|  |  |
| --- | --- |
| Московский авиационный институт | |
| (Национальный исследовательский университет) | |
| Институт №8 «Компьютерные науки и прикладная математика» | |
| Кафедра вычислительной математики и программирования | |
|  | |
| **Курсовая работа** | |
| по курсу «Численные методы»  «Интерполяция экспоненциальными сплайнами» | | |
|  | |
|  | |
|  | Выполнила: Железнов Д. Е. |
|  | Группа: М8О-409Б-20 |
|  | Преподаватель:Пивоваров Д.Е. |
|  | Дата: |
|  | Оценка: |

**Постановка задачи**

Получить интерполяционную формулу по входным данным в виде вектора Входные данные задаются в оконном приложении. В качестве интерполяционной функции использовать экспоненту.

# Теоретические сведения

На основе экспоненты представляются функции синус и косинус – гиперболические и тригонометрические. Такого рода обобщения называются экспоненциальными L-сплайнами и предлагаются для самого широкого использования. Однако проработанные теоретические основы оказываются весьма далекими от практического внедрения. Общие формулы, рассчитанные на все случаи жизни, нередко оказываются настолько сложными, что их не применяют ни в одном конкретном случае. Трудоемкость же вычислений часто оказывается существенно меньшей при применении рекуррентных процедур. Поэтому для практической деятельности по части формирования градуировочных зависимостей сенсорных устройств возникла необходимость выполнить интерполяцию экспонентами, хоть и не по обобщенной методике, но в значительной мере универсальной, ограничиваемой лишь условием монотонности возрастания (или убывания) требуемой функции.

На вход программы подается массив точек (х, у). Для каждых трех точек, начиная с где – количество точек, должна быть построена экспоненциальная функция в виде Параметры A, B и C вычисляются следующим образом:

1. В случае, когда , т. е. когда точки лежат на одной прямой, построить экспоненту не получится. Можно воспользоваться линейной интерполяцией.
2. Если , можно воспользоваться формулами:

,

,

Параметр B можно вычислить по одной из трех формул:

1. В остальных случаях (при условии возрастания) параметры вычисляются следующим образом:

Система уравнений:

Исключим параметр С:

Исключим параметр B:

Полученное уравнение относительно A удобно записать в виде функции, приравниваемой к нулю (т. к. R(0) = 0):

Найдем через производную:

Удвоенное расстояние до точки минимума, являющееся приближенным решением уравнения, возьмем в качестве начального значения для итерационного процесса по методу касательных:

Таким образом находим коэффициент А, затем подстановкой в уравнения выше коэффициенты B и C.

Интервалы от до являются общими для функций и . Поэтому необходимо произвести склеивание функций на данных интервалах. Это можно сделать двумя способами:

1. Усреднение:

В этом случае непрерывность производной не соблюдается.

1. Склеивающая функция вида:

В данном случае соблюдается непрерывность производной.

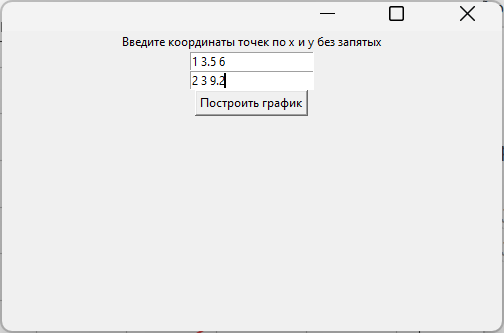
# Программная реализация

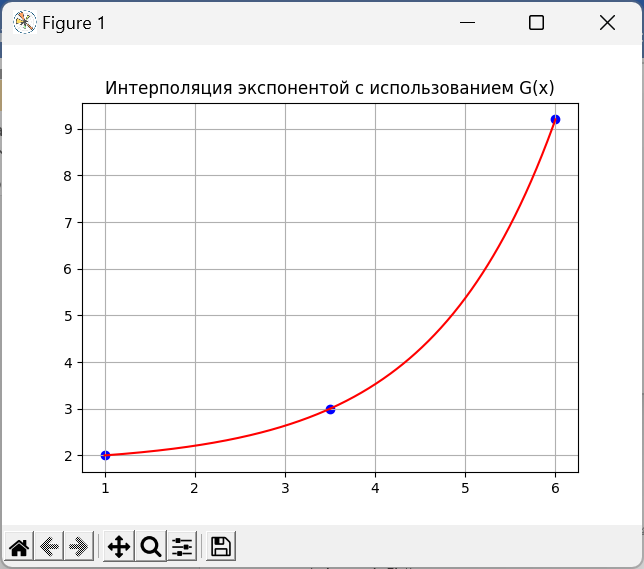
**import** tkinter **as** tk  
**import** numpy **as** np  
**import** matplotlib.pyplot **as** plt  
  
