РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

**Отчет по лабораторной работе №2**

ПО ДИСЦИПЛИНЕ ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ

02.04.02 — Фундаментальная информатика и информационные технологии

Выполнила Коняева Марина Александровна

Студент группы НФИбд-01-21

Студенческий билет №: 1032217044

Москва 2023

**Содержание**

Теоритическое введение и постановка задачи 5

Программный код 6

Вывод программы 8

Вывод пункты 5 и 6 лабораторной работы №1 9

Заключение 10

**Теоретическое введение и постановка задачи**

Пусть на отрезке [*a,b*] определена некоторая функция , однако полная информация о ней недоступна. Известны лишь ее значения в конечном числе точек этого отрезка, которые будем считать занумерованными в порядке возрастания:

Требуется по известным значениям «восстановить», хотя бы приближенно, исходную функцию , то есть построить на отрезке [*a,b*] функцию *F*(*x*), достаточно близкую к . Функцию *F*(*x*) принято называть интерполирующей, точки - узлами интерполяции.

Выберем некоторую систему функций , заданных на отрезке [*a,b*], и будем строить *F*(*x*) как их линейную комбинацию:

(3)

где числовые коэффициенты , подлежат определению согласно условиям (4)

Равенства (4) представляют собой систему линейных алгебраических уравнений относительно коэффициентов :

(5)

или в развернутом виде:

Изображение выглядит как текст, Шрифт, рукописный текст, типография

Автоматически созданное описаниеДля того чтобы коэффициенты , , можно было определить, и притом единственным образом, необходимо и достаточно, чтобы определитель полученной системы линейных уравнений был отличен от нуля:

(6)

Интерполяционный полином в форме Лагранжа, несмотря на свое изящество, неудобен для вычислений тем, что при увеличении числа узлов интерполяции приходится перестраивать весь полином заново. Перепишем интерполяционный полином Лагранжа в иной, эквивалентной форме:

Изображение выглядит как Шрифт, текст, рукописный текст, типография

Автоматически созданное описание

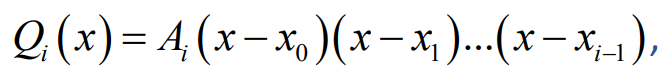
(13)

Изображение выглядит как Шрифт, типография, рукописный текст, каллиграфия

Автоматически созданное описаниеИзображение выглядит как Шрифт, рукописный текст, каллиграфия, символ

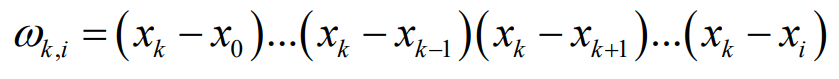
Автоматически созданное описаниегде — полиномы Лагранжа степени i ≤ n, соответствующие узлам интерполирования . В частности, полином нулевой степени.

Полином имеет степень i и по построению обращается в нуль при поэтому его можно представить в виде

 (15)

Изображение выглядит как Шрифт, белый, текст, диаграмма

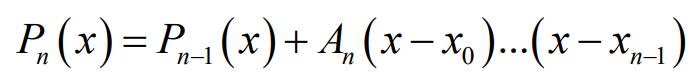
Автоматически созданное описаниегде Ai — числовой коэффициент при x^i. Поскольку не содержит степени i , то Ai просто совпадает с коэффициентом при x^i в полиноме . Согласно (9) и (12) его можно записать в виде

где (17)

Изображение выглядит как Шрифт, рукописный текст, линия, типография

Автоматически созданное описание

при этом (18)

Формулы (14) и (16) позволяют написать рекуррентное соотношение для полинома :

Выражая аналогичным образом по индукции через , через и т.д., получаем окончательную формулу для полинома

Изображение выглядит как текст, Шрифт, рукописный текст, каллиграфия

Автоматически созданное описание

Представление (20) удобно для вычислителя, поскольку увеличение n на единицу требует только добавления к «старому» многочлену одного дополнительного слагаемого. Такое представление интерполяционного полинома называют интерполяционным полиномом в форме Ньютона.

