Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

ОТЧЕТ

ПО ПРЕДМЕТУ “МЕТОДЫ ЧИСЛЕННОГО АНАЛИЗА”

НА ТЕМУ “ Интерполяция алгебраическими многочленами”

студентки 2 курса 2 группы

Курец Любови Олеговны

Преподаватель

Горбачева Юлия Николаевна

1. Постановка задачи

На отрезке [a,b] заданы функции $f_1(x)$ и $f_2(x)$. Построить многочлены степени n = 3, 5, 7, 10, 15, интерполирующие каждую из них по узлам

а) равномерно расположенным на указанном отрезке;

б) расположенным на указанном отрезке оптимальным (минимизирующим погрешность) образом. В отчет включить представление, использованное при построении многочленов, способ выбора узлов, графики функций $f_1(x)$ и $f_2(x)$, графики полученных интерполяционных многочленов, а также графики абсолютных погрешностей приближения функций многочленами. Сделать выводы о сходимости интерполяционного процесса по равноотстоящим и чебышѐвским узлам.

2. Краткие теоретические сведения

При глобальной интерполяции на всем интервале [a,b] строится единый многочлен. Одной из форм записи интерполяционного многочлена для глобальной интерполяции является многочлен Лагранжа:

$$L_n(x)=\sum_{i=0}^{n} y_i \cdot l_i(x)$$

где $l_i(x)$ – базисные многочлены степени n:

$$l_i(x)=\prod_{\substack{j=1\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}=\frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

То есть многочлен Лагранжа можно записать в виде: $$L_n(x)=\sum_{i=0}^{n} y_i \cdot \prod_{\substack{j=1\\ j\neq i}}^{n}\frac{x-x_j}{x_i-x_j}$$

Многочлен $l_i(x)$ удовлетворяет условию $l_i(x_j)=\begin{cases}1, & i=j\\ 0, & i\neq j\end{cases}$. Это условие означает, что многочлен равен нулю при каждом x_j кроме x_i , то есть $x_0, x_1, \ldots x_{i-1}, x_{i+1}, \ldots x_n$ – корни этого многочлена. Таким образом, степень многочлена $L_n(x)$ равна n и при $x!=x_i$ обращаются в ноль все слагаемые суммы, кроме слагаемого с номером $i=j$, равного y_j.

Многочлены Чебышева первого рода $T_n(x)$ могут быть также определены с помощью равенства: $T_n(\cos(\theta))=\cos(n\theta)$.

Многочлены Чебышева первого рода $T_n(x)$ могут быть определены с помощью рекуррентного соотношения:

$T_0(x)=1$; $T_1(x)=x$; $T_{n+1}(x)=2xT_n(x)-T_{n-1}(x)$;

Вычисление равноотстоящих узлов по формуле:

$$x_i = a + \frac{b-a}{n-1}(i-1),\ i = 1, \dots, n.$$

Вычисление чебышевских узлов:

$$x_m = \frac{b+a}{2} + \frac{b-a}{2}\cos\left(\frac{\pi(2m-1)}{2n}\right), \quad m = 1, \dots, n,$$

3. Листинг программы

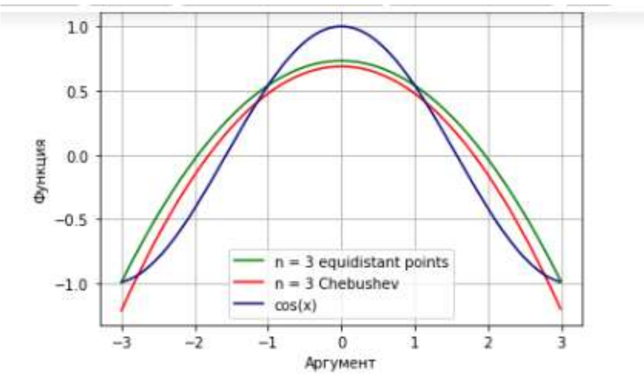
```
import math
import numpy as np
import matplotlib.pyplot as plt


def f1(x):
    return math.cos(x)


def f2(x):
    return abs(2 * math.sin(2 * x) - 1)


def plot_points(fx, a1, b1, h):
    k = round((b1 - a1) / h)
    x = np.zeros(k);
    y = np.zeros(k);
    for i in range(k):
        y[i] = fx(a1)
        x[i] = a1
        a1 = a1 + h
    return x, y


def ds_equidistant_points(a, b, n):
    h = (b - a) / n;
    v = a;
    x = np.zeros(n + 1)
    for i in range(n + 1):
        x[i] = v
        v = v + h
    return x


# разделение отрезка на точки по Чебышеву
def ds_Chebushev(a, b, n):
```

