МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики»

ФАКУЛЬТЕТ ПИИКТ

ЛАБОРАТОРНАЯ РАБОТА № 1

по дисциплине

‘Вычислительная математика’

Вариант:

Аппроксимация сигмоидами

Выполнил:

Студент группы P3233

Хасаншин Марат Айратович

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2024

Описание численного метода:

Аппроксимация сигмоидами используется для получения новой функции, которую проще высчитывать и которая будет вести себя примерно как наша исходная.

Сигмоидом называется уравнение вида:

$$\sigma = \frac{1}{1 + e^{-b*(x-c)}}$$

Тогда воспользуемся методом наименьших квадратов, для нахождения таких коэффициентов b и c, при которых наш сигмоид будет похож на исходную функцию.

$$b = \frac{n * \sum_{i=0}^{n} x_i * y_i - \sum_{i=0}^{n} x_i * \sum_{i=0}^{n} y_i}{n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2}$$

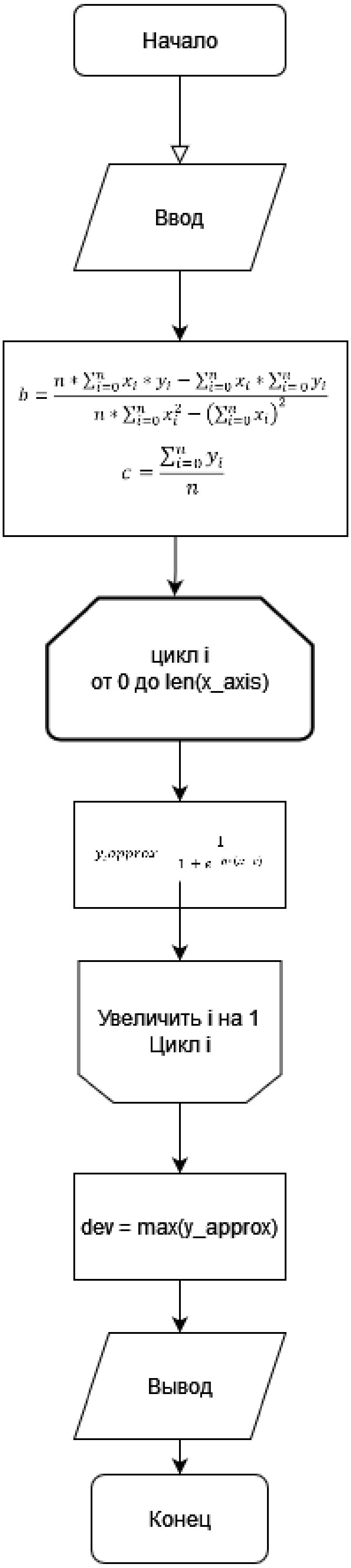
$$c = \frac{\sum_{i=0}^{n} y_i}{n}$$

, где n – количество значений x.

После этого высчитываем и возвращаем максимальную разницу между y и аппроксимацией.

$$y_i approx = \frac{1}{1 + e^{-b*(x-c)}}$$

Блок-схема:


Начало
Ввод
$b = \frac{n * \sum_{i=0}^{n} x_i * y_i - \sum_{i=0}^{n} x_i * \sum_{i=0}^{n} y_i}{n * \sum_{i=0}^{n} x_i^2 - \left(\sum_{i=0}^{n} x_i\right)^2}$
$c = \frac{\sum_{i=0}^{n} y_i}{n}$
цикл i
от 0 до len(x_axis)
Увеличить i на 1
Цикл i
dev = max(y_approx)
Вывод
Конец

Код:

```
def sigmoid(x, b, c):
    return 1 / (1 + math.exp(-b * (x - c)))

def fit_sigm(x_axis, y_axis):
    n = len(x_axis)
    x_square = sum(x ** 2 for x in x_axis)
    xy = sum(x * y for x, y in zip(x_axis, y_axis))
    delta = n * x_square - (sum(x_axis) ** 2)
    delta_b = n * xy - sum(x_axis) * sum(y_axis)
    b = delta_b / delta
    c = sum(y_axis) / n
    return b, c

def approximate_sigmoid(x_axis, y_axis):
    b_c = fit_sigm(x_axis, y_axis)
    y_approximate = [sigmoid(x, *b_c) for x in x_axis]
    dev = [abs(y - y_approx) for y, y_approx in zip(y_axis, y_approximate)]

    return max(dev)
```

Примеры работы программы:

№1:

f = 2x

```
3
1 2 3
2 4 6
5.880797077977882
```

№2:

f = 1/x

```
3
0.25 0.5 1
4 2 1
3.0004357111456317
```

№3:

$f = x^2 + x$

```
3
1 2 3
2 5 10
9.999976691442159
```

№4:

f $= x^2$

```
10
1 2 3 4 5 6 7 8 9 10
1 4 9 16 25 36 49 64 81 100
100.0
```

№5:

f = sin(x)

```
4
0 1 3 5
0 0.841 0.1411 -0.9589
1.1880769886312872
```

Вывод:

Тесты показывают погрешность в аппроксимации при разных функциях – линейных, тригонометрических, гиперболических.

В сравнение с другими методами можно выделить следующие минусы и плюсы:

Минусы:

- Погрешность высока для линейных и квадратичных функций.

Плюсы:

- Метод очень легко реализовать.
- Для гиперболических и тригонометрических функций малая погрешность.

Алгоритм может быть применен для любых функций, но может быть огромная погрешность, в зависимости от функции.

Алгоритмическая сложность метода – O(n), где n – число переменных.

Численная погрешность для линейных функций огромна, для гиперболических и тригонометрических - нет.