

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине

«Математическое программирование»

Студент

Группа АИ-20-1

Подпись, дата

Глубоков Г.В.

Руководитель

к.т.н., доц.

Подпись, дата

Качановский Ю.П.

Липецк 2022г.

Задание кафедры:

Написать программу, реализующую один из методов оптимизации. Проверить работу программы тестами из лабораторных работ. Сравнить и проанализировать результаты. Программа должна позволять вводить тесты из файла и из формы ввода, а также сохранять отчет о результатах работы.

Вариант 6: Метод Хука-Дживса

1. Описание методов и алгоритмов решения

1.1 Метод деления интервала пополам

Метод Хука — Дживса, также известный как **метод конфигураций** — как и алгоритм Нелдера — Мида, служит для поиска безусловного локального экстремума функции и относится к прямым методам, то есть опирается непосредственно на значения функции. Алгоритм делится на две фазы: исследующий поиск и поиск по образцу.

На начальном этапе задаётся стартовая точка (обозначим её 1) и шаги h_i по координатам. Затем замораживаем значения всех координат кроме 1-й, вычисляем значения функции в точках x_0+h_0 и x_0-h_0 (где x_0 — первая координата точки, а h_0 — соответственно значение шага по этой координате) и переходим в точку с наименьшим значением функции. В этой точке замораживаем значения всех координат кроме 2-й, вычисляем значения функции в точках x_1+h_1 и x_1-h_1 , переходим в точку с наименьшим значением функции и т. д. для всех координат. В случае, если для какой-нибудь координаты значение в исходной точке меньше, чем значения для обоих направлений шага, то шаг по этой координате уменьшается. Когда шаги по всем координатам h_i станут меньше соответствующих значений ϵ_i , алгоритм завершается, и точка 1 признаётся точкой минимума.

На этапе поиска по образцу откладывается точка 3 в направлении от 1 к 2 на том же расстоянии. Её координаты получаются по формуле $x_i = x_1 + \lambda(x_2 - x_1)$, где x_i — точка с номером i , λ — параметр алгоритма, обычно выбирающийся равным $2/3$. Затем в новой точке 3 проводится исследующий поиск, как на 1 фазе алгоритма, за исключением того, что шаг на этой фазе не уменьшается. Если на этой фазе в результате исследующего поиска удалось получить точку 4 , отличную от точки 3 , то точку 2 переобозначим на 1 , а 4 на 2 и повторим поиск по образцу. В случае если не удаётся найти точку 4 , отличную от точки 3 , то точку 2 переобозначим на точку 1 и повторим 1-ю фазу алгоритма — исследующий поиск.

Алгоритм:

Шаг 1. Определить:

начальную точку $x^{(0)}$

приращения $\Delta^i, i = 1, 2, 3, \dots, N$,

коэффициент уменьшения шага $\alpha > 1$,

параметр окончания поиска $\varepsilon > 0$.

Шаг 2. Провести исследующий поиск.

Шаг 3. Был ли исследующий поиск удачным (найдена ли точка с меньшим значением целевой функции)?

Да: перейти к шагу 5.

Нет: продолжать.

Шаг 4. Проверка на окончание поиска. Выполняется ли неравенство $\|\Delta x\| < \varepsilon$?

Да: прекратить поиск; текущая точка аппроксимирует точку оптимума x^* .

Нет: уменьшить приращения по формуле

$$\Delta^i = \Delta^i / \alpha, i = 1, 2, 3, \dots, N,$$

Перейти к шагу 2.

Шаг 5. Провести поиск по образцу:

$$x^{(k+1)} = x^{(k)} + (x^{(k)} - x^{(k-1)}).$$

Шаг 6. Провести исследующий поиск, используя $x^{(k+1)}$ в качестве базовой точки; пусть $x^{(k+1)}$ — полученная в результате точка.

Шаг 7. Выполняется ли неравенство $f(x^{(k+1)}) < f(x^{(k)})$?

Да: положить $x^{(k-1)} = x^{(k)}$, $x^{(k)} = x^{(k+1)}$.

Перейти к шагу 5.

Нет: перейти к шагу 4.

2. Руководство оператора

2.1 Назначение программы

Данная программа предназначена для нахождения минимума функции методом Хука-Дживса

Входными данными являются: целевая функция, начальная точка, приращение, точность вычислений, шаг для исследующего поиска, коэффициент усиления для поиска по образцу.

Выходными данными является отчет, содержащий все шаги данного метода.

- 1) Ввод данных пользователем. Указанная функция позволяет пользователю вручную ввести данные. Доступно пользователю сразу после запуска web страницы.
- 2) Загрузка исходных данных. Данная функция позволяет загрузить из файла исходные данные для поиска минимума. После загрузки данные могут быть отредактированы без изменения исходного файла с данными. Доступно пользователю сразу после запуска web страницы.
- 3) Поиск минимума функции с подробным описанием шагов выполнения программы. Позволяет пользователю найти минимум функции с помощью метода Хука-Дживса. Доступно пользователю после ввода всех необходимых данных или после загрузки исходных данных из текстового файла.
- 4) Сохранение результата работы программы. Дает пользователю возможность сохранить результат работы программы в текстовый файл.

Приложение реализовано с помощью HTML, css, javascript.

