**Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

Факультет «Информатика и системы управления»

Кафедра «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 10

по курсу « Методы оптимизации »

**МЕТОД ГОЛЬШТАЙНА И ПРАЙСА**

Студент: Яровикова А. С.

Группа: ИУ9-81Б

Преподаватель: Посевин Д. П.

Москва, 2024

СОДЕРЖАНИЕ

[ЦЕЛЬ И ПОСТАНОВКА ЗАДАЧИ 3](#_Toc158799284)

[ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ 3](#_Toc158799285)

[РЕЗУЛЬТАТЫ 6](#_Toc158799286)

[ВЫВОДЫ 7](#_Toc158799287)

# **ЦЕЛЬ И ПОСТАНОВКА ЗАДАЧИ**

**Цель работы:**

Реализовать поиск экстремума методом Гольштайна и Прайса.

**Постановка задачи:**

1. Исследовать алгоритм методом Гольштайна и Прайса.
2. Реализовать численные методы оптимизации на языке программирования Julia, включая функции для вычисления градиента и гессиана, а также методы поиска оптимального шага.
3. Применить алгоритмы оптимизации к заданной функции многих переменных с целью минимизации значения функционала, используя различные начальные условия и параметры методов по вариантам

Вариант 1. .

# **ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ**

**using** LinearAlgebra

**using** Plots

**function test**(x)

(x[**1**]-**4**\*x[**2**])^**2** +(x[**2**]+**5**)^**2**

**end**

**function df**(f, x::Vector{**Float64**}, i, h=**1e-5**)

x\_plus = copy(x)

x\_minus = copy(x)

x\_plus[i] += h

x\_minus[i] -= h

df\_dx = (f(x\_plus) - f(x\_minus)) / (**2**h)

**return** df\_dx

**end**

**function gradient**(f, x)

**return** [df(f, x, **1**), df(f, x, **2**)]

**end**

**function find\_alpha**(f, delta, x\_k, phi)

alpha = **100.0**

F(alpha) = -(f(x\_k) - f(x\_k + alpha \* phi)) / (alpha \* transpose(gradient(f, x\_k)) \* phi)

**while** !(delta <= F(alpha) <= (**1** - delta))

alpha /= **2**

# if alpha == 1

# alpha -= 0.03

# end

**if** alpha < **1e-8**

**break**

**end**

**end**

**return** alpha

**end**

**function golstein\_price\_**(f, x\_0, delta=**0.25**, r=**1.0**, eps=**1e-3**)

E = [**1.0** **0.0**; **0.0** **1.0**]

x = [x\_0]

x\_k = x\_0

hess = [**0.0** **0.0**; **0.0** **0.0**]

k = **0**

phi = -gradient(f, x\_k)

teta = r

alpha = **100.0**

**while** true

**for** i **in** **1**:**2**

h = gradient(f, x\_k + teta \* E[:, i]) - gradient(f, x\_k)

hess[:, i] = h

**end**

**if** transpose(gradient(f, x\_k)) \* inv(hess) \* gradient(f, x\_k) > **0**

phi = -inv(hess) \* gradient(f, x\_k)

**else**

phi = -gradient(f, x\_k)

**end**

alpha = find\_alpha(f, delta, x\_k, phi)

x\_k = x\_k + alpha \* phi

push!(x, x\_k)

teta = r \* norm(phi)

k += **1**

**if** norm(phi) < eps

**break**

**end**

**end**

**return** x\_k, x, k

**end**

x\_0 = [**10.0**, -**5.0**]

x\_min, x, k = golstein\_price\_(test, x\_0)

println("Метод Гольштайна и Прайса:**\n** точка минимума: $x\_min**\n** кол-во итераций = $k")

px = [x[i][**1**] **for** i **in** **1**:length(x)]

py = [x[i][**2**] **for** i **in** **1**:length(x)]

x = y = -**15**:**1**:**5**;

x = -**30**:**1**:**100**

y = -**50**:**1**:**30**

contour(x, y, (x, y) -> test([x, y]), levels = **10**, xlabel="x", ylabel="y", colorbar=false, size=(**500**, **500**))

scatter!(px, py, markersize=**2**, markershape=:circle, markercolor=:red, label = "")

p = plot!(px, py, label="Движение", line=:red)