  
  
**def** print\_entered\_value():  
 **global** value  
 **global** value1  
 value = entry.get().split()  
 value = np.array(value,dtype=float)  
 value1 = entry1.get().split()  
 value1 = np.array(value1, dtype=float)  
  
 window.quit()  
  
**def** F(A, B, C, x):  
 **return** C + B \* np.exp(A \* x)  
  
  
**def** check(x, y):  
 swapped = **True  
 while** swapped:  
 swapped = **False  
 for** i **in** range(len(x) - 1):  
 **if** x[i] > x[i + 1]:  
 x[i], x[i + 1] = x[i + 1], x[i]  
 y[i], y[i + 1] = y[i + 1], y[i]  
 swapped = **True  
 for** i **in** range ((len(x) - 1)):  
 **if** (y[i+1] <y[i]):  
 **return** 1,x,y  
  
 **return** 0,x,y  
  
  
**def** exp\_interpolation(x, y):  
  
 A = np.zeros(len(x) - 2)  
 B = np.zeros(len(x) - 2)  
 C = np.zeros(len(x) - 2)  
 res = np.zeros(len(x) - 2)  
  
 d = 0.00001  
 **for** i **in** range(len(A)):  
 dif = (y[i + 2] - y[i + 1]) \* (x[i + 1] - x[i]) / ((y[i + 1] - y[i]) \* (x[i + 2] - x[i + 1]))  
  
 **if** check(x, y)[0] == 1:  
 A[i], B[i], C[i], res = 0, 0, 0, 0  
 **break  
 else**:  
 x, y = check(x, y)[1:]  
  
 **if** abs(dif - 1) < d:  
 A[i] = (y[i + 2] - y[i]) / (x[i + 2] - x[i])  
 B[i] = y[i] - A[i] \* x[i]  
 print(**'При данном наборе точек применима только линейная интерполяция'**)  
 C[i] = 0  
 res[i] = 1  
  
 **if** x[i + 2] + x[i] == x[i + 1] + x[i + 1]:  
 z = (2 \* y[i + 1] - y[i] - y[i + 2])  
 r = (y[i + 2] - y[i + 1]) / (y[i + 1] - y[i])  
 A[i] = np.log(r) / (x[i + 2] - x[i + 1])  
 C[i] = (y[i + 1] \* y[i + 1] - y[i] \* y[i + 2]) / z  
 B[i] = (y[i] - C[i]) \* np.exp(-A[i] \* x[i])  
 res[i] = 2  
  
 **else**:  
  
 Amin = np.log(dif) / (x[i + 2] - x[i])  
 A0 = 2 \* Amin  
 **while** (1):  
 u = np.exp(A0 \* (x[i + 2] - x[i + 1]))  
 v = np.exp(-A0 \* (x[i + 1] - x[i]))  
 F = (y[i + 1] - y[i]) \* (u - 1) + (y[i + 2] - y[i + 1]) \* (v - 1)  
 FF = (y[i + 1] - y[i]) \* (x[i + 2] - x[i + 1]) \* u - (y[i + 2] - y[i + 1]) \* (x[i + 1] - x[i]) \* v  
 dA = -F / FF  
 A0 += dA  
 **if** (abs(dA / Amin) < 0.000000001):  
 **break** A[i] = A0  
 B[i] = (y[i] - y[i + 1]) / (np.exp(A0 \* x[i]) - np.exp(A0 \* x[i + 1]))  
 C[i] = y[i] - B[i] \* np.exp(A0 \* x[i])  
 res[i] = 2  
  
 **return** A, B, C, res  
  
  
**def** plot\_graphics(A, B, C, x, y):  
  
 n = len(A)  
 x1 = np.arange(x[0], x[1] + 0.01, 0.01)  
 y1 = C[0] + B[0] \* np.exp(A[0] \* x1)  
 plt.title(**"Интерполяция экспонентой с использованием G(x)"**)  
 plt.plot(x1, y1, color=**"red"**)  
 plt.scatter(x, y, c=**'b'**)  
  
 **if** (n > 1):  
 **for** i **in** range(1, n):  
 x1 = np.arange(x[i], x[i + 1] + 0.01, 0.01)  
 y1 = ((x[i + 1] - x1) \* (C[i - 1] + B[i - 1] \* np.exp(A[i - 1] \* x1)) + (x1 - x[i]) \* (  
 C[i] + B[i] \* np.exp(A[i] \* x1))) / (x[i + 1] - x[i])  
 plt.plot(x1, y1, color=**"red"**)  
  
 x1 = np.arange(x[-2], x[-1] + 0.01, 0.01)  
 y1 = C[n - 1] + B[n - 1] \* np.exp(A[n - 1] \* x1)  
 plt.plot(x1, y1, color=**"red"**)  
  