2.2 Условия выполнения программы

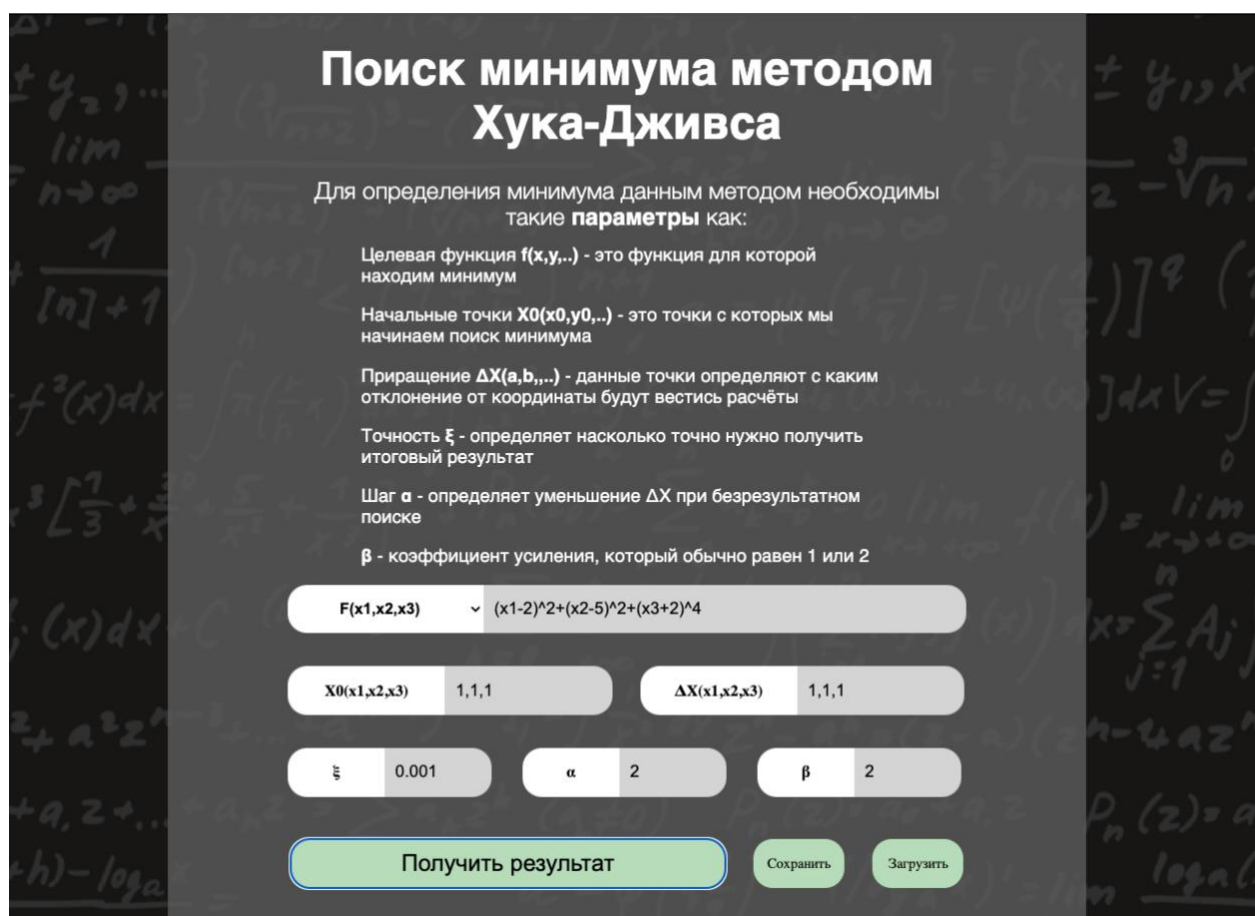
Для запуска web страницы необходимо иметь на компьютере браузер (Google Chrome, Opera, Firefox, Safari, Яндекс, Internet Explorer и др.)

Минимальные требования для работы браузера «Яндекс»:

1. Windows 7, Windows 8, Windows 8.1, Windows 10 или более поздней версии.
2. Процессор Pentium 4 1 GHz
3. 512 МБ свободной оперативной памяти;
4. 400 МБ свободной памяти на жёстком диске;
5. Мышь, клавиатура и Монитор;

2.3 Выполнение программы

Интерфейс программы представлен на рисунке 1.



Поиск минимума методом Хука-Дживса

Для определения минимума данным методом необходимы такие **параметры** как:

Целевая функция $f(x, y, \dots)$ - это функция для которой находим минимум

Начальные точки $X_0(x_0, y_0, \dots)$ - это точки с которых мы начинаем поиск минимума

Приращение $\Delta X(a, b, \dots)$ - данные точки определяют с каким отклонением от координаты будут вестись расчёты

Точность ξ - определяет насколько точно нужно получить итоговый результат

Шаг α - определяет уменьшение ΔX при безрезультатном поиске

β - коэффициент усиления, который обычно равен 1 или 2

$F(x_1, x_2, x_3)$ \vee $(x_1-2)^2 + (x_2-5)^2 + (x_3+2)^4$

$X_0(x_1, x_2, x_3)$ $1, 1, 1$ $\Delta X(x_1, x_2, x_3)$ $1, 1, 1$

ξ 0.001 α 2 β 2

Получить результат **Сохранить** **Загрузить**

Рисунок 1 – Интерфейс программы

Заголовок поля « $F(x_1, x_2, x_3)$ » позволяет выбрать количество переменных в функции.

Кнопка «Загрузить файл» позволяет загрузить функцию и ее параметры из файла.

Кнопка «Сохранить файл» позволяет сохранить введенные данные в файл.

