Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет ИТМО»

Факультет Программной инженерии и компьютерной техники

Лабораторная работа по

Вычислительной математике №1

Вариант «Аппроксимация по методу наименьших модулей»

Работу выполнила:

Касьяненко В.М.

Группа:

P3220

Санкт-Петербург,

2024

Описание численного метода.

Метод наименьших модулей (МНМ) – один из методов регрессионного анализа для оценки неизвестных величин по результатам измерений, содержащих случайные ошибки.
МНМ похож на метод наименьших квадратов (МНК). Отличие состоит в минимизации не суммы квадратов невязок, а (взвешенной) суммы их абсолютных значений.

$$\sum_{i=1}^{n} |y_i - f(x_i)|$$

В МНМ вычисляется вес функции

$$p_i = \sum_{i=1}^{n} \frac{1}{|y_i - b - k * x_i|} + \varepsilon,$$

где ε малая величина.
Коэффициенты:

$$k = \frac{sumPXY * sumP \; - (sumPX * sumPY)}{sumPX2 * sumP - sumPX * sumPX};$$

$$b = \frac{sumPY - k * sumPX}{sumP},$$

где

$$sumPXY = \sum_{i=1}^{n} (p_i * x_i * y_i)$$

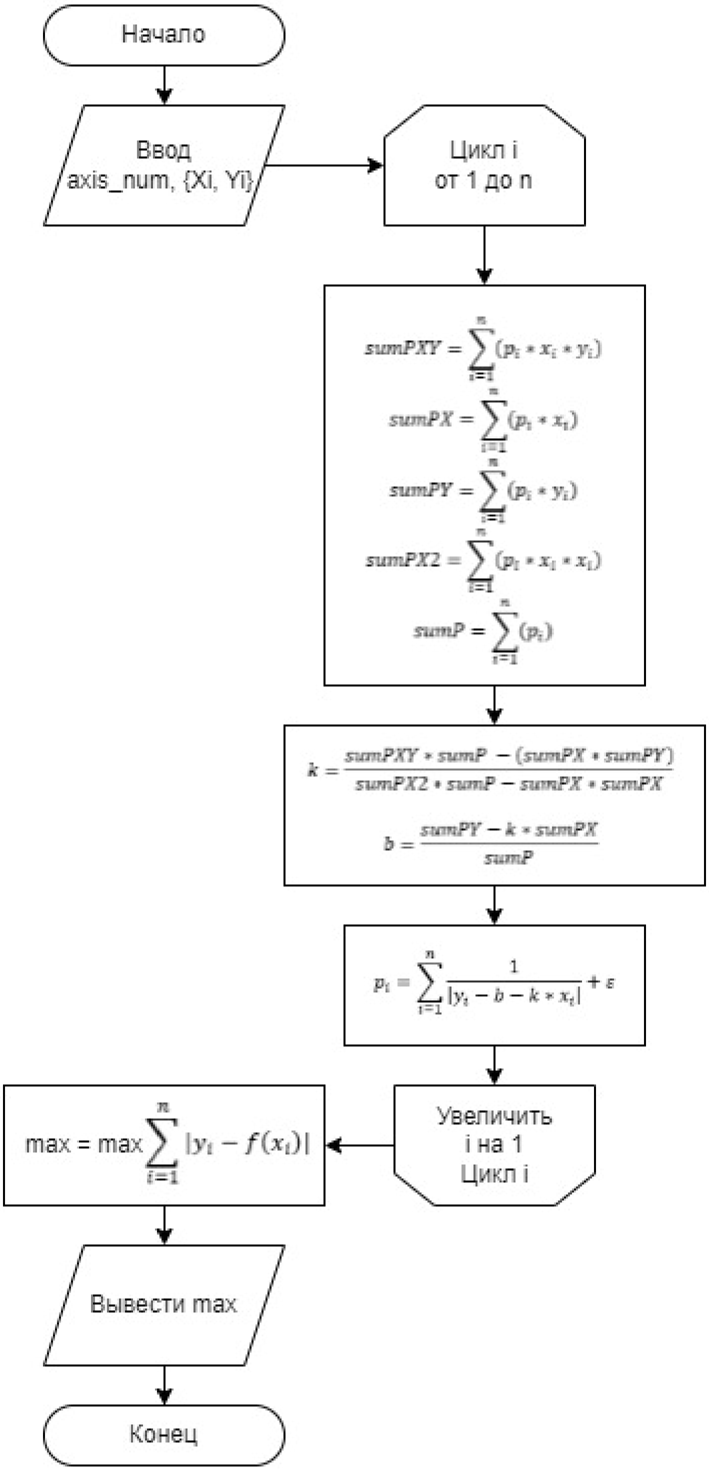
$$sumPX = \sum_{i=1}^{n} (p_i * x_i)$$

$$sumPY = \sum_{i=1}^{n} (p_i * y_i)$$

$$sumPX2 = \sum_{i=1}^{n} (p_i * x_i * x_i)$$

$$sumP = \sum_{i=1}^{n} (p_i)$$

Блок-схема


Начало
Ввод
axis_num, {Xi, Yi}
Цикл i
от 1 до n
$sumPXY = \sum_{i=1}^{n}(p_i * x_i * y_i)$
$sumPX = \sum_{i=1}^{n}(p_i * x_i)$
$sumPY = \sum_{i=1}^{n}(p_i * y_i)$
$sumPX2 = \sum_{i=1}^{n}(p_i * x_i * x_i)$
$sumP = \sum_{i=1}^{n}(p_i)$