 *# xnew = np.arange(x[0], x[-1]+ 0.01, 0.01)  
 # ynew = [lagranz(x, y, i) for i in xnew]  
 # #plt.plot(xnew, ynew)* plt.grid()  
 plt.show()  
 plt.title(**"Интерполяция экспонентой с использованием H(x)"**)  
 x1 = np.arange(x[0], x[1] + 0.01, 0.01)  
 y1 = C[0] + B[0] \* np.exp(A[0] \* x1)  
  
 plt.plot(x1, y1, color=**"green"**)  
 plt.scatter(x, y, c=**'b'**)  
  
 **if** (n > 1):  
 **for** i **in** range(1, n):  
 x1 = np.arange(x[i], x[i + 1] + 0.01, 0.01)  
 y2 = (C[i] + B[i] \* np.exp(A[i] \* x1) + C[i - 1] + B[i - 1] \* np.exp(A[i - 1] \* x1)) / 2  
 plt.plot(x1, y2, color=**"green"**)  
  
 x1 = np.arange(x[-2], x[-1] + 0.01, 0.01)  
 y1 = C[n - 1] + B[n - 1] \* np.exp(A[n - 1] \* x1)  
 plt.plot(x1, y1, color=**"green"**)  
 *# xnew = np.arange(x[0], x[-1] + 0.01, 0.01)  
 # ynew = [lagranz(x, y, i) for i in xnew]  
 # # plt.plot(xnew, ynew)* plt.grid()  
 plt.show()  
  
**def** lagranz(x, y, t):  
 z = 0  
 **for** j **in** range(len(y)):  
 p1 = 1;  
 p2 = 1  
 **for** i **in** range(len(x)):  
 **if** i == j:  
 p1 = p1 \* 1;  
 p2 = p2 \* 1  
 **else**:  
 p1 = p1 \* (t - x[i])  
 p2 = p2 \* (x[j] - x[i])  
 z = z + y[j] \* p1 / p2  
 **return** z  
*#x = [0, 1, 2, 3, 4, 5.3, 6.2, 10]  
#y = [830, 2310, 3069, 3533, 3705, 3817, 3935, 4046]*window = tk.Tk()  
window.geometry(**'500x300+200+100'**)  
window.title(**"Ввод данных"**)  
label = tk.Label(window, text=**"Введите координаты точек по х и у без запятых"**)  
label.pack()  
  
entry = tk.Entry(window)  
entry.pack()  
entry1 = tk.Entry(window)  
entry1.pack()  
  
  
button = tk.Button(window, text=**"Построить график"**, command=print\_entered\_value)  
button.pack()  
  
  
window.mainloop()  
  
x = value  
y = value1  
res = check(x, y)[0]  
x,y = check(x, y)[1:]  
res  
**if** res == 1:  
  
 window1 = tk.Tk()  
 window1.geometry(**'300x200+200+100'**)  
 window1.title(**"Ошибка"**)  
 label1 = tk.Label(window1, text=**"Нарушен порядок возрастания"**)  
 label1.pack(expand=**True**, ipadx=10, ipady=10)  
 window1.mainloop()  
**else**:  
 A, B, C, res = exp\_interpolation(x, y)  
 plot\_graphics(A, B, C, x, y)