Поле « $F(x_1, x_2, x_3)$ » позволяет указать функцию для которой необходимо найти корень.

Поле « $X_0(x_1, x_2, x_3)$ » позволяют указать начальную точку метода Хука-Дживса.

Поле « $\Delta X(x_1, x_2, x_3)$ » позволяет указать приращение для метода Хука-Дживса.

Поле « ξ » позволяет указать точность расчёта для метода Хука-Дживса.

Поле « α » позволяет указать уменьшение приращения.

Поле « β » позволяет указать коэффициент усиления для метода Хука-Дживса.

Сценарий:

- 1) Пользователь запускает web страницу.
- 2) Вписывает значения в поля « $F(x_1, x_2, x_3)$ », « $X_0(x_1, x_2, x_3)$ », « $\Delta X(x_1, x_2, x_3)$ », « ξ », « α », « β »
- 3) Пользователь нажимает на кнопку «Получить результат». Ниже кнопки появляется результат нахождения минимума, расписанный по шагам.
- 4) При необходимости пользователь может сохранить результат нажатием кнопки «Сохранить результат (.txt)».
- 5) Для повторного использования достаточно изменить необходимые данные в полях и снова нажать на кнопку «Получить результат».

2.4 Сообщение оператору

В данном разделе приведены сообщения об ошибках и отчет результата программы. Вывод данных сообщений происходит в дополнительном всплывающем окне.

Ошибки

- 1) «Ошибка ввода данных (подробная ошибка).» - значит, что пользователь не корректно ввёл данные.
- 2) «Ошибка сервера» - значит, что на серверной стороне произошла ошибка.

3. Результаты тестирования программы

На рисунке 2 представлен интерфейс программы после запуска.

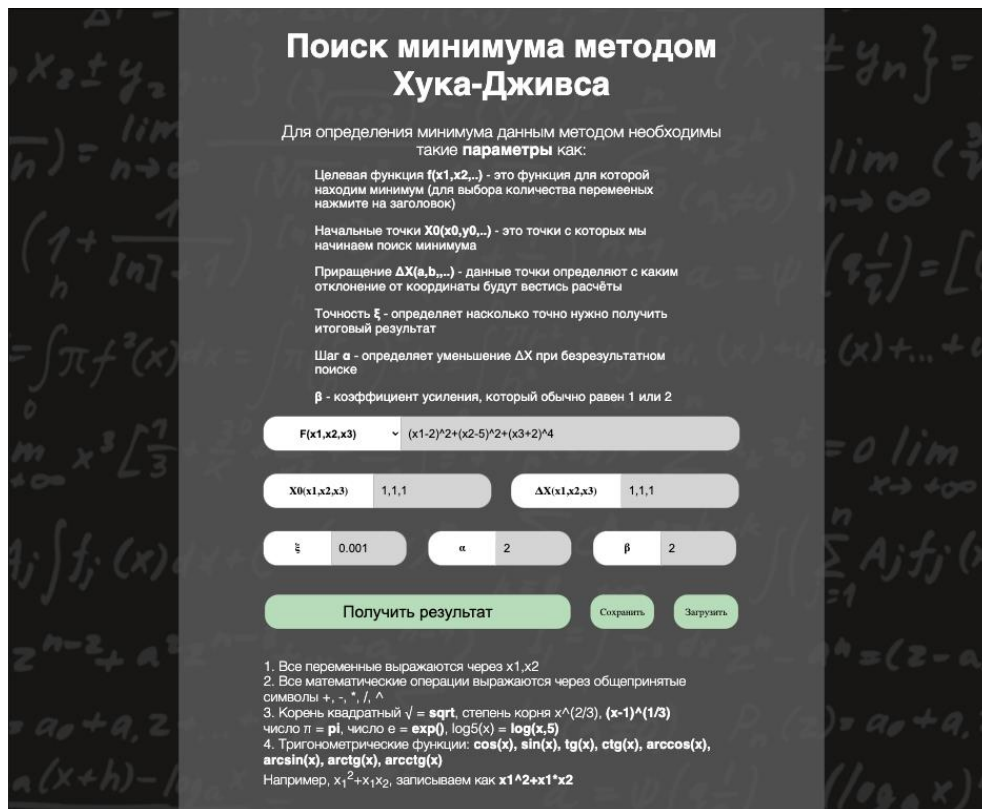


Рисунок 2 – Интерфейс программы после запуска

На рисунке 3 представлен ввод данных из текстового файла.

