ГУАП

КАФЕДРА № 14

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Доцент, к.т.н. |  |  |  | С.А. Андронов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1 |
| «МЕТОДЫ ОДНОМЕРНОГО ПОИСКА» |
| по курсу: «Методы оптимизации» |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 1143М |  |  |  | С.О. Абабков |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2022

1. Общее задание и конкретный вариант

Цель работы: изучение основных алгоритмов, применяемых в задачах поиска минимума функции одного переменного и получение практических навыков программной реализации методов линейного поиска.

Язык программной реализации: Python

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант № п. п. | Функция, интервал, шаг поиска, число итераций, точность | Решение, Начальное приближение | Используемый метод поиска | Литература |
| 1 |  | x\*=0.357403 | Фибоначчи | [1] |

1. Математическая формулировка задачи и описание метода решения

Найти минимальное значение в интервале от 0 до 1.

В качестве метода решения задачи, предполагается использовать метод Фибоначчи. Данный метод применяется для поиска безусловного экстремума функции одной переменной. Он основан на последовательном сокращении интервала неопределённости, выбирая на каждом шаге две точки, причём одна из которых используется в качестве внутренней и для следующего интервала. Он назван в честь Фибоначчи, поскольку в нем реализована стратегия, обеспечивающая максимальное сокращение интервала неопределённости при заданном количестве вычислений функции, которая опирается на числа Фибоначчи. Числа Фибоначчи определяются по формуле:

1. Схема алгоритма решения задачи

Выписать несколько первых чисел последовательности Фибоначчи.

Шаг 1. Задать количество вычислений функции N. Поиск минимума заканчивается, когда длина текущего интервала неопределённости оказывается меньше установленной величины ()

Шаг 2. Для того чтобы найти требуемое число *N,* используем условие и числа Фибоначчи , где L-заданный интервал .

Шаг 3. Вычислить

Шаг 4. Вычислить *.* Данный шаг является началом итерационной части.

Шаг 5. Сравнить *.*

Если то положить:

Перейти к шагу 6.

Если то положить:

Шаг 6. Проверить условие окончания.

1. Анализ полученного решения на оптимальность (теоретический и практический)

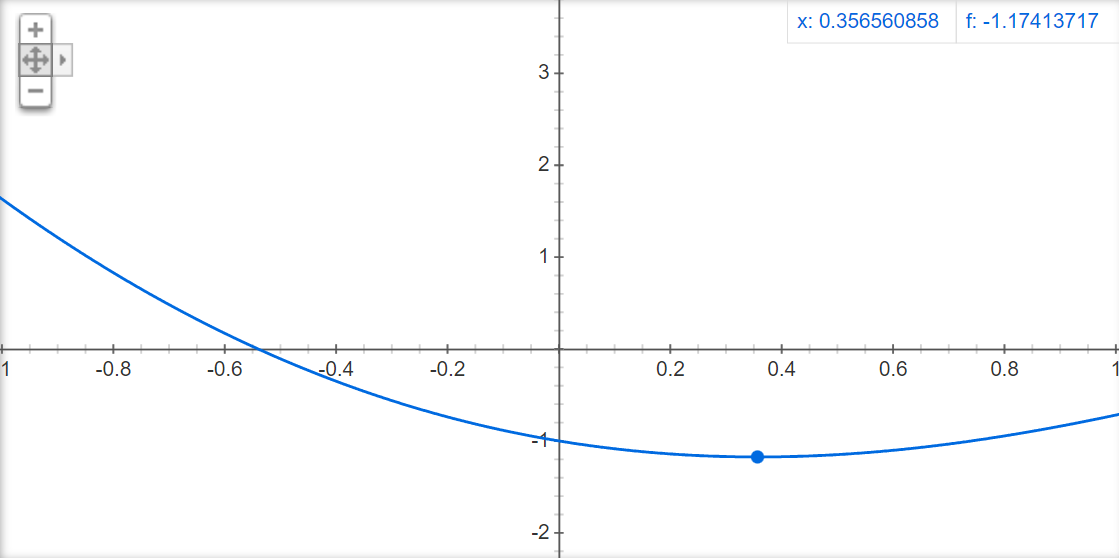


Рисунок 1 График заданной функции

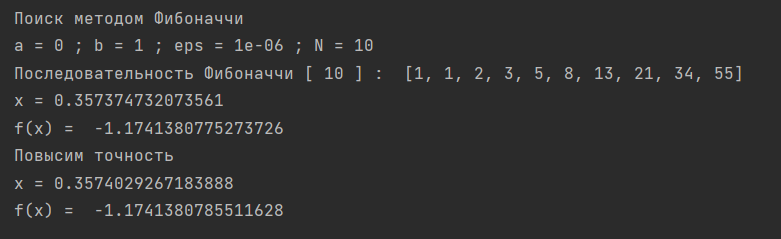


Рисунок 2 Практический результат N=10

Изменяя N, можно изменить и получившийся результат, к примеру на рисунке ниже, погрешность вычислений гораздо выше эпсилон