p

**function ros**(x)

**return** (**1** - x[**1**])^**2** + **100**\*(x[**2**] - x[**1**]^**2**)^**2**

**end**

x\_0 = [**2.0**, **2.0**]

x\_min, x, k = golstein\_price\_(Rosenbrok, x\_0, **0.25**, **1.0**, **1e-9**)

println("Метод Гольштайна и Прайса:**\n** точка минимума: $x\_min**\n** кол-во итераций = $k")

px = [x[i][**1**] **for** i **in** **1**:length(x)]

py = [x[i][**2**] **for** i **in** **1**:length(x)]

x = y = -**15**:**1**:**5**;

contour(x, y, (x, y) -> ros([x, y]), levels = **10**, xlabel="x", ylabel="y", colorbar=false, size=(**500**, **500**))

scatter!(px, py, markersize=**2**, markershape=:circle, markercolor=:red, label = "")

p = plot!(px, py, label="Движение", line=:red)

p

# **РЕЗУЛЬТАТЫ**

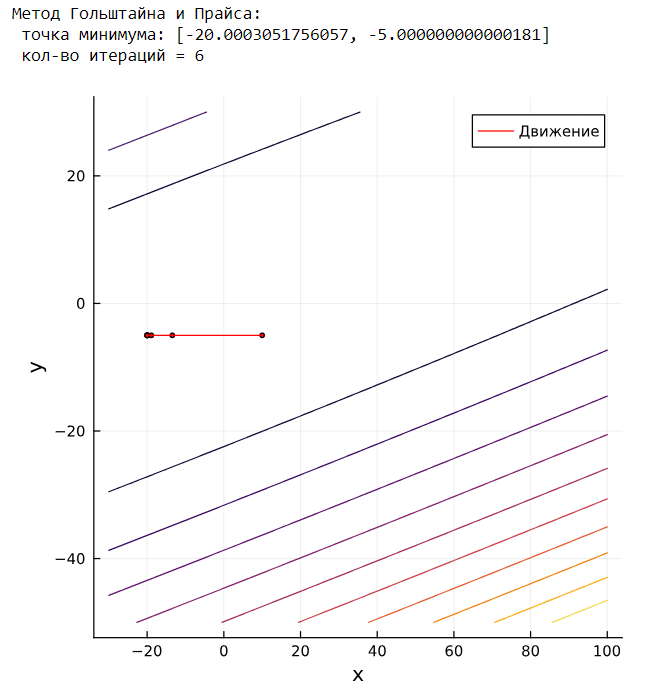


Рисунок 1 - Метод Гольштайна и Прайса на функции по варианту

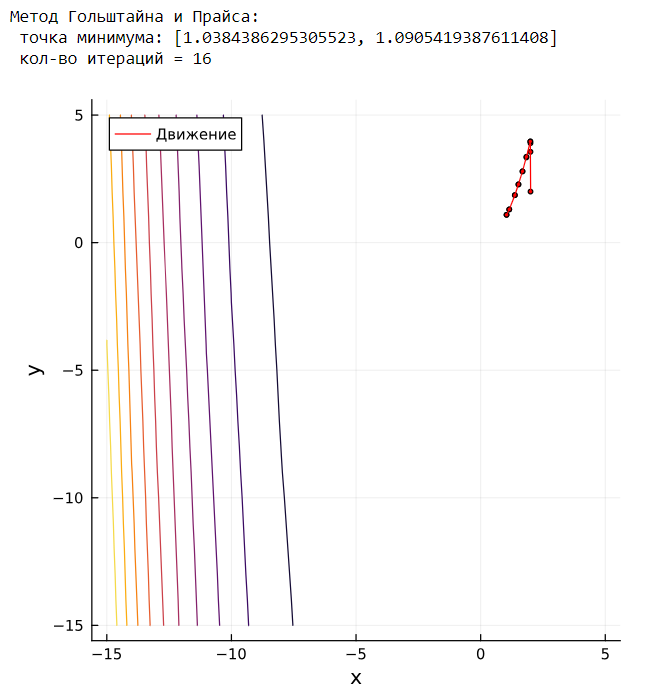


Рисунок 2 - Метод Гольштайна и Прайса на функции Розенброка

# **ВЫВОДЫ**

В ходе выполнения лабораторной работы был реализован метод Гольштайна и Прайса.