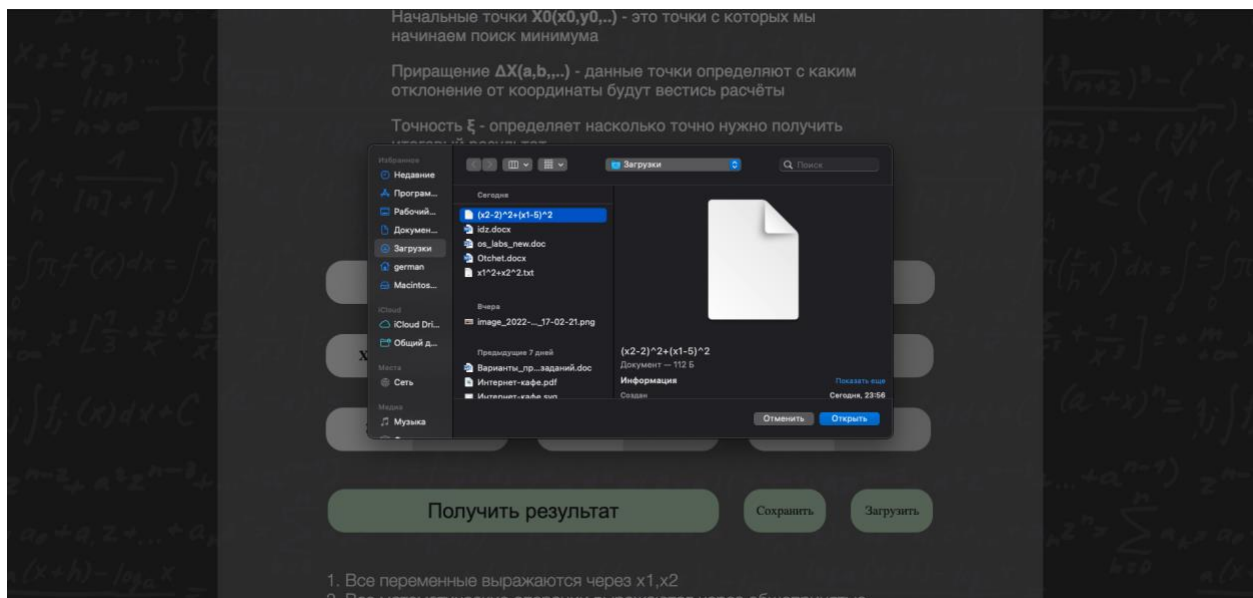


Рисунок 3 – Ввод данных из файла

На рисунке 4 представлен результат ввода данных из текстового файла.

Приращение $\Delta X(a,b,...)$ - данные точки определяют с каким отклонением от координаты будут вестись расчёты

Точность ξ - определяет насколько точно нужно получить итоговый результат

Шаг α - определяет уменьшение ΔX при безрезультатном поиске

β - коэффициент усиления, который обычно равен 1 или 2

$F(x1,x2)$ \downarrow $(x2-2)^2+(x1-5)^2$

$X0(x1,x2)$ $1,1$ $\Delta X(x1,x2)$ $1,1$

ξ 0.001 α 2 β 2

Получить результат Сохранить Загрузить

Рисунок 4 – Результат ввода данных из файла

На рисунке 5 представлен результат расчета программы.

Например, $x_1^2 + x_1 x_2$, записываем как $x1^2 + x1 * x2$

Временная базисная точка (11,5)

Итерация номер: 3
Исследующий поиск
Временная базисная точка (1,1)
Исследующий поиск был удачный
Новая базисная точка: (10,6)
Значение функции в временной базисной точке $f(1,1) = 45$
Значение функции в новой базисной точке $f(10,6) = 41$

Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке
Исследующий поиск
Временная базисная точка (1,1)
Новая базисная точка: (5,2)
Значение функции в временной базисной точке $f(1,1) = 1$
Значение функции в новой базисной точке $f(5,2) = 0$

Поиск по образцу
Временная базисная точка (5,0)

Итерация номер: 4
Исследующий поиск
Временная базисная точка (1,1)
Исследующий поиск был удачный
Новая базисная точка: (5,1)
Значение функции в временной базисной точке $f(1,1) = 4$
Значение функции в новой базисной точке $f(5,1) = 1$

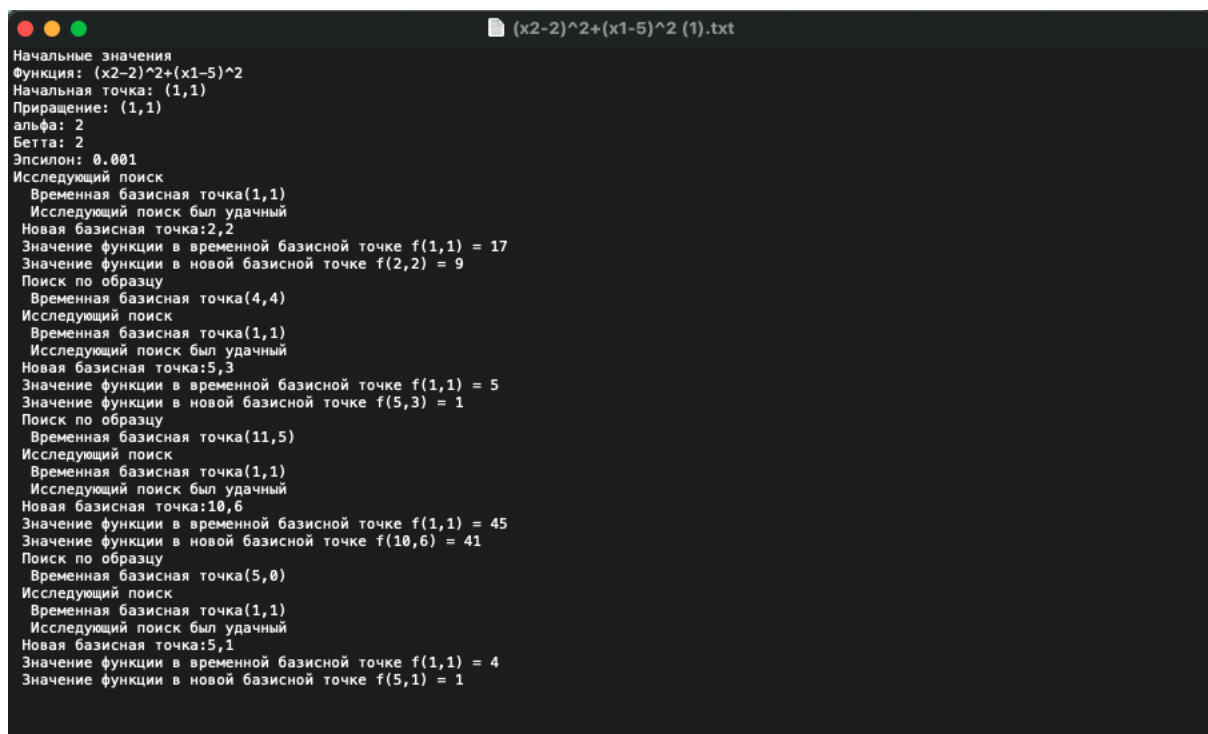
Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке
Исследующий поиск
Временная базисная точка (1,1)
Новая базисная точка: (5,2)
Значение функции в временной базисной точке $f(1,1) = 0$
Значение функции в новой базисной точке $f(5,2) = 0$