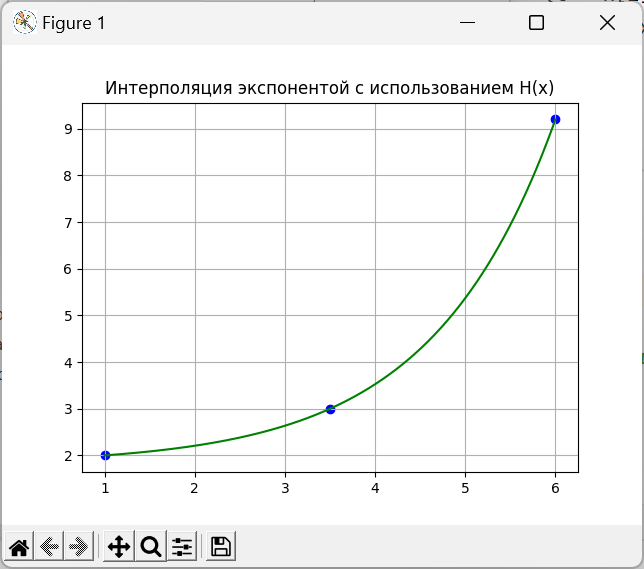
# Вывод программы

Тест №1

|  |  |  |
| --- | --- | --- |
| 1 | 3.5 | 6 |
| 2 | 3 | 9.2 |



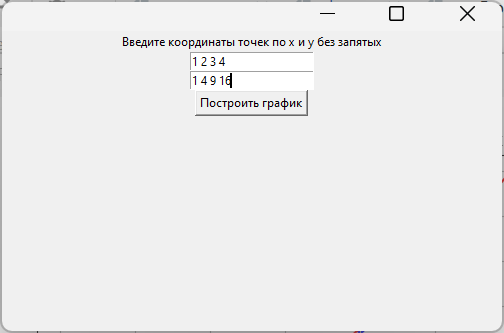


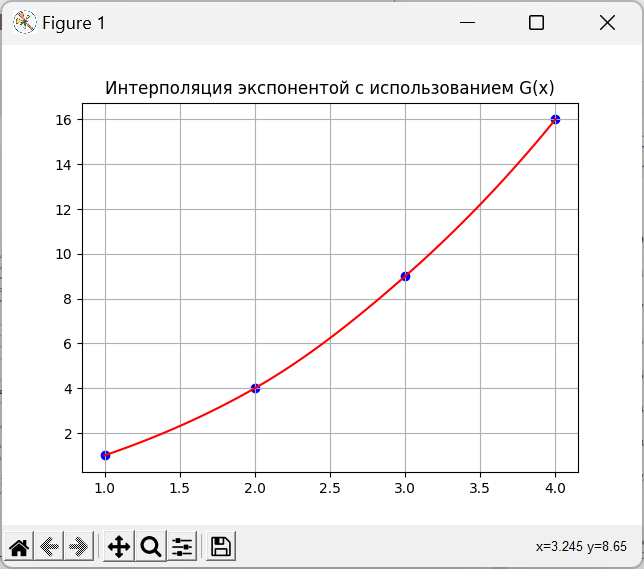


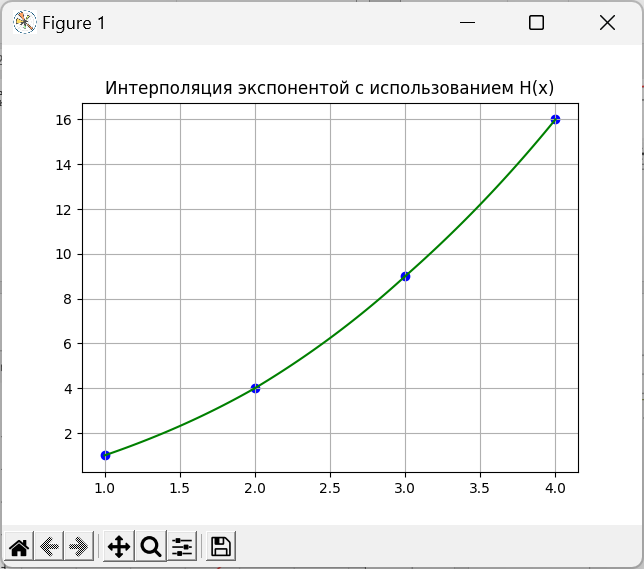
Тест №2

(

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1 | 4 | 9 | 16 |

