КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Семестровая работа по предмету: «Вычислительные методы» на тему: « Интерполирование функций»

Вариант №6

Работу выполнил:
Студент 3-го курса
Группы 09-409
Набиев М.А.

Работу проверила:
Профессор Павлова М.Ф.
20 апреля 2017 г.

Казань-2017

1. Постановка задачи

Одна из специальных задач функций математической физики — интеграл Френеля, определяется следующим образом:

$$C(x)=\int_0^x \cos\left(\frac{\pi t^2}{2}\right)dt$$

Цель задания — изучить и сравнить различные способы приближенного вычисления этой функции

Для этого:

1. Протабулировать *C(x)* на отрезке *[a,b]* с шагом *h*, и точностью ε , основываясь на ряде Тейлора, предварительно вычислив его:

$$C(x)=\sum_{n=0}^{\infty}\frac{(-1)^n\left(\frac{\pi}{2}\right)^{2n}}{(2n)!(4n+1)}x^{4n+1}$$

где a=0, b=1.5, h=0.15, $\varepsilon=10^{-6}$, и получить таким образом таблицу:

x_0	x_1	x_2	...	x_n
f_0	f_1	f_2	...	f_n

$$f_i=C(x_i),\quad x_i=a+i*h,\quad i=0\ldots n,\quad h=\frac{b-a}{n}$$

2. По полученной таблице найти приближенное значение производной функции *C(x)*, используя формулу :

$$L'_n(x)=\sum_{k=0}^{n} f_k*l_k(x),\quad где\quad l_k(x)=\sum_{\substack{j=0\\ j\neq k}}^{n}\frac{\prod_{i=0}^{n}(x-x_i)}{(x-x_k)(x-x_j)\prod_{\substack{i=0\\ i\neq j}}^{n}(x_k-x_i)}$$

и вычислить погрешность интерполирования:

$$\varepsilon_n=\max_{x\in(a,b)}\varepsilon(x),\varepsilon(x)=|C'(x)-L'(x)|$$

Значения производной интерполяционного полинома Лагранжа вычислять в точках на отрезке [a,b], находящихся посередине между двумя узлами интерполяции. В качестве узлов интерполяции взять:

1) равномерно распределенные узлы $\{x_i\}_{i=0}^{n}$, где n=10

2) корни полинома Чебышева вычисляемые по формуле

$x_k=\frac{b-2}{2}t_k+\frac{b+a}{2}$, где

$$t_k=\cos\left(\frac{2k+1}{2(n+1)}\pi\right),\quad k=0,\ldots,n$$

Сравнить погрешности при различном выборе узлом интерполяции.

3. Провести эксперимент:

а) Построить интерполяционный полином Лагранжа по n равномерно распределенным узлам, При этом:

- Вычислять значения полученного полинома Лагранжа в n точках на отрезке [a,b], находящихся между каждыми двумя узлами интерполяции.

- измерять максимальную погрешность:

$$\varepsilon_n = \max_{x\in(a,b)} \varepsilon(x), \varepsilon(x) = |C'(x) - L'(x)|$$

б) Построить интерполяционный полином Лагранжа по корням полинома Чебышева, затем:

- вычислять значения полученного полинома Лагранжа в n точках на отрезке [a,b], находящихся между каждыми двумя узлами интерполяции.
- измерять максимальную погрешность.

$$\varepsilon_n = \max_{x\in(a,b)} \varepsilon(x), \varepsilon(x) = |C'(x) - L'(x)|$$

в) Исследовать зависимость погрешности решения от числа узлов интерполяции.

- Для равномерно распределенных узлов.
- Для корней полинома Чебышева.

г) Построить графики изменения максимальных погрешностей в зависимости от числа узлов.

- Для равномерно распределенных узлов.
- Для корней полинома Чебышева

Замечание. *При вычислении ряда Тейлора* $\sum_{n=0}^{\infty} a_n$ *учесть, что каждый последующий член ряда* a_{n+1} *получается из предыдущего члена* a_n *, умножением на некоторую величину* q_n *,т.е.* $a_{n+1} = a_n * q_n$

Это позволит избежать переполнения при вычислении факториалов, встречающихся в рассматриваемом ряде.