Минимум функции: [5,2]

Сохранить результат

Рисунок 5 – Результат расчета программы

На рисунке 6 представлено сохранение результата расчета программы в файл с помощью приложения.



```

(x2-2)^2+(x1-5)^2 (1).txt
Начальные значения
Функция: (x2-2)^2+(x1-5)^2
Начальная точка: (1,1)
Приращение: (1,1)
альфа: 2
Бетта: 2
Эпсилон: 0.001
Исследующий поиск
  Временная базисная точка(1,1)
  Исследующий поиск был удачный
  Новая базисная точка:2,2
  Значение функции в временной базисной точке f(1,1) = 17
  Значение функции в новой базисной точке f(2,2) = 9
  Поиск по образцу
  Временная базисная точка(4,4)
  Исследующий поиск
  Временная базисная точка(1,1)
  Исследующий поиск был удачный
  Новая базисная точка:5,3
  Значение функции в временной базисной точке f(1,1) = 5
  Значение функции в новой базисной точке f(5,3) = 1
  Поиск по образцу
  Временная базисная точка(11,5)
  Исследующий поиск
  Временная базисная точка(1,1)
  Исследующий поиск был удачный
  Новая базисная точка:10,6
  Значение функции в временной базисной точке f(1,1) = 45
  Значение функции в новой базисной точке f(10,6) = 41
  Поиск по образцу
  Временная базисная точка(5,0)
  Исследующий поиск
  Временная базисная точка(1,1)
  Исследующий поиск был удачный
  Новая базисная точка:5,1
  Значение функции в временной базисной точке f(1,1) = 4
  Значение функции в новой базисной точке f(5,1) = 1

```

Рисунок 6 – Сохранение результата расчета программы в файл с помощью приложения

Сравним результаты, полученные с помощью созданного приложения, с результатами приложения из первой лабораторной работы. В таблице 1 представлены результаты данного сравнения.

Таблица 1 – Результаты сравнения

Метод Хука-Дживса (программа из второй лабораторной работы)	Метод Хука-Дживса (созданное приложение)
<p>Оптимизация методом Хука - Дживса</p> <p>/-----/</p> <p>Целевая функция: $(x_2-2)^2+(x_1-5)^2$</p> <p>Начальная точка:</p> <p>$x_{1_0}=1$</p> <p>$x_{2_0}=1$</p> <p>Точность вычислений: 0,001</p> <p>Начальный шаг: 1</p> <p>/-----/</p> <p>Итерация № 1</p> <p>Текущее приближение к X_{opt}:</p> <p>$x_{1_k}=2$</p> <p>$x_{2_k}=2$</p> <p>Значение функции в X_k:</p> <p>$F(X_k)=9$</p> <p>/-----/</p> <p>Итерация № 2</p> <p>Текущее приближение к X_{opt}:</p> <p>$x_{1_k}=3$</p> <p>$x_{2_k}=3$</p> <p>Значение функции в X_k:</p> <p>$F(X_k)=5$</p> <p>/-----/</p> <p>Итерация № 3</p> <p>Текущее приближение к X_{opt}:</p> <p>$x_{1_k}=4$</p> <p>$x_{2_k}=4$</p> <p>Значение функции в X_k:</p> <p>$F(X_k)=5$</p> <p>/-----/</p> <p>Итерация № 4</p> <p>Текущее приближение к X_{opt}:</p> <p>$x_{1_k}=5$</p> <p>$x_{2_k}=3$</p> <p>Значение функции в X_k:</p> <p>$F(X_k)=1$</p> <p>/-----/</p> <p>Итерация № 5</p> <p>Текущее приближение к X_{opt}:</p> <p>$x_{1_k}=7$</p>	<p>Начальные значения</p> <p>Функция: $(x_2-2)^2+(x_1-5)^2$</p> <p>Начальная точка: (1,1)</p> <p>Приращение: (1,1)</p> <p>альфа: 2</p> <p>Бетта: 2</p> <p>Эпсилон: 0.001</p> <p>Итерация номер: 1</p> <p>Исследующий поиск</p> <p>Временная базисная точка (1,1)</p> <p>Исследующий поиск был удачный</p> <p>Новая базисная точка: (2,2)</p> <p>Значение функции в временной базисной точке $f(1,1)$ = 17</p> <p>Значение функции в новой базисной точке $f(2,2) = 9$</p> <p>Поиск по образцу</p> <p>Временная базисная точка (4,4)</p> <p>Итерация номер: 2</p> <p>Исследующий поиск</p> <p>Временная базисная точка (1,1)</p> <p>Исследующий поиск был удачный</p> <p>Новая базисная точка: (5,3)</p> <p>Значение функции в временной базисной точке $f(1,1)$ = 5</p> <p>Значение функции в новой базисной точке $f(5,3) = 1$</p> <p>Поиск по образцу</p> <p>Временная базисная точка (11,5)</p> <p>Итерация номер: 3</p> <p>Исследующий поиск</p> <p>Временная базисная точка (1,1)</p> <p>Исследующий поиск был удачный</p> <p>Новая базисная точка: (10,6)</p> <p>Значение функции в временной базисной точке $f(1,1)$ = 45</p> <p>Значение функции в новой базисной точке $f(10,6) = 41$</p>

