БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

**Лабораторная работа №3.1**

**Метод наименьших квадратов**

**Вариант 1**

Выполнил: Белоушко Степан,

2 курс 9 группа

Преподаватель:

Будник Анатолий Михайлович

**Условие:**

По условию дан отрезок [𝑎,𝑏]=[,], узлы находятся по формуле .

Функция:

Точки восстановления:

Методом наименьших квадратов построить аппроксимирующий многочлен степени m=n/2. Весовая функция в рассматриваемом пространстве равна 1.

Вычислить приближенные значения функции f(x) в точках x\*, x\*\* и x\*\*\*. Оценить погрешность.

**План решения:**

Строим таблицу дискретизации по правилу .

Элемент наилучшего приближения (ЭНП) будем искать в пространстве всех действительных квадратично интегрируемых по весу функций

ЭНП определим, как линейную комбинацию вида:

, где .

Для определения необходимо решить следующую систему линейных алгебраических уравнений: методом Гаусса решения СЛАУ

Погрешность приближения определяется среднеквадратичным отклонением, определяемым по формуле:

Истинная погрешность находится по формуле в точках .

**Листинг программы**

import math

import numpy as np

def f(x):

return 0.15 \* math.exp(x) + 0.85 \* math.sin(x)

def phi(C, x):

res = 0

for i in range(6):

res += C[i] \* x\*\*i

return res

def sampling(x0, h):

X = [x0 + h \* i for i in range(11)]

Y = [f(X[i]) for i in range(11)]

return X, Y

X, Y = sampling(0.15, 0.1)

print('X')

print(X)

print('Y')

print(Y)

def leastSquareMethod(X, Y):

h = 0.1

delta = 0

nodes = [0, 0, 0]

phiY = [0, 0, 0]

r = [0, 0, 0]

A = [[0] \* 6 for \_ in range(6)]

b = [0] \* 6

for l in range(6):

for k in range(6):

sum = 0

for i in range(11):

sum += X[i]\*\*(k + l)

A[l][k] = sum

sum = 0

for i in range(11):

sum += Y[i] \* X[i]\*\*l

b[l] = sum

C = np.linalg.inv(np.array(A)).dot(np.array(b))

nodes[0] = X[0] + (2/3) \* h

nodes[1] = X[5] + (1/2) \* h

nodes[2] = X[10] - (1/3) \* h

for i in range(3):

phiY[i] = phi(C, nodes[i])

for i in range(3):

r[i] = f(nodes[i]) - phi(C, nodes[i])

for i in range(11):

delta += (phi(C, X[i]) - Y[i])\*\*2

delta = math.sqrt(delta)

return C, phiY, delta, r

C, phiY, delta, r = leastSquareMethod(X, Y)

**Результаты**

1. Таблица дискретизации:

|  |  |
| --- | --- |
|  |  |
| 0.15 | 0.3012975490118018 |
| 0.25 | 0.4028971778695057 |
| 0.35 | 0.5043232686261222 |
| 0.45 | 0.6049675318180711 |
| 0.55 | 0.7042720972711696 |
| 0.65 | 0.801739569227718 |
| 0.75 | 0.8969429485117353 |
| 0.85 | 0.9895353721581475 |
| 0.95 | 1.0792596279683446 |
| 1.05 | 1.165957409464389 |
| 1.15 | 1.2495782856749078 |

1. Коэффициенты ЭНМ

|  |  |
| --- | --- |
|  | 0.15000571 |
|  | 0.99991829 |
|  | 0.07541632 |
|  | -0.11763577 |
|  | 0.00728761 |
|  | 0.00800044 |

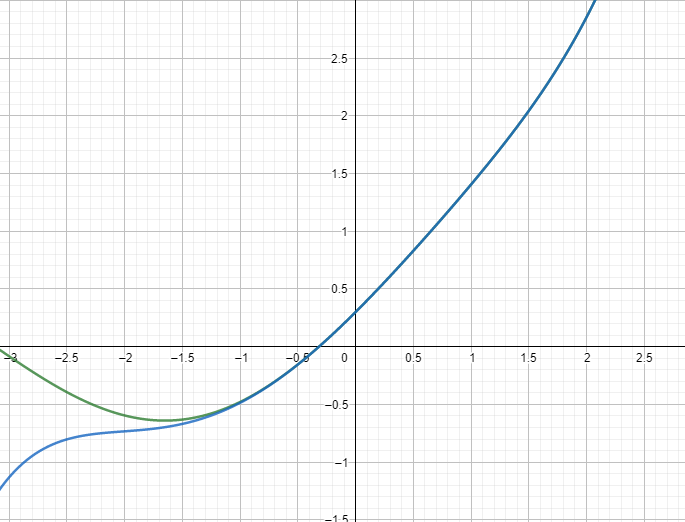
1. Значения в точках

|  |  |
| --- | --- |
|  | 0.36901842213444414 |
|  | 0.8496478273019252 |
|  | 1.2220447686504183 |

1. Истинные погрешности

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

**Вывод**



Интерполирование функции методом наименьших квадратов дало среднеквадратичное отклонение порядка 10-7. Исходя из графика, исходная функция и ее интерполяция почти неразличимы на графике, в частности и на отрезке .