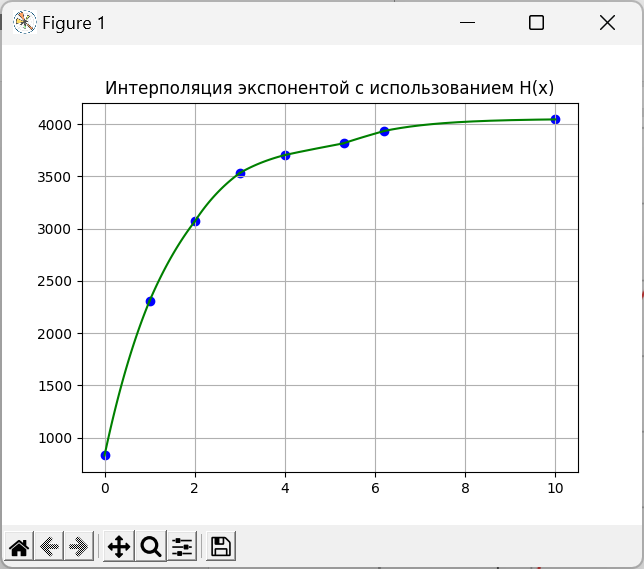
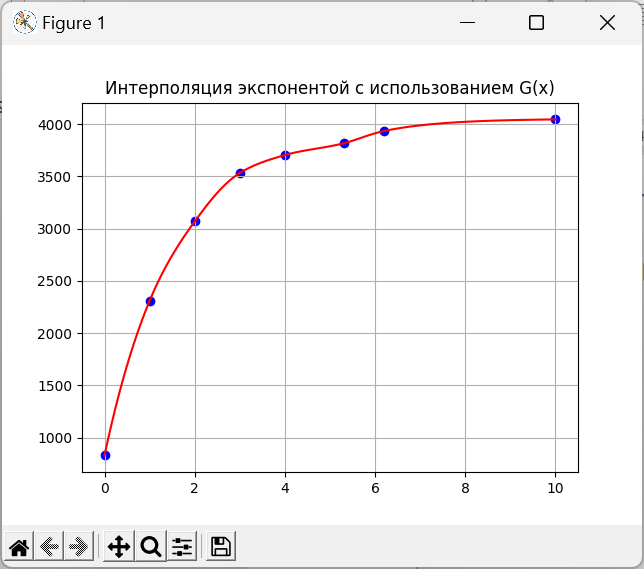
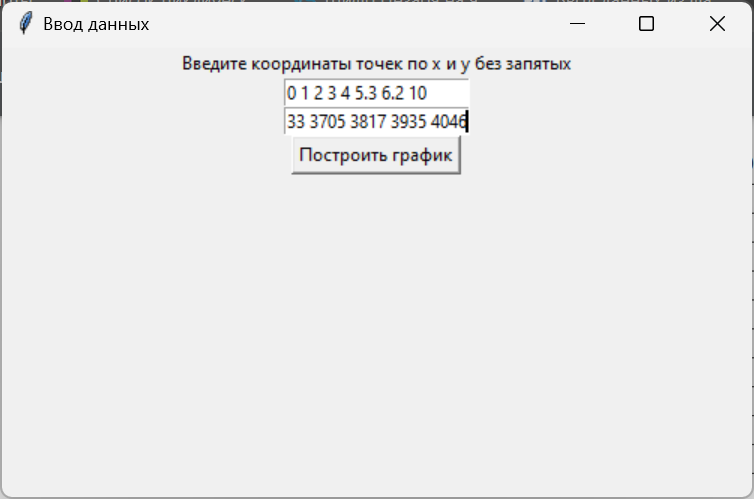






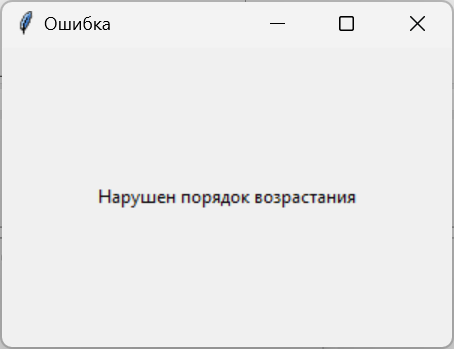
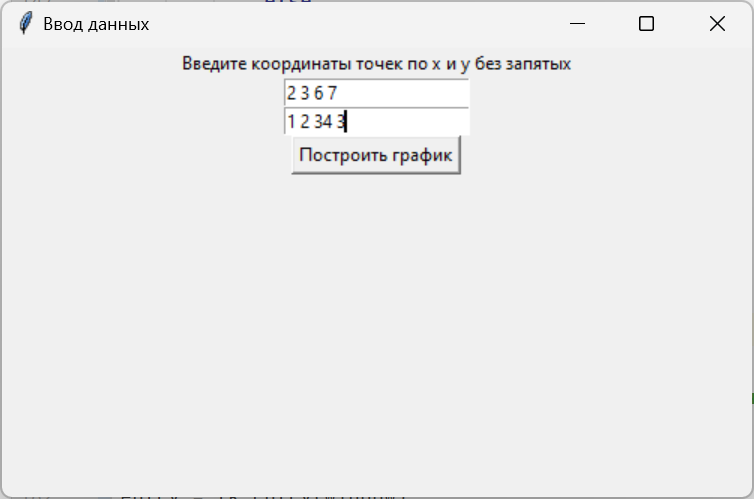
Тест №3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5.3 | 6.2 | 10 |
| 830 | 2310 | 3069 | 3533 | 3705 | 3817 | 3935 | 4046 |