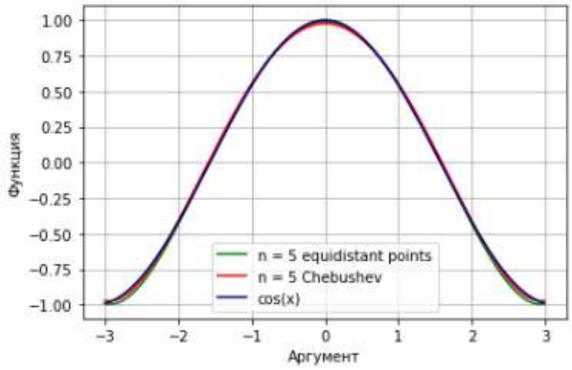
```
    n += 1
    x = np.zeros(n)
    a1 = (a + b) / 2
    a2 = (b - a) / 2
    for i in range(n):
        x[i] = a1 + a2 * math.cos((2 * (i + 1) - 1) / (2 * n) * math.pi)
    return x


# интерполяционный многочлен Лагранжа
def lagrange_polynomial(v, x, fx):
    n = x.size
    res = 0
    for i in range(n):
        p = fx(x[i])
        for j in range(n):
            if (j != i):
                p = p * (v - x[j]) / (x[i] - x[j])
        res += p
    return res


def lagrange(fx, divideMethod, a, b, n, a1, b1, h):
    x = divideMethod(a, b, n)
    k = round((b1 - a1) / h)
    y = np.zeros(k)
    z = np.zeros(k)
    for i in range(k):
        y[i] = lagrange_polynomial (a1, x, fx)
        z[i] = a1
        a1 = a1 + h
    return z, y


# вычисление погрешности
def error(fx, x, y):
    err = np.zeros(x.size)
    for i in range(x.size):
        err_ = math.fabs(fx(x[i]) - y[i])
        # if(errtemp>err):
        err[i] = err_
    return err


a = -3
b = 3
a1 = -3
b1 = 3
h = 0.001
n = np.array((3, 5, 7, 10, 15))


def main(fx, label):
    xf, yf = plot_points(fx, a1, b1, h)
    for i in range(n.size):
        x, y = lagrange(fx, ds_equidistant_points, a, b, n[i], a1, b1, h)
        x2, y2 = lagrange(fx, ds_Chebushev, a, b, n[i], a1, b1, h)
        error_EP = error(fx, x, y)
        error_Ch = error(fx, x2, y2)

        plt.plot(x, y, color="green", label=u"n = " + n[i].astype(str) + "equidistant 
points")
```

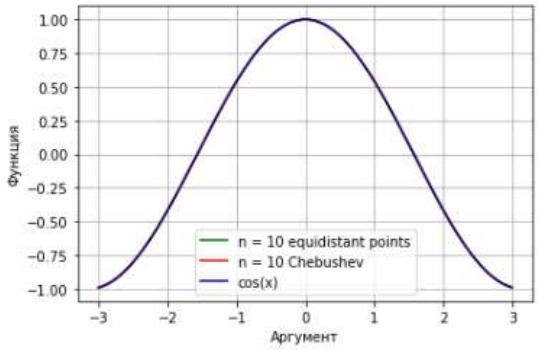
```
        plt.plot(x2, y2, color="red", label=u"n = " + n[i].astype(str) + "
Chebushev")
        plt.plot(xf, yf, color="navy", label=label)
                plt.legend()
        plt.grid(True)
        plt.xlabel(u'Аргумент')
        plt.ylabel(u'Функция')
        plt.show()
   for i in range(n.size):
        x, y = lagrange(fx, ds_equidistant_points, a, b, n[i], a1, b1, h)
        x2, y2 = lagrange(fx, ds_Chebushev, a, b, n[i], a1, b1, h)
        error_EP = error(fx, x, y)
        error_Ch = error(fx, x2, y2)
        plt.plot(x, error_EP, color="silver", label=u" error at equidistant nodes")
        plt.plot(x, error_Ch, color="grey", label=u" error at Chebyshev nodes")
        plt.grid(True)
        plt.xlabel(u'Аргумент')
        plt.ylabel(u'Функция')
        plt.show()

main(f1, "cos(x)")
main(f2, "|2sin(2x)-1|")
```