$k = \frac{sumPXY * sumP - (sumPX * sumPY)}{sumPX2 * sumP - sumPX * sumPX}$
$b = \frac{sumPY - k * sumPX}{sumP}$
$p_i = \sum_{i=1}^{n}\frac{1}{|y_i - b - k * x_i|} + \varepsilon$
Увеличить
i на 1
Цикл i
max = max $\sum_{i=1}^{n}|y_i - f(x_i)|$
Вывести max
Конец

Код численного метода

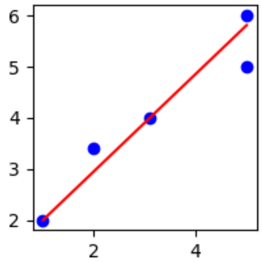
```
def weighted_abs_sum(x, y, w):
    return sum(w[i] * abs(y[i] - x[i]) for i in range(len(x)))

def approximate_linear_least_modules(x, y):
    n = len(x)
    weights = [1] * n
    epsilon = 1e-5
    max_iterations = 1000
    k = 0
    b = 0
    for _ in range(max_iterations):
        k_prev = k
        b_prev = b
        sumPXY = sum(weights[i] * x[i] * y[i] for i in range(n))
        sumPX = sum(weights[i] * x[i] for i in range(n))
        sumPY = sum(weights[i] * y[i] for i in range(n))
        sumPX2 = sum(weights[i] * x[i] * x[i] for i in range(n))
        sumP = sum(weights)
        k = (sumPXY * sumP - (sumPX * sumPY)) / (sumPX2 * sumP - sumPX * sumPX)
        b = (sumPY - k * sumPX) / sumP
        for i in range(n):
            weights[i] = 1 / (abs(y[i] - b - k * x[i]) + epsilon)
        if abs(k - k_prev) < epsilon and abs(b - b_prev) < epsilon:
            break

    max_deviation = max(abs(y[i] - (b + k * x[i])) for i in range(n))
    return max_deviation
```

Примеры работы программы

1)
Входные данные:
5
1 2 3.1 5 5
2 3.4 4 5 6

Выходные данные:
0.8095265736677311

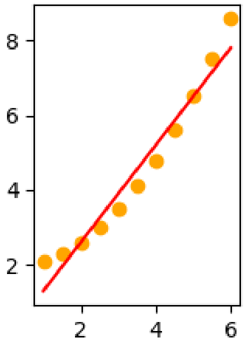
График:



2)
Входные данные:
11
1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0
2.1 2.3 2.6 3.0 3.5 4.1 4.8 5.6 6.5 7.5 8.6

Выходные данные:
0.8000172485912311

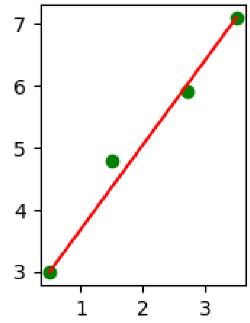
График:



3)
Входные данные:
4
0.5 1.5 2.7 3.5
3 4.8 5.9 7.1

Выходные данные:
0.43332731125467117

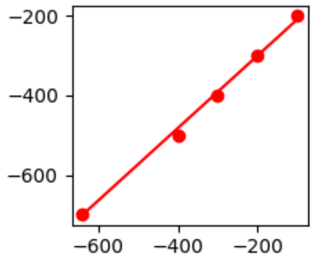
График:



4)
Входные данные:
5
-100 -200 -300 -400 -640
-200 -300 -400 -500 -700

Выходные данные:
18.181674508487106

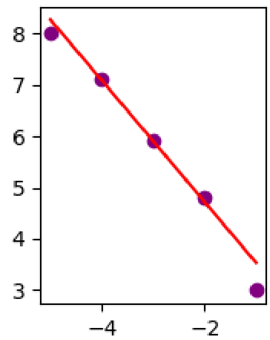
График:



5)
Входные данные:
5
-1 -2 -3 -4 -5
3 4.8 5.9 7.1 8

Выходные данные:
0.5367455808607158

График:



Выводы

В результате выполнения лабораторной работы была реализована линейная аппроксимация методом наименьших модулей на языке Python. Метод наименьших модулей (МНМ) является альтернативой методу наименьших квадратов (МНК) и используется для аппроксимации данных, особенно в случаях, когда имеются выбросы или сильные отклонения, которые могут сильно исказить результаты, если использовать МНК.
Результаты запуска реализованного метода на различных данных:

1. Нормальные данные: Если данные ведут себя хорошо, и линейная аппроксимация в целом подходит для описания данных, метод наименьших модулей может дает очень хорошие результаты, так как минимизирует суммы абсолютных значений модулей.
2. С выбросами: В случае наличия выбросов, метод наименьших модулей обычно работает лучше, чем МНК, так как он учитывает абсолютные отклонения, а не квадраты отклонений, что делает его менее чувствительным к выбросам.
3. Случайные данные: Если данные случайны и не имеют явной линейной зависимости, оба метода могут дать схожие результаты, но в случае наличия выбросов, МНМ может дать более устойчивую аппроксимацию.

Сравнение с методом наименьших квадратов (МНК) и методом наибольшего правдоподобия (MLE): МНМ менее чувствителен к выбросам, так как использует модуль отклонения вместо квадрата как МНК. Это позволяет ему лучше работать с данными, содержащими выбросы или нестандартные отклонения. MLE, в зависимости от формы функции правдоподобия, может быть более чувствительным к выбросам, особенно если выбросы искажают форму функции правдоподобия.
Таким образом, метод наименьших модулей полезен в случаях, когда есть выбросы в данных и необходима устойчивость к выбросам.
В общем случае сложность алгоритма O(n). Сложность итерационного процесса зависит от числа итераций до сходимости, которое может быть достигнуто за конечное число шагов, но в худшем случае может быть ограничено только максимальным числом итераций.
При выполнении алгоритма МНМ могут возникнуть численные ошибки из-за ограничений на максимальное число итераций. Также этот метод может иметь численные проблемы, связанные с неустойчивостью численного решения при неопределенностях в исходных данных.
Неустойчивость может привести к сильным колебаниям в решении или даже к сходимости к неправильному результату.