpar=fn(par,...)
    for(i in 1:control$maxit)
```

```
    {
      par1=change(par,lower,upper)
      fpar1=fn(par1,...)
      if(control$REPORT>0 &&(i==1||i%%control$REPORT==0))
        cat("i:",i,"s:",par,"f:",fpar,"s’",par1,"f:",fpar1,"\n")
      if( (type=="min" && fpar1<fpar) || (type=="max" && fpar1>fpar))
      {
        par=par1;fpar=fpar1
      }
    }
    if(control$REPORT>=1) cat("best:",par,"f:",fpar,"\n")
      return(list(sol=par,eval=fpar))
}

hchange=function(par,lower,upper,dist,round=TRUE,...)
{
    D=length(par)
    step=dist(D,...)
    if(round)
      step=round(step)
    par1=par+step
    return(ifelse(par1<lower,lower,ifelse(par1>upper,upper,par1)))
}

ichange=function(par,lower,upper)
{
    hchange(par,lower,upper,rnorm,mean=0,sd=1)
}
```

## 4. Функция для вызова определённого метода

```
result=function(Mymethod, Myfunction, arg)
{
    answer = "Такого метода нет"
    time = Sys.time()
    if(Mymethod == "Отжиг"){
        C=list(maxit=10,temp=10,tmax=1,trace=TRUE,REPORT=1)
        answer=optim(arg,Myfunction,gr=bchange,method="SANN", control=C)
        cat("best:",answer$par,"f:",answer$value,"(max: fs:",sum(answer$par),")\n")
    }
    if(Mymethod == "МК"){
        answer=mcsearch(10000,arg,rep(0,5),Myfunction,"min")
        cat("best:",answer$sol,"f:",answer$eval,"\n")
    }
    if(Mymethod == "Вершина"){
        answer=list(maxit=10,REPORT=1)
        D=8
```

```r
        hclimbing(arg,Myfunction,change=ichange,lower=rep(0,D),upper=rep(1,D), control=answer,type="min")
    }
    if(answer == "Такого метода нет"){
        cat(answer)
        return (-1)
    }
    time = paste0(round(as.numeric(difftime(time1 = Sys.time(), time2 = time, units = "secs")), 3))
    cat("Метод:", Mymethod, "\nвремя работы алгоритма:",time)
    return (list(timer = as.numeric(time), method = Mymethod))
}
```

Применение методов к целевым функциям

```
sann objective function values
initial       value 60.000000
iter        1 value 59.000000
iter        2 value 59.000000
iter        3 value 58.000000
iter        4 value 58.000000
iter        5 value 58.000000
iter        6 value 58.000000
iter        7 value 58.000000
iter        8 value 58.000000
iter        9 value 58.000000
final         value 58.000000
sann stopped after 9 iterations
best: 1 1 f: 58 (max: fs: 2 )
Метод: Отжиг
время работы алгоритма: 0.037
```

Рисунок 3 – Результат применения метода имитации отжига к первой функции

```
sann objective function values
initial       value 60.256611
iter        1 value 60.256611
iter        2 value 60.256611
iter        3 value 60.256611
iter        4 value 60.256611
iter        5 value 60.256611
iter        6 value 60.256611
iter        7 value 60.256611
iter        8 value 60.256611
iter        9 value 60.256611
final         value 60.256611
sann stopped after 9 iterations
best: 0 0 0 f: 60.25661 (max: fs: 0 )
Метод: Отжиг
время работы алгоритма: 0.036
```

Рисунок 4 – Результат применения метода имитации отжига ко второй функции

```
best: 0 0 f: 60
Метод: МК
время работы алгоритма: 0.096
best: 0 0 0 f: 60.25661
Метод: МК
время работы алгоритма: 0.115
```

Рисунок 5 – Результат применения метода Монте-Карло к функциям

```
i: 1 s: 0 0 f: 60 s’ 1 0 1 0 1 0 1 0 f: 236
i: 2 s: 0 0 f: 60 s’ 1 0 1 0 1 0 1 0 f: 236
i: 3 s: 0 0 f: 60 s’ 0 1 0 1 0 1 0 1 f: 236
i: 4 s: 0 0 f: 60 s’ 1 0 1 0 1 0 1 0 f: 236
i: 5 s: 0 0 f: 60 s’ 0 1 0 1 0 1 0 1 f: 236
i: 6 s: 0 0 f: 60 s’ 0 1 0 1 0 1 0 1 f: 236
i: 7 s: 0 0 f: 60 s’ 0 0 0 0 0 0 0 0 f: 240
i: 8 s: 0 0 f: 60 s’ 1 0 1 0 1 0 1 0 f: 236
i: 9 s: 0 0 f: 60 s’ 0 0 0 0 0 0 0 0 f: 240
i: 10 s: 0 0 f: 60 s’ 0 0 0 0 0 0 0 0 f: 240
best: 0 0 f: 60
Метод: Вершина
время работы алгоритма: 0.656
```

Рисунок 6 – Результат применения метода восхождения к вершине к первой функции

```
i: 1 s: 0 0 0 f: 60.25661 s’ 0 1 0 0 1 0 0 1 f: 267.2221
i: 2 s: 0 0 0 f: 60.25661 s’ 0 0 0 0 0 0 0 0 f: 160.6843
i: 3 s: 0 0 0 f: 60.25661 s’ 0 1 0 0 1 0 0 1 f: 267.2221
i: 4 s: 0 0 0 f: 60.25661 s’ 0 1 0 0 1 0 0 1 f: 267.2221
i: 5 s: 0 0 0 f: 60.25661 s’ 1 1 0 1 1 0 1 1 f: 373.76
i: 6 s: 0 0 0 f: 60.25661 s’ 1 1 0 1 1 0 1 1 f: 373.76
i: 7 s: 0 0 0 f: 60.25661 s’ 1 0 1 1 0 1 1 0 f: 338.2474
i: 8 s: 0 0 0 f: 60.25661 s’ 0 1 1 0 1 1 0 1 f: 338.2474
i: 9 s: 0 0 0 f: 60.25661 s’ 0 0 0 0 0 0 0 0 f: 160.6843
i: 10 s: 0 0 0 f: 60.25661 s’ 0 1 0 0 1 0 0 1 f: 267.2221
best: 0 0 0 f: 60.25661
Метод: Вершина
время работы алгоритма: 0.644
```

Рисунок 7 – Результат применения метода восхождения к вершине ко второй функции

```
  timer method
1 0.037  Отжиг
  timer method
1 0.036  Отжиг
```

Рисунок 8 – Метод, отработавший за наименьшее время для двух целевых функций

## Заключение

В ходе лабораторной работы были изучены неклассические процедуры оптимизации, такие как: метод имитации отжига, метод Монте-Карло и восхождения к вершине.

Применение данных методов, реализованных на R, к заданным целевым функциям показало, что более эффективным является метод имитации отжига, отработавший за наименьшее время по сравнению с другими методоми из приведённого списка.