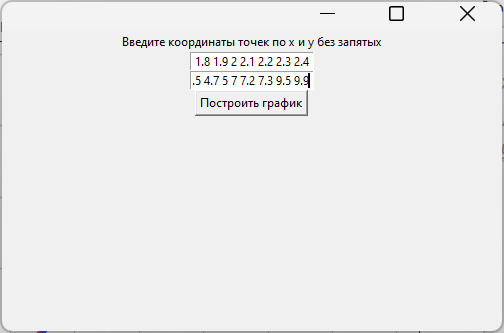
Тест №4

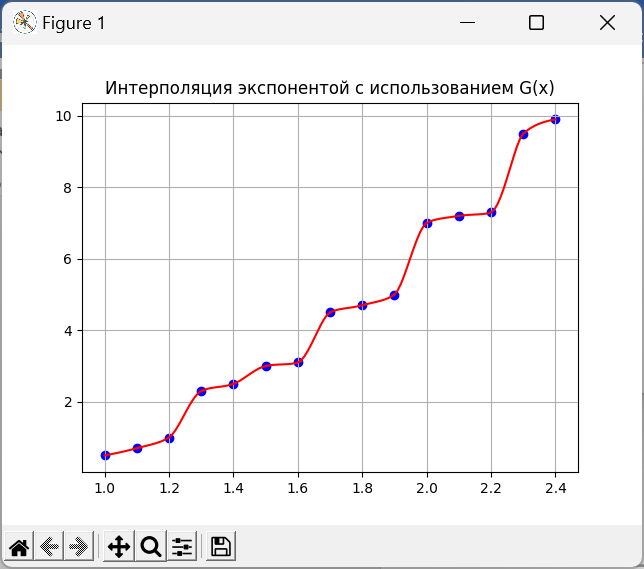
(Нарушено условие возрастания)

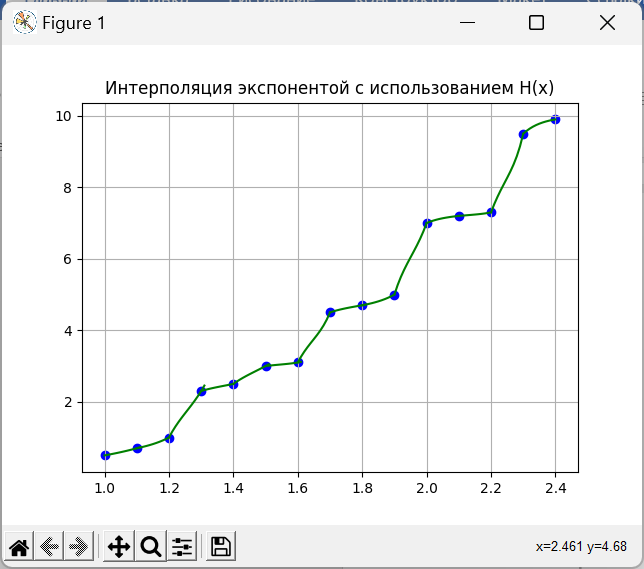


Тест №5

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1.0 | 1.1 | 1.2 | 1.3 | 1.4 | 1.5 | 1.6 | 1.7 | 1.8 | 1.9 | 2.0 | 2.1 | 2.2 | 2.3 | 2.4 |
| 0.5 | 0.7 | 1 | 2.3 | 2.5 | 3.0 | 3.1 | 4.5 | 4.7 | 5.0 | 7.0 | 7.2 | 7.3 | 9.5 | 9.9 |