2.Табулирование функции на отрезке.

При вычислении ряда Тейлора $\sum_{n=0}^{\infty} a_n$ будем учитывать, что каждый последующий член ряда

$$C(x) = \sum_{n=0}^{\infty} \frac{(-1)^n \left(\frac{\pi}{2}\right)^{2n}}{(2n)!(4n+1)} 2^{4n+1}$$ получается из предыдущего члена a_n умножением на некоторую величину q_n, т. е.

$a_{n+1} = a_n * q_n$

Это позволит избежать переполнения при вычислении факториалов, встречающихся в рассматриваемом ряде.

Найдем q_n:

$$a_n = \frac{(-1)^n \left(\frac{\pi}{2}\right)^{2n}}{(2n)!(4+1)} x^{4n+1} \qquad a_{n+1} = \frac{(-1)^{n+1} \left(\frac{\pi}{2}\right)^{2n+2}}{(2n+2)!(4n+5)} x^{4n+5}$$

$$q_n = \frac{\frac{(-1)^{n+1} \left(\frac{\pi}{2}\right)^{2n+2}}{(2n+2)!(4n+5)} x^{4n+5}}{\frac{(-1)^n \left(\frac{\pi}{2}\right)^{2n}}{(2n)!(4+1)} x^{4n+1}} = -\frac{\left(\frac{\pi}{2}\right)(4n+1)}{(2n+1)(2n+2)(4n+5)} x^4$$

Для решения поставленной задачи воспользуемся свойством знакопеременных рядов.

Оценка остатка знакопеременного ряда:
Пусть знакочередующийся числовой ряд сходится по признаку Лейбница и его сумма равна S. Обозначим через S_m частичную сумму ряда, включающую n членов. Тогда остаток знакочередующегося ряда по абсолютной величине меньше модуля первого отброшенного слагаемого

$$|S-S_m|<|a_{m+1}|,\ \ где\, S_m=\sum_{i+1}^{m} a_i$$

Имеем, что: $|S-S_m|<|a_{m+1}|<\varepsilon$, $где\, \varepsilon - заданная\, точность, равная\, 10^{-6}$.

При вычислении данного ряда получаются следующие данные, где n_i - количество членов ряда, необходимых для достижения заданной точности.

x_i	$f(x_i)$	n_i
0.0	0.0000000000	0
0.15	0.1499812643	2
0.30	0.2994009763	2
0.45	0.4454682286	3
0.60	0.5810954470	4
0.75	0.6935259936	4
0.90	0.7648230199	5
1.05	0.7759095224	6
1.20	0.7154377221	7
1.35	0.5922667500	8
1.50	0.4452611747	9

Таблица 1. Значения функции вычисленные по ряду Тейлора

График данной функции имеет такой вид:

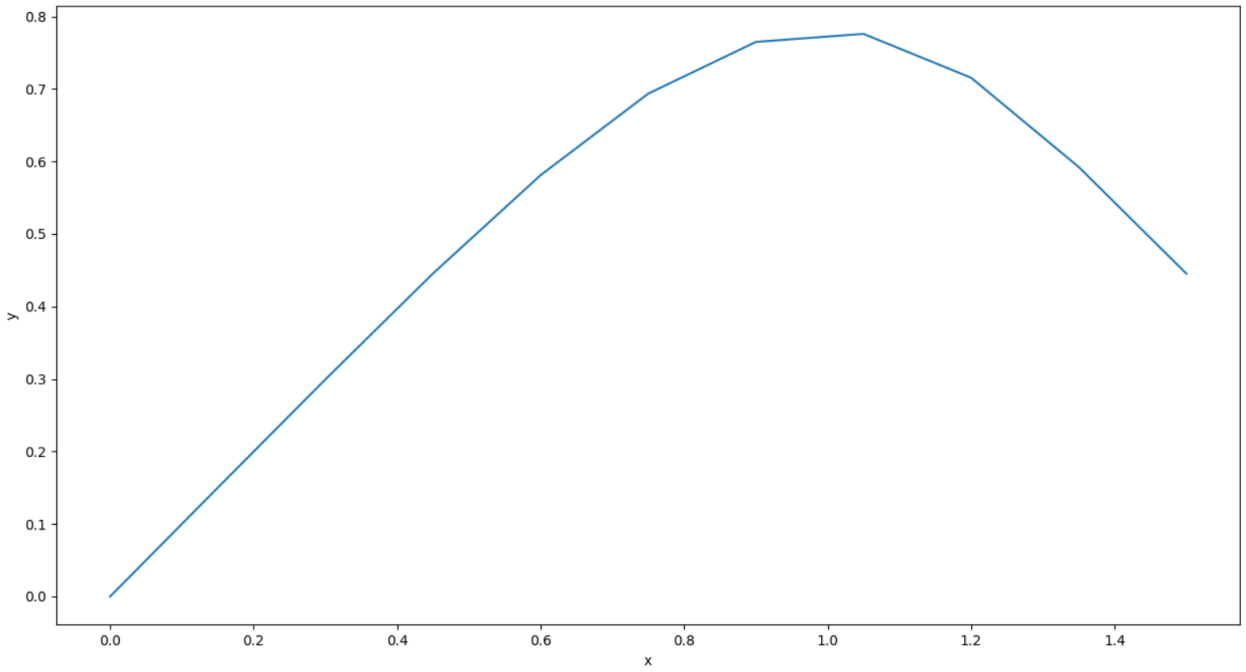

График 1. График функции C(x)

3. Построение интерполяционного полинома.

В результате работы программы, мною были получены следующие значения:
1) для равномерно распределенных узлов:
При a=0, b=1.5, h=0.15

x_i	$L_n'(x)$	$C'(x)$	$\lvert L_n'(x)-C'(x)\rvert$
0	0.9949232185	1	0.0050767815
0.15	0.9995095201	0.9993755041	0.000134016
0.3	0.9900088145	0.9900236577	1.4843251019E-05
0.45	0.9498386789	0.9498356793	2.999591766E-06
0.6	0.8443271008	0.8443279255	8.2473561E-07
0.75	0.6343934878	0.6343932842	2.03676585E-07
0.9	0.2940403759	0.2940403252	5.0644004E-08
1.05	-0.1603122019	-0.1603118919	0.00000031
1.2	-0.6374244362	-0.6374239897	4.46451369E-07
1.35	-0.9613480218	-0.9613818739	3.3852115424E-05
1.5	-0.9264391816	-0.9238795325	0.0025596491

таблица 2. Значения интерполяционного полинома, вычисленные по равномерным узлам интерполяции

График ошибок:

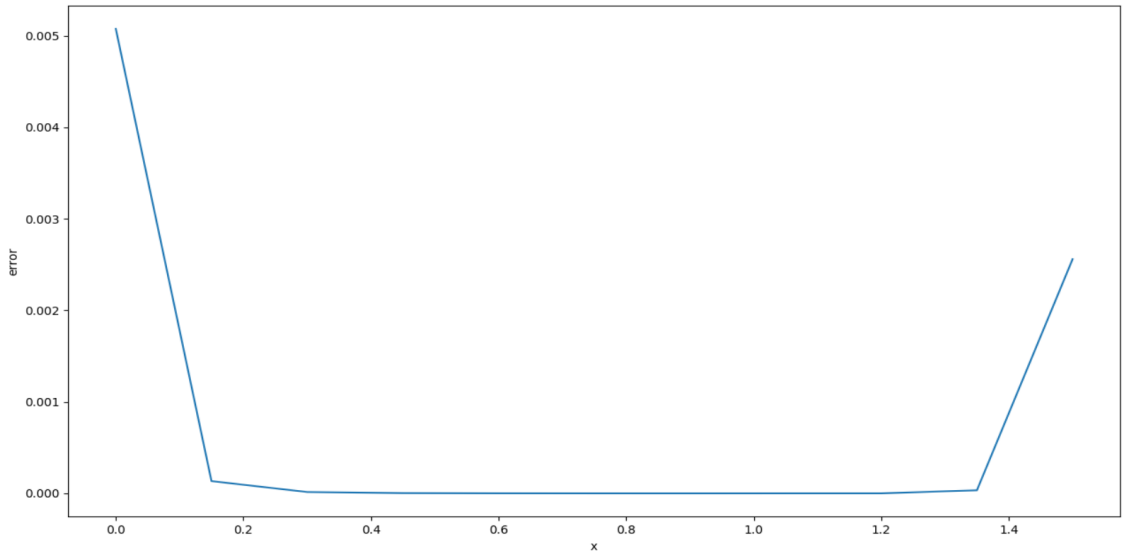

График 2. Ошибки в точках при равномерно выбранных узлах интерполяции и n=10

Для полинома Чебышева:

x_i	$L_n'(x)$	$C'(x)$	$\lvert L_n'(x)-C'(x)\rvert$
0	0.9998306867	1	0.0001693133
0.15	0.999355434	0.9993755041	2.0070111264E-05
0.3	0.9900400794	0.9900236577	1.6421673412E-05
0.45	0.9498323964	0.9498356793	3.28290563E-06
0.6	0.8443115464	0.8443279255	1.6379067152E-05
0.75	0.6344214105	0.6343932842	2.812630894E-05
0.9	0.2940222116	0.2940403252	1.8113658186E-05
1.05	-0.1603205079	-0.1603118919	0.000008616
1.2	-0.6373887633	-0.6374239897	3.52264212E-05
1.35	-0.9614338641	-0.9613818739	5.199022298E-05
1.5	-0.924391342	-0.9238795325	0.0005118095

Таблица 3. Значения интерполяционного полинома, вычисленные по корням полинома Чебышева

График ошибок:

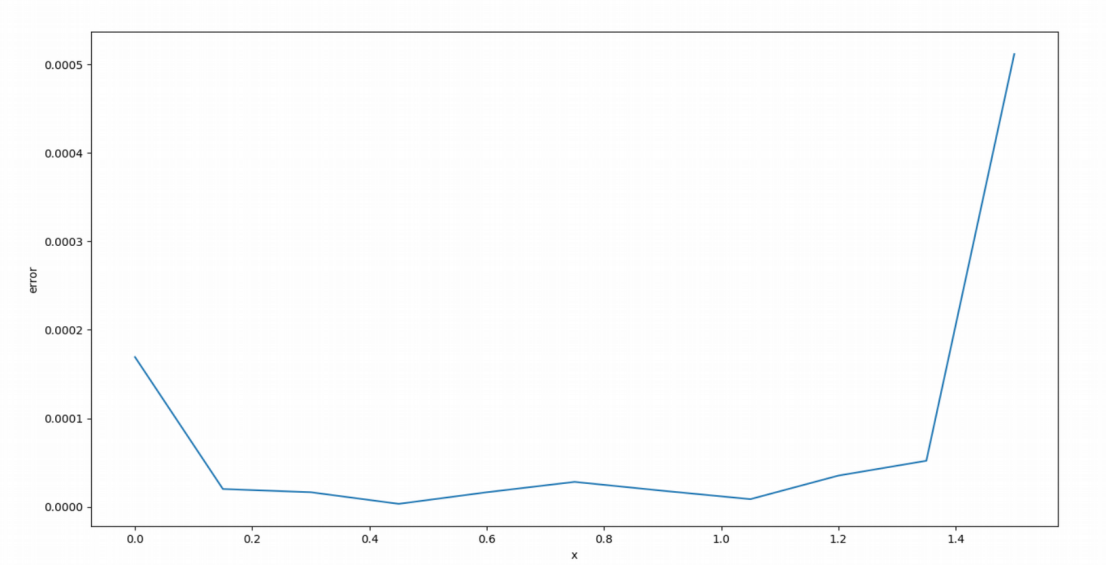

График 3. Ошибки в точках при узлах интерполяции, выбранных как корни полинома Чебышева и n=10