<p>$x2_k = 3$</p> <p>Значение функции в X_k:</p> <p>$F(X_k) = 5$</p> <p>/-----/</p> <p>Итерация № 6</p> <p>Текущее приближение к X_{opt}:</p> <p>$x1_k = 6$</p> <p>$x2_k = 2$</p> <p>Значение функции в X_k:</p> <p>$F(X_k) = 1$</p> <p>/-----/</p> <p>Итерация № 7</p> <p>Текущее приближение к X_{opt}:</p> <p>$x1_k = 5,5$</p> <p>$x2_k = 2$</p> <p>Значение функции в X_k:</p> <p>$F(X_k) = 0,25$</p> <p>/-----/</p> <p>Итерация № 8</p> <p>Текущее приближение к X_{opt}:</p> <p>$x1_k = 6$</p> <p>$x2_k = 1$</p> <p>Значение функции в X_k:</p> <p>$F(X_k) = 2$</p> <p>/-----/</p> <p>Итерация № 9</p> <p>Текущее приближение к X_{opt}:</p> <p>$x1_k = 5,5$</p> <p>$x2_k = 1,5$</p> <p>Значение функции в X_k:</p> <p>$F(X_k) = 0,5$</p> <p>/-----/</p> <p>Итерация № 10</p> <p>Текущее приближение к X_{opt}:</p> <p>$x1_k = 5,25$</p> <p>$x2_k = 1,75$</p> <p>Значение функции в X_k:</p> <p>$F(X_k) = 0,125$</p> <p>/-----/</p> <p>Итерация № 11</p> <p>Текущее приближение к X_{opt}:</p> <p>$x1_k = 5$</p> <p>$x2_k = 1,5$</p> <p>Значение функции в X_k:</p> <p>$F(X_k) = 0,25$</p> <p>/-----/</p> <p>Итерация № 12</p> <p>Текущее приближение к X_{opt}:</p> <p>$x1_k = 5$</p> <p>$x2_k = 1,75$</p>	<p>Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке</p> <p>Исследующий поиск</p> <p>Временная базисная точка (1,1)</p> <p>Новая базисная точка: (5,2)</p> <p>Значение функции в временной базисной точке $f(1,1) = 1$</p> <p>Значение функции в новой базисной точке $f(5,2) = 0$</p> <p>Поиск по образцу</p> <p>Временная базисная точка (5,0)</p> <p>Итерация номер: 4</p> <p>Исследующий поиск</p> <p>Временная базисная точка (1,1)</p> <p>Исследующий поиск был удачный</p> <p>Новая базисная точка: (5,1)</p> <p>Значение функции в временной базисной точке $f(1,1) = 4$</p> <p>Значение функции в новой базисной точке $f(5,1) = 1$</p> <p>Исследующий поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся к старой базисной точке</p> <p>Исследующий поиск</p> <p>Временная базисная точка (1,1)</p> <p>Новая базисная точка: (5,2)</p> <p>Значение функции в временной базисной точке $f(1,1) = 0$</p> <p>Значение функции в новой базисной точке $f(5,2) = 0$</p> <p>Минимум функции: [5,2]</p>
---	--

Значение функции в X_k :

$$F(X_k) = 0,0625$$

/-----/

Итерация № 13

Текущее приближение к X_{opt} :

$$x1_k = 4,75$$

$$x2_k = 1,75$$

Значение функции в X_k :

$$F(X_k) = 0,125$$

/-----/

Итерация № 14

Текущее приближение к X_{opt} :

$$x1_k = 5$$

$$x2_k = 2$$

Значение функции в X_k :

$$F(X_k) = 0$$

/-----/

Итерация № 15

Текущее приближение к X_{opt} :

$$x1_k = 5$$

$$x2_k = 2,25$$

Значение функции в X_k :

$$F(X_k) = 0,0625$$

/-----/

Итерация № 16

Текущее приближение к X_{opt} :

$$x1_k = 5$$

$$x2_k = 2$$

Значение функции в X_k :

$$F(X_k) = 0$$

/-----/

Итерация № 17

Текущее приближение к X_{opt} :

$$x1_k = 5$$

$$x2_k = 2$$

Значение функции в X_k :

$$F(X_k) = 0$$

/-----/

Итерация № 18

Текущее приближение к X_{opt} :

$$x1_k = 5$$

$$x2_k = 2$$

Значение функции в X_k :

$$F(X_k) = 0$$

/-----/

Итерация № 19

Текущее приближение к X_{opt} :

$$x1_k = 5$$

$$x2_k = 2$$

Значение функции в X_k :