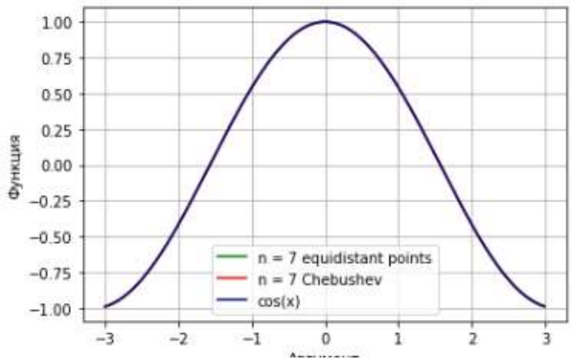
4. Результаты

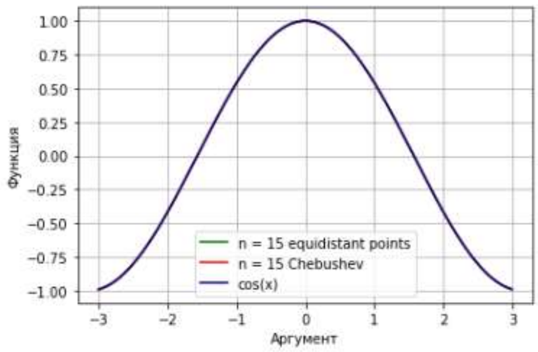
f₁(x)=cos(x)