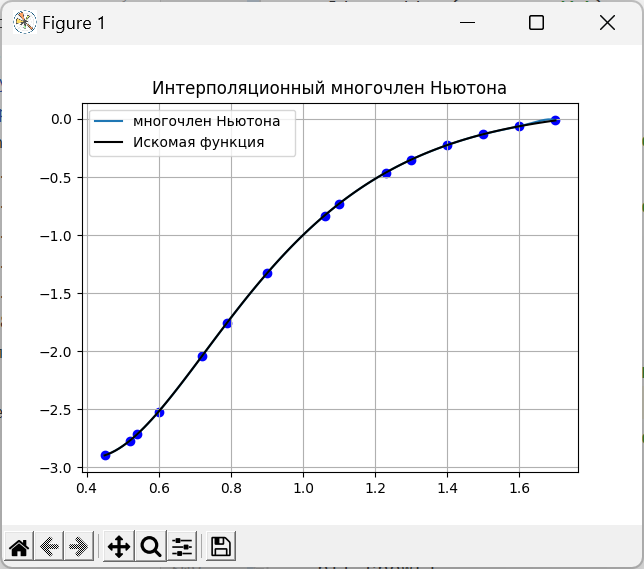
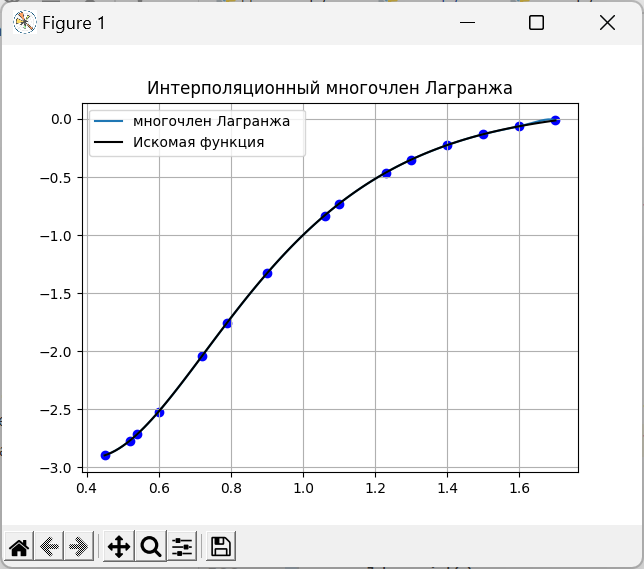
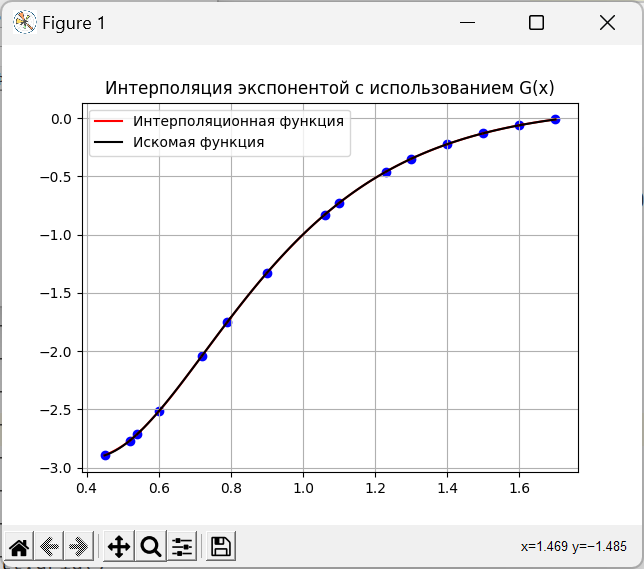