<p>$F(X_k) = 0$ /-----/ Итерация № 20 Текущее приближение к X_{opt}: $x1_k = 5$ $x2_k = 2$ Значение функции в X_k: $F(X_k) = 0$ /-----/ Итерация № 21 Текущее приближение к X_{opt}: $x1_k = 5$ $x2_k = 2$ Значение функции в X_k: $F(X_k) = 0$ /-----/ Итерация № 22 Текущее приближение к X_{opt}: $x1_k = 5$ $x2_k = 2$ Значение функции в X_k: $F(X_k) = 0$ /-----/ Итерация № 23 Текущее приближение к X_{opt}: $x1_k = 5$ $x2_k = 2$ Значение функции в X_k: $F(X_k) = 0$ /-----/ Итерация № 24 Текущее приближение к X_{opt}: $x1_k = 5$ $x2_k = 2$ Значение функции в X_k: $F(X_k) = 0$ /-----/ Итерация № 25 Текущее приближение к X_{opt}: $x1_k = 5$ $x2_k = 2$ Значение функции в X_k: $F(X_k) = 0$ /-----/ Итерация № 26 Текущее приближение к X_{opt}: $x1_k = 5$ $x2_k = 2$ Значение функции в X_k: $F(X_k) = 0$</p>	
---	--

<pre> /-----/ Итерация № 27 Текущее приближение к X_opt: x1_k= 5 x2_k= 2 Значение функции в X_k: F(X_k)= 0 /-----/ Итерация № 28 Текущее приближение к X_opt: x1_k= 5 x2_k= 2 Значение функции в X_k: F(X_k)= 0 /-----/ Итерация № 29 Текущее приближение к X_opt: x1_k= 5 x2_k= 2 Значение функции в X_k: F(X_k)= 0 /-----/ Итерация № 30 Текущее приближение к X_opt: x1_k= 5 x2_k= 2 Значение функции в X_k: F(X_k)= 0 /-----/ Оптимум найден на итерации: 30 Точка оптимума X_opt: x1_opt= 5 x2_opt= 2 Значение функции в X_opt: F(X_opt)= 0 Точность: 7,62939453125E-6 Длина шага: 7,62939453125E-6 </pre>	
---	--

4. Выводы по результатам сравнения

Согласно полученным результатам, можно сделать вывод, что созданное приложение производит правильные вычисления.

Приложения А

Алгоритм программы поиска методом Хука-Дживса на языке javascript

```
const math = require('mathjs');
```

```
const parser = math.parser();
```

```
class Huk {
```

```
  input_data = {
```

```
  }
```

```
  resES = [
```

```
    {
```

```
      flag: 1,
```

```
    }
```

```
  ];
```

```
  export_data = {}
```

```
  valueFunc(x) {
```

```
    try {
```

```
      let arg = "f(" + x.join(",") + ")";
```

```
      parser.evaluate("f"+this.input_data.type_func + ' = ' +
```

```
this.input_data.function)
```

```
      return (parser.evaluate(arg))
```

```
    } catch (e) {
```

```
      throw "Проверьте корректность введенных данных ( "+e+" )";
```

```
      console.log(e);
```

```
    }
```

```
}
```

```
reduceH(h) {  
  let flag = 0, i;  
  for (i = 0; i < h.length; i++) {  
    if (h[i] > this.input_data.epsilant) {  
      h[i] /= this.input_data.alpha;  
      flag = 1;  
    }  
  }  
  return {  
    flag: flag,  
    h: h,  
  };  
}
```

```
exploratory_search(x0, h) {  
  let time_base_point = x0.slice(0), i, new_base_point, j, flag = 1, z_tbp,  
  z_x0;  
  
  let new_h = {}  
  for (j = 0; flag === 1; j++) {  
    console.log("h= " + h);  
    for (i = 0; i < x0.length; i++) {  
      time_base_point[i] += h[i];  
      z_tbp = this.valueFunc(time_base_point);  
      z_x0 = this.valueFunc(x0);  
      if (z_tbp < z_x0) {  
        continue;  
      } else {  
        time_base_point[i] -= 2 * h[i];  
      }  
    }  
  }  
}
```

```

        z_tbp = this.valueFunc(time_base_point);
        z_x0 = this.valueFunc(x0);
        if (z_tbp < z_x0) {
            continue;
        } else {
            time_base_point[i] += h[i];
        }
    }
}

if (z_tbp < z_x0) {
    new_base_point = time_base_point.slice(0);
    time_base_point = x0.slice(0);
    break;
} else {
    new_h = this.reduceH(h.slice(0));
    h = new_h.h;
    if (new_h.flag === 0) {
        flag = new_h.flag;
        new_base_point = x0;
        break;
    }
}

return {
    time_base_point,
    new_base_point,
    flag,
}
}

```

```

search_obr(x1, x0, b) {
  try{
    let i, x2 = [];
    for (i = 0; i < this.input_data.x0.length; i++) {
      x2[i] = x1[i] + b * (x1[i] - x0[i]);
    }
    return x2;
  }
  catch (e){
    throw "Ошибка сервера";
  }
}

```

```

main(req) {
  this.export_data = {};
  try {
    this.export_data = {
      request_params: [
        {
          iteration: 0,
          initial_data: {},
          ES: {
            h: [],
            time_base_point: this.input_data.x0,
            f_NBP: 0,
            f_TBP: 0,
            new_base_point: this.input_data.x0,
          },
          SO: {