Для решения данной задачи необходимо сделать:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание



**Программный код**

import math  
  
def newtonPolynomial(x, y, n, xx): # полином Ньютона  
 sum = y[0]  
 for s in range(1, n): # вычисление разделенных разностей  
 sigma = 0 # сумма сигма  
 for j in range(0, s+1):  
 q = 1  
 for i in range(0, s+1): # y[i] разделенные на q c переменной сигма  
 if i != j:  
 q \*= (x[j] - x[i])  
 sigma += y[j] / q  
 for k in range(0, s):  
 sigma \*= (xx - x[k])  
 sum += sigma  
 return sum  
  
a = 0 # начало отрезка  
b = 2 # конец отрезка  
N = 10 # количество частей  
M = 3 \* N  
dx\_1 = (b - a) / N # размер каждого интервала  
dx\_2 = (b - a) / M  
  
# Создание списков для хранения значений x, y, N = 10  
mas1\_x = [0] \* (N + 1)  
mas1\_y = [0] \* (N + 1)  
  
print("x", "\t\t", "f(x)")  
  
# Заполнение списков значениями функции (sqrt(x) - x)  
for i in range(0, N+1):  
 mas1\_x[i] = a + i \* dx\_1  
 mas1\_y[i] = math.sqrt(mas1\_x[i]) - mas1\_x[i]  
 print(mas1\_x[i], "\t\t", mas1\_y[i])  
  
# Создание списков для хранения значений x, y  
mas2\_x = [0] \* (M + 1)  
mas2\_y = [0] \* (M + 1)  
  
print("\nx", "\t\t", "f(x)")  
  
# Вывод значений функции в точках отрезка с более малым шагом  
for i in range(0, M+1):  
 mas2\_x[i] = a + i \* dx\_2  
 mas2\_y[i] = math.sqrt(mas2\_x[i]) - mas2\_x[i]  
 print(mas2\_x[i], "\t\t", mas2\_y[i])  
  
print("\nResult of Newton for M = 3N")  
print("x", "\t\t", "f(x)", "\t\t", "P", "\t\t", "Delta")  
  
for i in range(0, M+1):  
 P = newtonPolynomial(mas1\_x, mas1\_y, N + 1, mas2\_x[i])  
 print(mas2\_x[i], "\t\t", mas2\_y[i], "\t\t", P, "\t\t", abs(P - mas2\_y[i]))  
  
print("\nTask #6")  
  
for N in range(1, 101):  
 M = 3 \* N  
 DX1 = (b - a) / N # размер каждого интервала  
 DX2 = (b - a) / M  
 MAS1\_x = [0] \* (N + 1)  
 MAS1\_y = [0] \* (N + 1)  
 MAS2\_x = [0] \* (M + 1)  
 MAS2\_y = [0] \* (M + 1)  
 for j in range(0, N+1):  
 MAS1\_x[j] = a + j \* DX1  
 MAS1\_y[j] = math.sqrt(MAS1\_x[j]) - MAS1\_x[j]  
 for i in range(0, M+1):  
 MAS2\_x[i] = a + i \* DX2  
 MAS2\_y[i] = math.sqrt(MAS2\_x[i]) - MAS2\_x[i]  
  
 Delta = [0] \* (M + 1)  
 for h in range(0, M+1):  
 newton = newtonPolynomial(MAS1\_x, MAS1\_y, N + 1, MAS2\_x[h])  
 Delta[h] = abs(newton - MAS2\_y[h])  
  
 count = 0.0  
 for k in range(0, M+1):  
 if Delta[k] > 0.1:  
 break  
 else:  
 count += 1  
 if count >= M:  
 print("Answer N =", N)  
 break  
  
 del MAS1\_x  
 del MAS1\_y  
 del MAS2\_x  
 del MAS2\_y  
 del Delta  
  
del mas1\_x  
del mas1\_y  
del mas2\_x  
del mas2\_y

**Вывод программы**

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, меню, дизайн

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, меню, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

**Вывод пунктов 5 и 6 лабораторной работы №2**

**Изображение выглядит как текст, снимок экрана, меню

Автоматически созданное описание**

**Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание**

**Заключение**

В ходе данной лабораторной работы было построено равномерное разбиение отрезка из задания на N = 10 частей, рассчитано значения функции из задания в узлах интерполяции, построено интерполяционный полином Ньютона согласно значениям из предыдущих пунктов, построено равномерное разбиение отрезка из задания на M=3N частей, посчитано значения исходной функции и построенного в полинома Ньютона, при котором