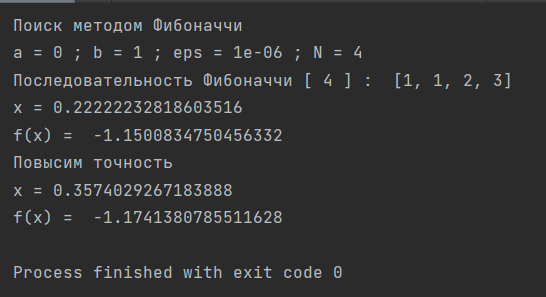
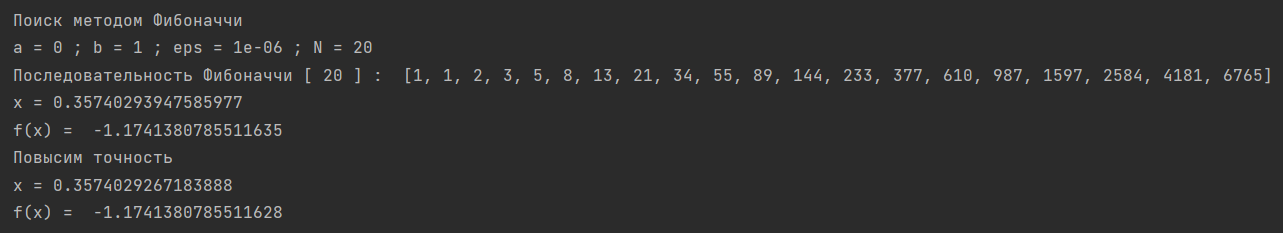


Рисунок 3 Результат для N=4

Увеличивая N, можно добиться более точных результатов:

Рисунок 4 Результат для N=20

Полученные результаты при округлении до 6 знака полностью совпадают с начальным приближением x\*=0.357403

Для анализа полученного решения, подставим значение.





Значение, полученное в результате работы моей программы оказалось меньше: f(x) = -1.1741380785511635, что свидетельствует о более высокой точности.

1. Вывод

В ходе выполнения лабораторной работы, ознакомился с методом Фибоначчи для поиска минимума функции одной переменной. Были изучены методический материалы, составлен алгоритм и написана программа, находящая минимум функции с помощью заданного метода.

1. Листинг программы

Программа написана на языке Python 3.7

**import** math

**def** func\_task**(**x**):**

**return** 2 **\*** math**.pow(**x**,** 2**)** **-** math**.**exp**(**x**)**

# Поиск последовательности Фибоначчи

**def** fib**(**input\_num**,** show**=**0**):**

n **=** **int(**input\_num**)**

fb **=** **[**1**,** 1**]**

**for** k **in** **range(**2**,** n**):**

fb**.**append**(**fb**[**k **-** 2**]** **+** fb**[**k **-** 1**])**

**if** show **==** 1**:**

**print(**'Последовательность Фибоначчи'**,** '['**,** input\_num**,** ']'**,** ': '**,** fb**)**

**else:**

**return** fb**[**input\_num**-**1**]**

**def** fibonacci**(**a**,** b**,** eps**,** N**):**

**print(**"Поиск методом Фибоначчи"**)**

**print(**'a ='**,** a**,** '; b ='**,** b**,** '; eps ='**,** eps**,** '; N ='**,** N**)**

a0 **=** a

b0 **=** b

k **=** 0

y0 **=** a0**+((**fib**(**N**-**2**)/**fib**(**N**))\*(**b0**-**a0**))**

z0 **=** a0**+((**fib**(**N**-**1**)/**fib**(**N**))\*(**b0**-**a0**))**

**while** **abs(**b0 **-** a0**)** **>** eps**:**

**if** func\_task**(**y0**)** **>** func\_task**(**z0**):**

a0 **=** y0

y0 **=** z0

z0 **=** a0**+((**fib**(**N**-**k**-**2**)/**fib**(**N**-**k**-**1**))\*(**b0**-**a0**))**

**else:**

b0**=**z0

z0**=**y0

y0**=**a0**+((**fib**(**N**-**k**-**3**)/**fib**(**N**-**k**-**1**))\*(**b0**-**a0**))**

**if** **++**k **>** 10e5**:**

**break**

**return** **(**a0 **+** b0**)** **/** 2

**def** gold**(**a**,** b**,** eps**):**

**print(**"Повысим точность"**)**

g **=** **(**math**.**sqrt**(**5**)** **-** 1.0**)** **/** 2

k **=** 0

a1 **=** a **+** **(**1 **-** g**)** **\*** **(**b **-** a**)**

b1 **=** a **+** g **\*** **(**b **-** a**)**

**while** **abs(**b **-** a**)** **>** eps**:**

**if** func\_task**(**a1**)** **>** func\_task**(**b1**):**

a **=** a1

a1 **=** b1

b1 **=** a **+** g **\*** **(**b **-** a**)**

**else:**

b **=** b1

b1 **=** a1

a1 **=** a **+** **(**1 **-** g**)** **\*** **(**b **-** a**)**

**if** **++**k **>** 10e5**:**

**break**

**return** **(**a **+** b**)** **/** 2

**if** \_\_name\_\_ **==** '\_\_main\_\_'**:**

**print(**'e = '**,** math**.**e**)**

a **=** 0

b **=** 1

N **=** 20

eps **=** 0.000001

x **=** fibonacci**(**a**,** b**,** eps**,** N**)**

fx **=** func\_task**(**x**)**

fib**(**N**,** 1**)**

**print(**'x ='**,** x**)**

**print(**'f(x) = '**,** fx**)**

#print('Округлим значение')

#print('x =', round(x, 6))

#print('f(x) =', round(fx, 6))

x **=** gold**(**a**,** b**,** eps**)**

fx **=** func\_task**(**x**)**

**print(**'x ='**,** x**)**

**print(**'f(x) = '**,** fx**)**