Далее приводятся таблицы и графики максимальных ошибок.
а) Для равномерно распределенных узлов:

h	max(ε)
0.0157894737	2.55314167758384E+018
0.0166666667	7.56673739447392E+016
0.0176470588	1.96849254277281E+015
0.01875	84272003680086.6
0.02	2525582385831.92
0.0214285714	60738270952.9984
0.0230769231	2866392289.00728
0.025	80496764.4335996
0.0272727273	1649067.67514456
0.03	137916.35658501
0.0333333333	1472.7290360488
0.0375	202.819053216
0.0428571429	7.0213938327
0.05	0.0165257179
0.06	0.0108772376
0.075	0.0003310534

Таблица 4. Зависимость максимальной ошибки от шага сетки для равномерно выбранных узлов

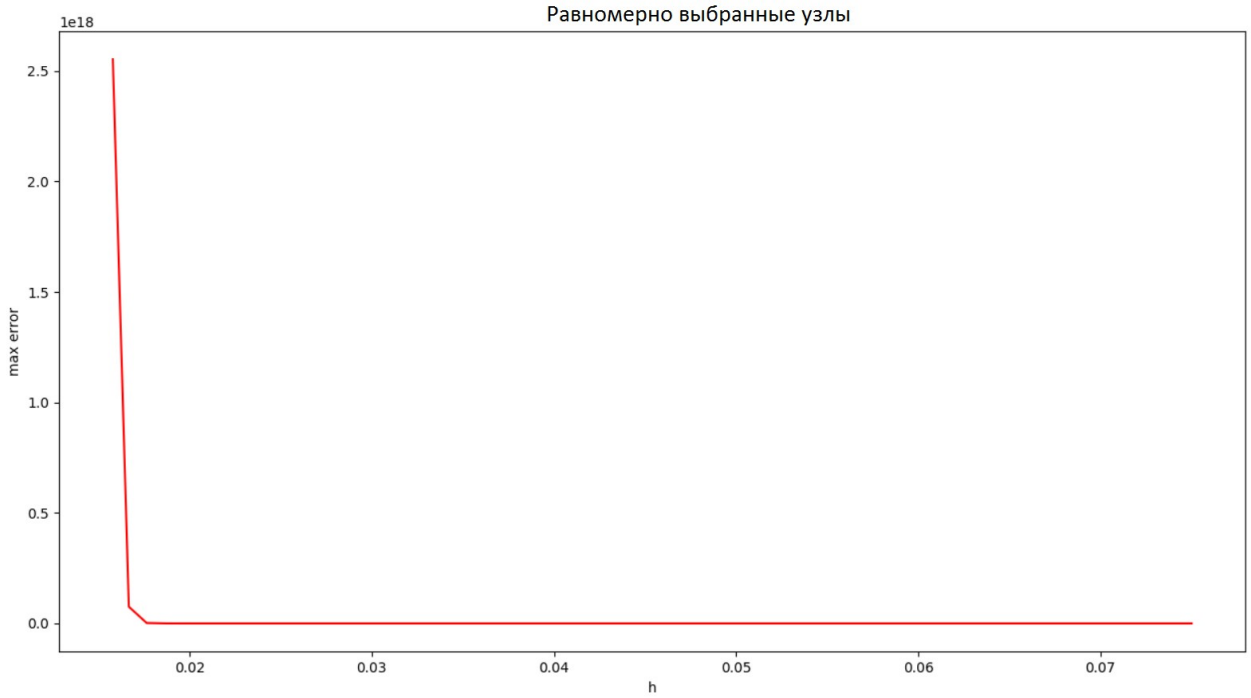

График 4. Зависимость максимальной ошибки от шага сетки для равномерно выбранных узлов интерполяции