```

```

        new_base_point: this.input_data.x0,
        f_NBP: 0,
        f_TBP: 0,
        time_base_point: this.input_data.x0,
    },
}
]
}
this.input_data = req;
this.export_data.request_params[0].initial_data.function =
req.function;
this.export_data.request_params[0].initial_data.x0 = req.x0;
this.export_data.request_params[0].initial_data.h = req.h;
this.export_data.request_params[0].initial_data.alpha = req.alpha;
this.export_data.request_params[0].initial_data.betta = req.betta;
this.export_data.request_params[0].initial_data.epsilon =
req.epsilon;
this.resES[0].time_base_point = this.input_data.x0;
this.resES[0].new_base_point = this.input_data.x0;
let flag = 0;
console.log("      H = " + this.input_data.h + ", Альфа = " +
this.input_data.alpha + ", Бетта = " + this.input_data.betta);
console.log("Начальная базисная точка: " + this.input_data.x0);
console.log("Номер итерации " + 1);
this.export_data.request_params[1] = {
    iteration: 1,
    ES: {},
    SO: {},
};

```

```

        this.resES[1] =
this.exploratory_search(this.resES[0].time_base_point.slice(0),
this.input_data.h.slice(0));

        this.export_data.request_params[1].ES.time_base_point =
this.resES[0].time_base_point;

        this.export_data.request_params[1].ES.new_base_point =
this.resES[1].new_base_point;

        this.export_data.request_params[1].ES.f_TBP =
this.valueFunc(this.resES[0].time_base_point);

        this.export_data.request_params[1].ES.f_NBP =
this.valueFunc(this.resES[1].new_base_point);

        console.log("Новая базисная точка: " +
this.resES[1].new_base_point)
        if (this.resES[0].flag === 0) {
            console.log("Минимум функции =" +
this.resES[i].new_base_point);
            flag = 1;
        }
        for (let i = 2; flag === 0; i++) {
            this.resES[i] = { };
            this.export_data.request_params[i] = {
                iteration: i,
                ES: { },
                SO: { },
                msg: "",
                DES: { }
            };
            console.log("Поиск по образцу");
            this.resES[i].time_base_point = this.search_obr(this.resES[i -
1].new_base_point, this.resES[i - 2].new_base_point, this.input_data.betta);

```

```

        this.export_data.request_params[i - 1].SO.new_base_point =
this.resES[1].new_base_point;

        this.export_data.request_params[i - 1].SO.time_base_point =
this.resES[i].time_base_point;

        console.log("Временная базисная точка: " +
this.resES[i].time_base_point)

        console.log("Номер итерации " + i);
        console.log("Исследующий поиск");
        console.log("F(" + this.resES[i].time_base_point + ") = " +
this.valueFunc(this.resES[i].time_base_point));

        this.resES[i] =
this.exploratory_search(this.resES[i].time_base_point.slice(0),
this.input_data.h.slice(0));

        this.export_data.request_params[i].ES.time_base_point =
this.resES[1].time_base_point;

        this.export_data.request_params[i].ES.new_base_point =
this.resES[i].new_base_point;

        this.export_data.request_params[i].ES.f_TBP =
this.valueFunc(this.resES[i].time_base_point);

        this.export_data.request_params[i].ES.f_NBP =
this.valueFunc(this.resES[i].new_base_point);

        console.log("Новая базисная точка: " +
this.resES[i].new_base_point)

        if (this.resES[i].flag === 0) {
            console.log("Минимум функции = " +
this.resES[i].new_base_point);

            flag = 1;
            break;
        }

```

```

        if (this.valueFunc(this.resES[i].new_base_point) <
this.valueFunc(this.resES[i - 1].new_base_point)) {
            continue;
        } else {
            console.log(this.valueFunc(this.resES[i].new_base_point) + ">"
+ this.valueFunc(this.resES[i - 1].new_base_point))
            console.log("Исследующий поиск вокруг этой временной
базисной точки неудачен поэтому возвращаемся к старой базисной точке")
            this.export_data.request_params[i].msg = "Исследующий
поиск вокруг этой временной базисной точки неудачен поэтому возвращаемся
к старой базисной точке";
            this.resES[i].time_base_point = this.resES[i -
1].new_base_point.slice(0);
            console.log("Исследующий поиск");
            console.log("F(" + this.resES[i].time_base_point + ") = " +
this.valueFunc(this.resES[i].time_base_point));
            this.resES[i] =
this.exploratory_search(this.resES[i].time_base_point.slice(0),
this.input_data.h.slice(0));
            this.export_data.request_params[i].DES.time_base_point =
this.resES[1].time_base_point;
            this.export_data.request_params[i].DES.new_base_point =
this.resES[i].new_base_point;
            this.export_data.request_params[i].DES.f_TBP =
this.valueFunc(this.resES[i].time_base_point);
            this.export_data.request_params[i].DES.f_NBP =
this.valueFunc(this.resES[i].new_base_point);
            console.log("Новая базисная точка: " +
this.resES[i].new_base_point)
            if (this.resES[i].flag === 0) {

```



```

        console.log("Минимум функции =" +
this.resES[i].new_base_point);
        flag = 1;
        break;
    }

    }
}
this.export_data.request_params.forEach(rp => {
    if (Object.keys(rp.SO).length === 0) {
        delete rp.SO
    }
})

    return this.export_data;
}
catch (e){
    console.log(e)
    this.export_data.err=e;
    return this.export_data;
}
}
}

module.exports = Huk;

```