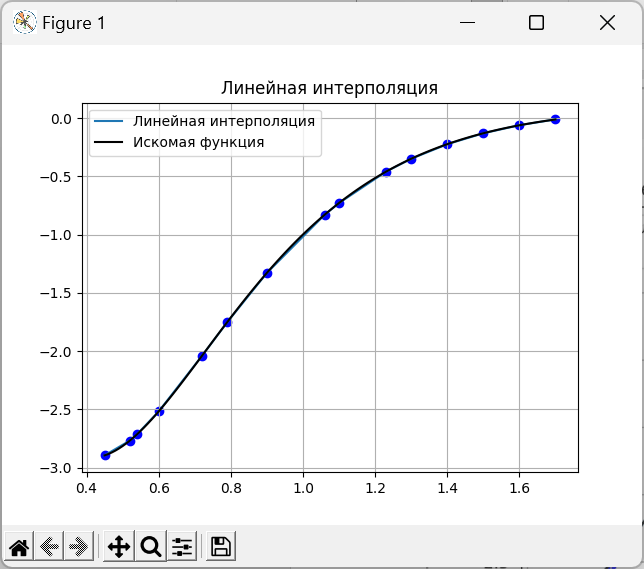


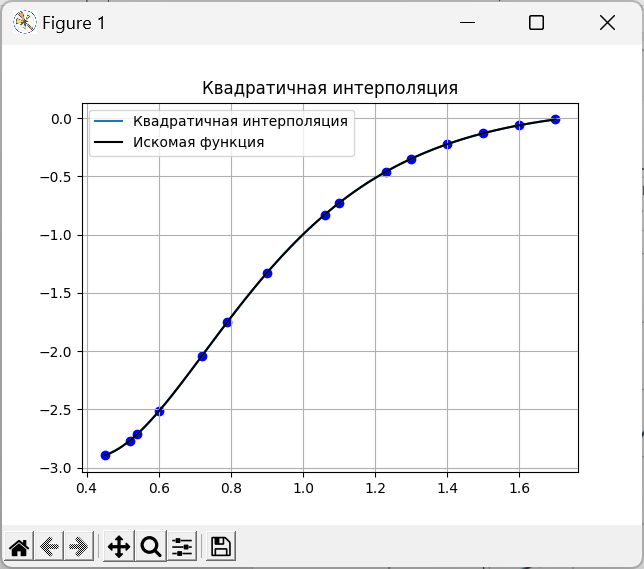
# Анализ метода

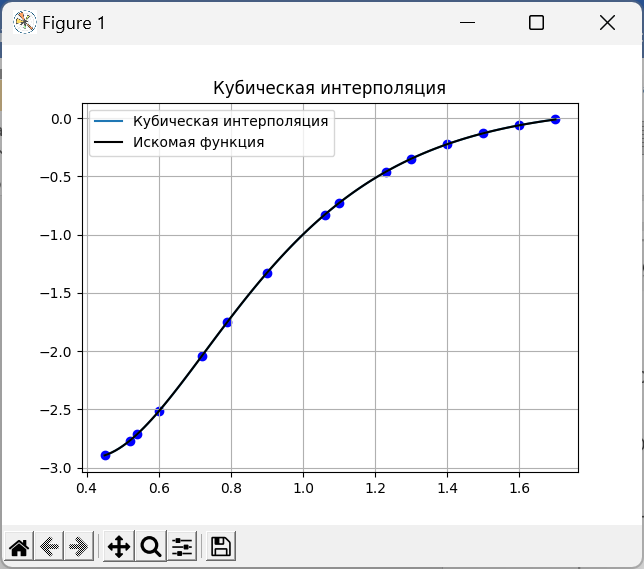
В качестве анализа метода возьмем возрастающую функцию, например, на промежутке

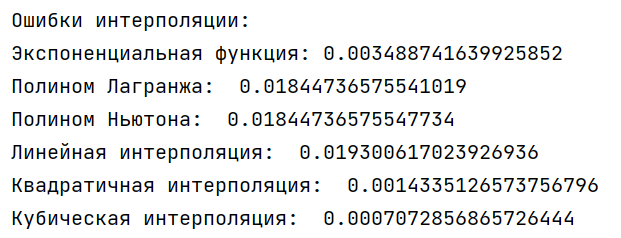
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0.45 | 0.52 | 0.54 | 0.6 | 0.72 | 0.79 | 0.9 | 1.06 | 1.1 | 1.23 | 1.3 | 1.4 | 1.5 | 1.6 | 1.7 |
| -2.896 | -2.769 | -2.715 | -2.519 | -2.042 | -1.753 | -1.33 | -0.831 | -0.73 | -0.461 | -0.35 | -0.225 | -0.131 | -0.062 | -0.012 |











# Прикладные задачи

В настоящее время имеется значительное число работ, посвященных численным аспектам интерполяции сплайнами функций одного переменного, связанных с локальным наследованием сплайном свойств монотонности и выпуклости исходных данных.

Сплайн можно использовать для уменьшения веса хранимых данных. Легче хранить коэффициенты интерполяции, чем всю таблицу точек. Например, если нужно хранить форму крыла, которая имеет вид, похожий на функцию экспоненты.

Формулу просто интегрировать, находить производную. К ней можно применять математические методы для исследования. Это следует из свойств функции экспоненты.

Также важной областью применения алгоритма является робототехника, прежде всего – сенсорные устройства. Обработка сигналов от датчиков обычно происходит с применением градуировочных функций, определяющих зависимость электрического сигнала от измеряемых физических величин внешнего пространства. Известно, что для обработки экспериментальных данных выбор метода интерполяции целесообразно осуществлять с учетом условий, характерных для данного анализируемого эксперимента: характер изменения данных, количество точек, точность измерения. Для градуировочных функций типично, что природа нам предоставляет монотонно возрастающие зависимости. Однако возрастание может быть неравномерным и переходящим в асимптотическое приближение к некоторому предельному уровню. Если на таких участках получены редкие исходные данные, то обычные способы построения интерполяционной кривой с помощью полиномов дадут неверную картину с выпуклостями. В связи с этим заслуживают внимание экспоненты.

# Вывод

В ходе данной работы я ознакомился с еще одним возможным способом интерполяции, конкретно с помощью экспоненциальной функции. Мною был сделан интерфейс программы, вывод графиков на языке Python с помощью библиотек tkinter и matplotlib.

Как показал анализ метода, интерполяция экспонентой дает неплохие показатели. Судя по ошибке интерполяции, экспонента может сравниться с квадратичной интерполяцией. Лучше всего из вышеприведенных алгоритмов справилась кубическая интерполяция. Линейная интерполяция, интерполяция многочленом Лагранжа и Ньютона дали ошибку больше.