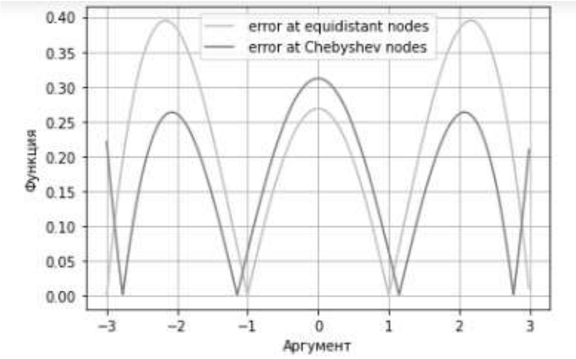


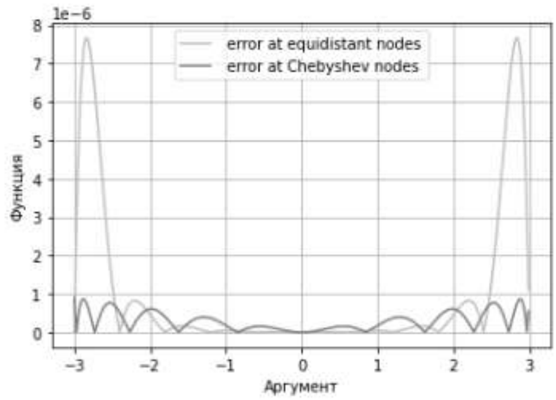


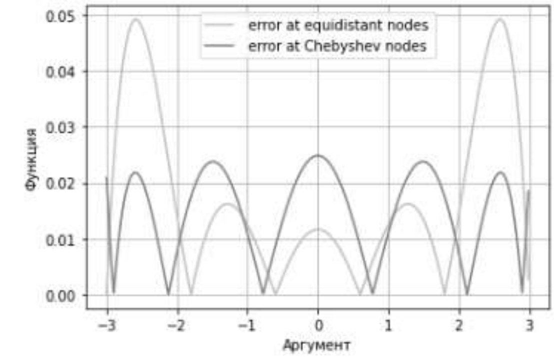


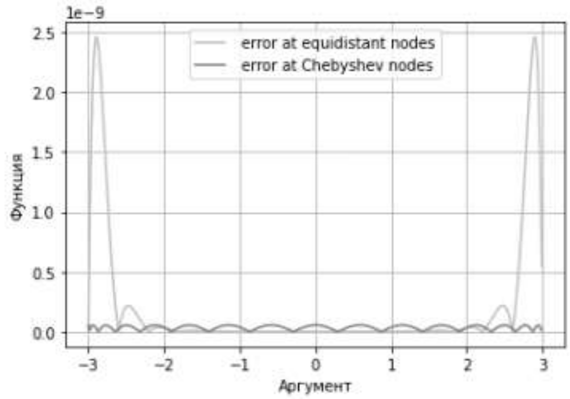


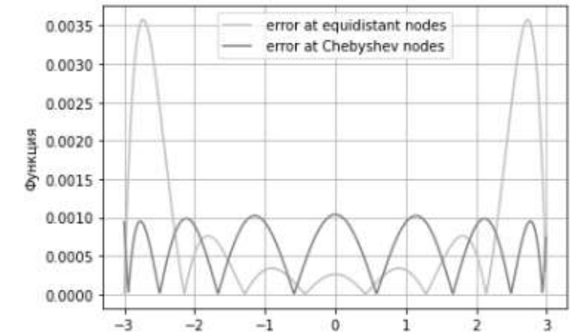
Графики погрешности:



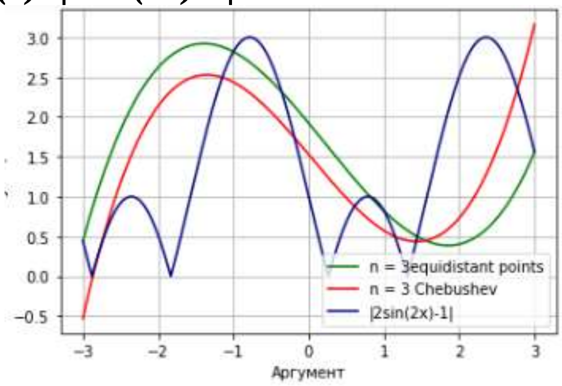


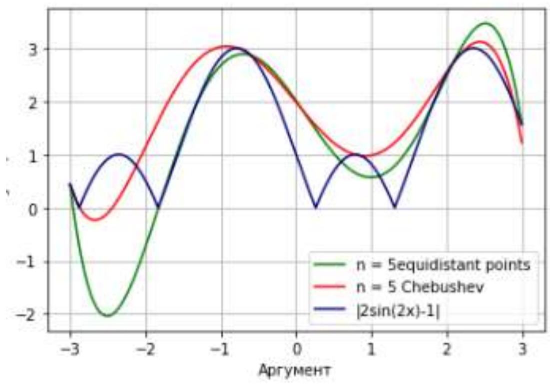


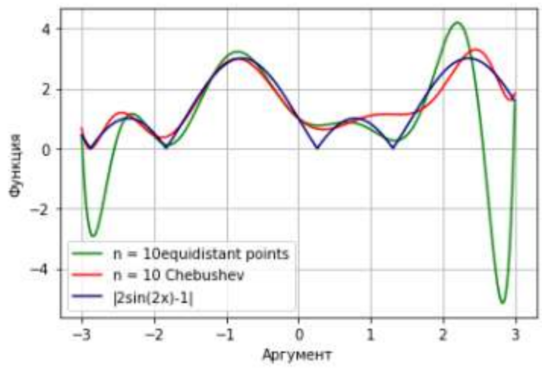


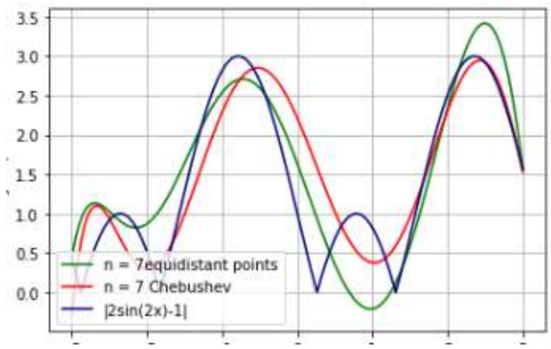


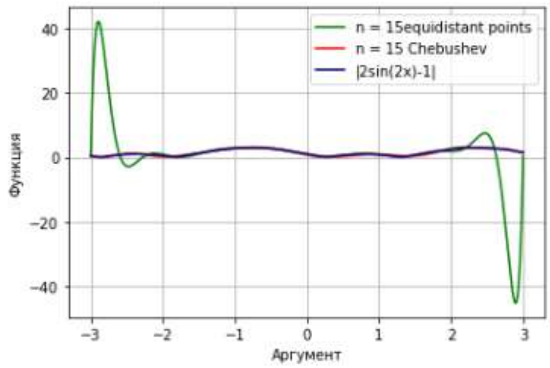
f₂(x)=|2sin(2x)-1|


n = 3equidistant points
n = 3 Chebushev
|2sin(2x)-1|
Аргумент

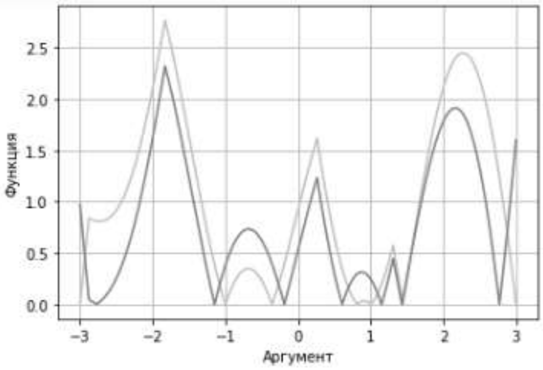

n = 5equidistant points
n = 5 Chebushev
|2sin(2x)-1|
Аргумент

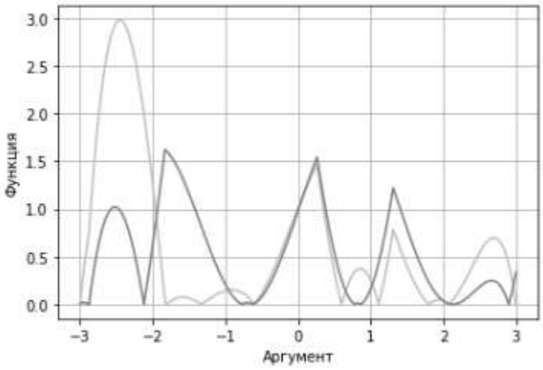

n = 10equidistant points
n = 10 Chebushev
|2sin(2x)-1|
Функция
Аргумент

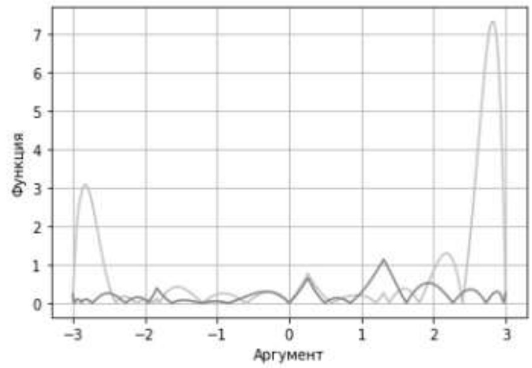

n = 7equidistant points
n = 7 Chebushev
|2sin(2x)-1|

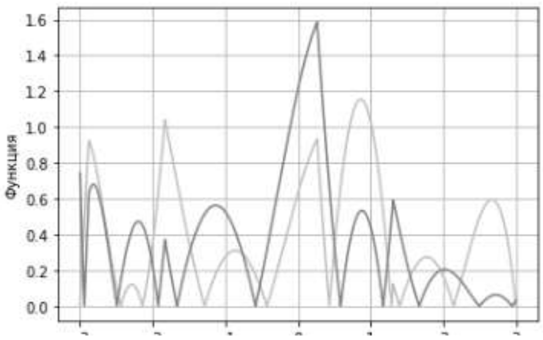

n = 15equidistant points
n = 15 Chebushev
|2sin(2x)-1|
Функция
Аргумент

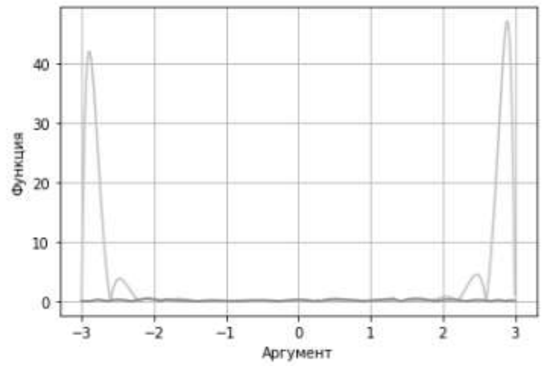
Графики погрешности:











5. Вывод

Многочлен Чебышева при увеличении степени сходится быстрее, чем многочлен Лагранджа по равностоящим узлам. Для многочлена Чебышева сходимость обязательна, в то же время как у многочлена Лагранджа сходимость не обязательна.