б) Для узлов, выбранных как корни полинома Чебышева

h	max(ε)
0.0157894737	1.520085742E-06
0.0166666667	1.717873825E-06
0.0176470588	0.000002196
0.01875	1.713924617E-06
0.02	1.990058902E-06
0.0214285714	1.764669868E-06
0.0230769231	2.547630093E-06
0.025	0.000002217
0.0272727273	2.259208292E-06
0.03	1.639664013E-06
0.0333333333	2.869088299E-06
0.0375	8.17403713E-07
0.0428571429	3.9022394E-07
0.05	2.58114984E-07
0.06	8.35054847E-07
0.075	6.95933648E-07

Таблица 5. Зависимость максимальной ошибки от шага сетки для узлов, выбранных как корни полинома Чебышева

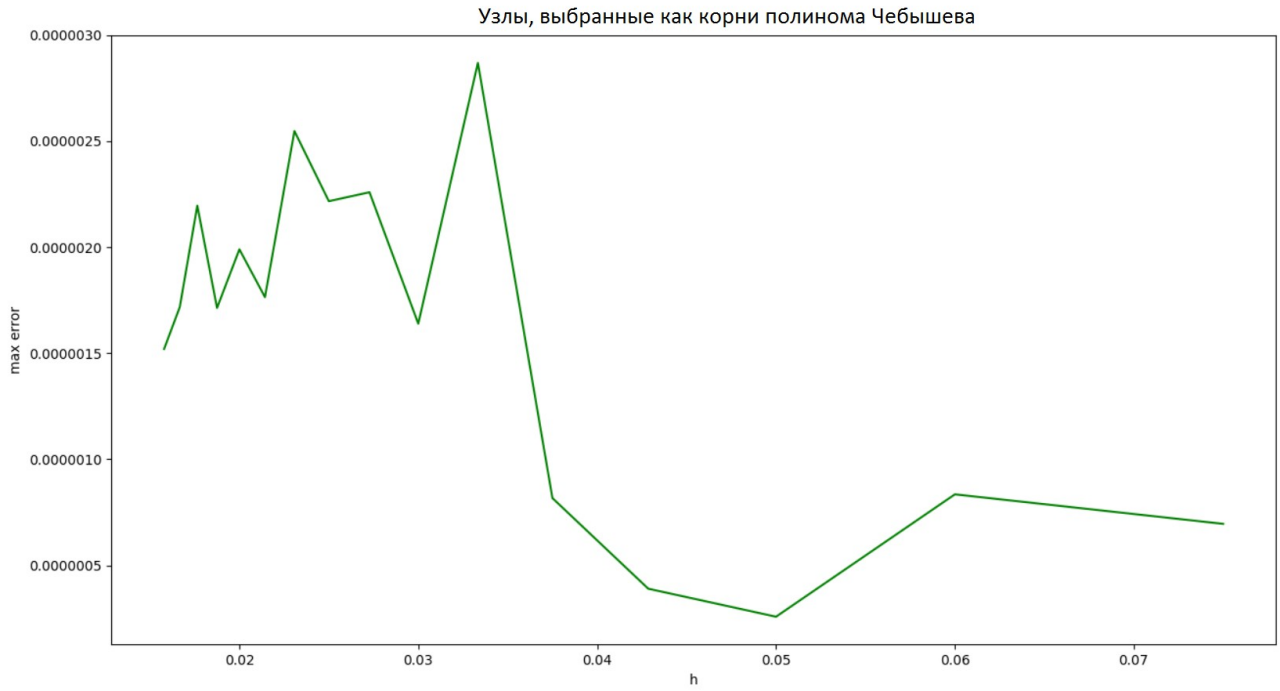

График 5. Зависимость максимальной ошибки от шага сетки для узлов, выбранных как корни полинома Чебышева

Вывод:

При вычислении интерполяционного полинома Лагранжа при фиксированном числе узлов интерполяции лучше использовать корни полинома Чебышева, потому что максимальная погрешность в этом случае будет меньше (см. таблицы 2 и 3)

При увеличении числа узлов интерполяции максимальная погрешность для узлов, выбранных как корни полинома Чебышева, так же будет меньше. Кроме того, для интерполяционного полинома, построенного по равномерно распределенным узлам интерполяции наблюдается потеря устойчивости и увеличение максимальной погрешности при увеличении числа узлов интерполяции, а для интерполяционных полиномов, построенных по корням полинома Чебышева такой тенденции нет. (см. таблица 4,5, графики 4,5).

Приложение. Листинг программы на языке Python 2.7

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

eps = 1e-06

def f(x, eps):
    sum = x
    a = x
    n = 0
    while (math.fabs(a) > eps):
        q = -(((math.pi / 2) ** 2) * (x ** 4) * (4 * n + 1)) / ((2 * n + 1) * (2 *
n + 2) * (4 * n + 5))
        a *= q
        sum += a
        n += 1
    return sum

def l(x_arr, x, k):
    result = 0.
    for i in range(len(x_arr)):
        if i == k:
            continue
        p1 = 1.0
        for j in range(len(x_arr)):
            p1 *= float(x - x_arr[j])
        p2 = 1.
        for j in range(len(x_arr)):
            if (j != k):
                p2 *= float(x_arr[k] - x_arr[j])
        result += p1 / float((x - x_arr[k]) * (x - x_arr[i]) * p2)
    return result

def c(x):
    return math.cos(math.pi*x*x/2)

def L(x_arr, x):
    result = 0.0
    for i in range(len(x_arr)):
        result += f(x_arr[i], eps) * l(x_arr, x, i)
    return result
d
ef chebishev_polynom(a, b, n):
    t_arr = []
    for k in range(n + 1):
        tk = math.cos(math.pi * (2 * k + 1) / float(2 * (n + 1)))
        t_arr.append(tk)
    x_arr = []
    alfa = float(b - a) / 2
    betta = float(a + b) / 2
    for t in t_arr:
        x = alfa * t + betta
        x_arr.append(x)
    return x_arr

def main():
```

```
    a = 0
    b = 1.5
    top = 10
    bot = 100
    step = 10
    max_errors1 = []
    h_arr1 = []
    for n1 in range(bot, top, step):
        print 'n1 =', n1
        x1 = []
        h1 = float(b - a) / n1
        for i in range(n1):
            x1.append(a + h1 / 2 + i * h1)
        x_arr = []
        for i in range(n1 + 1):
            x_arr.append(a + i * h1)
        errors = []
        for i in x1:
            L_val = L(x_arr, i)
            c_val = c(i)
            errors.append(math.fabs(L_val - c_val))
            print'{0:.3f}\t{1:.15f}\t{2:.15f}\t{3:.15f}'.format(i, L_val, c_val,
math.fabs(L_val-c_val))
        max_errors1.append(np.max(np.array(errors)))
        h_arr1.append(h1)
    #
    # chebishev
    #
    max_errors2 = []
    h_arr2 = []
    for n2 in range(bot, top, step):
        print'n2 = ', n2
        h2 = float(b - a) / n2
        x2 = []
        for i in range(n2):
            x2.append(a + h2 / 2 + i * h2)
        x_cheb = chebishev_polynom(a, b, n2)
        errors = []
        for i in x2:
            l_val = L(x_cheb, i)
            c_val = c(i)
            errors.append(math.fabs(l_val - c_val))
            print'{0:.3f}\t{1:.15f}\t{2:.15f}\t{3:.15f}'.format(i, L_val, c_val,
math.fabs(L_val - c_val))
        max_errors2.append(np.max(np.array(errors)))
        h_arr2.append(h2)
    h_arr1 = np.array(h_arr1)
    max_errors1 = np.array(max_errors1)
    h_arr2 = np.array(h_arr2)
    max_errors2 = np.array(max_errors2)
    plt.figure()
    plt.subplot(1, 2, 1)
    plt.plot(h_arr1, max_errors1, 'r')
    plt.title('Ravnomerno')
    plt.ylabel('max error')
    plt.xlabel('h')
    plt.subplot(1, 2, 2)
    plt.plot(h_arr2, max_errors2, 'g')
    plt.title('Chebishev')
```

```
    plt.ylabel('max error')
    plt.xlabel('h')
    plt.show()


if __name__ == "__main__":
    main()
```