|  |  |
| --- | --- |
| Изображение выглядит как текст, керамические изделия, фарфор  Автоматически созданное описание | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

Факультет «Фундаментальные науки»

Кафедра «Техническая физика»

**Лабораторная работа №4**

по курсу «Вычислительная физика»

Выполнили: Коберник Т. Н., Плетенёв Б. А.

Группа: ФН4-71Б

Преподаватели: Хасаншин Р.Х., Ивлиев П.А.

Москва, 2022 г.

Содержание

[1. Теоретическая часть 3](#_Toc116240698)

[1.1 Область применимости сплайн интерполяции 3](#_Toc116240699)

2.1. Кубический сплайн …………………………………………….4

[2. Постановка задачи 7](#_Toc116240700)

3. Программа…………………………………………………….……..7

[4. Решение 10](#_Toc116240701)

[5. Выводы 12](#_Toc116240702)

1. **Теоретическая часть**

## **Область применимости сплайн интерполяции**

Приближение функции с помощью интерполяционных многочленов Лагранжа и Ньютона на больших отрезках связан с большими трудностями.

Во-первых, интерполяционный многочлен высокой степени претерпевает большие колебания в окрестностях периферийных и узловых точек, поэтому невозможно устремить n к бесконечности для решения задачи интерполяции на большом отрезке.

Во-вторых, невозможно осуществить приближение функции с помощью многочлена с большим шагом, поскольку это приводит к недопустимому росту погрешности.

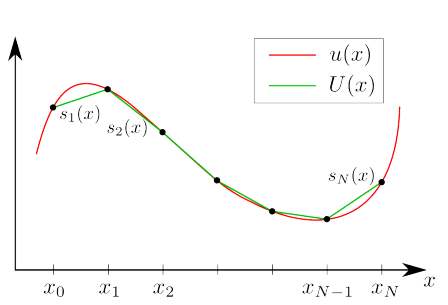
И наконец, в-третьих, использовать для приближения разбиение большого отрезка на несколько отрезков меньшего размера (с последующим построением на каждом отрезке отдельного многочлена) так же не представляется возможным, так как на стыках отрезков финальная интерполяционная функция будет претерпевать разрыв первой производной.

Поэтому для проведения интерполяции на больших отрезках используется другой метод – сплайн интерполяцию, требующую от функции меньшей гладкости.

Сплайн – функция, непрерывная на всём отрезке , а внутри каждого частичного отрезка , – представляющая собой алгебраический многочлен

Степень сплайна – максимальная степень многочленов по всем частичным отрезкам.

Дефект сплайна – разность между наивысшим порядком непрерывной на производной и степенью сплайна.



*Рис.1. Графическое представление сплайн интерполяции. U(x) – приближенная функция, u(x) – искомая функция*

* 1. **Кубический сплайн**

Кубический сплайн, принимающий в узлах  значения , имеет на частичном отрезке  вид

|  |  |
| --- | --- |
| (1) |  |

Нетрудно заметить, что для такого многочлена будет верно:

Утверждение. Любой многочлен, который в точке принимает значение , а его первая производная равно , полностью совпадает с многочленом (1).

Чтобы построить нужны значения функции и её наклоны в узловых точках.

Способы задания наклонов кубического сплайна:

1. *Упрощенный способ*.

Полагают

, (2)

, . (3)

Формулы (2), (3) являются формулами численного дифференцирования второго порядка точности относительно *h* = (*b-a*)/*N*.

**II**) Если известны  – значения производной  в узлах , то полагают



Способы А) и Б) называются ***локальными способами*** *задания наклонов*, поскольку с их помощью на каждом частичном отрезке  сплайн строится отдельно (непосредственно по формуле (1)). При этом соблюдается непрерывность в узлах  производной , а непрерывность второй производной  в узлах сплайна не гарантируется. Поэтому дефект такого сплайна обычно равен двум.

**III)** Попробуем обеспечить непрерывность второй производной. Для этого

Приравнивая выражения получим:

Чтобы замкнуть СЛАУ (4) надо ввести два дополнительных условия. Т.к. они, как правило, связаны с и , их называют краевыми условиями.

Рассмотрим 3 случая:

1. Предположим, что заданы производные функции в точках a, b, тогда
2. Формула численного дифференцирования 3 порядка точности
3. Предположим, что заданы вторые производные функции в точках a, b:

Тогда

Краевые условия (5)-(7) можно использовать независимо.

Утверждение. Если наклоны интерполяционного кубического сплайна заданы способами I) или II), то имеет место

Выводы:

1. Если , то есть k=3, а наклоны найдены способом III), то отклонение от , от , от порядка соответственно.
2. Если наклоны найдены способом II) при k=3, то скачки второй производной в узловых точках не превышают .
3. Неравенство, справедливое при , относится к I) способу определения наклонов.
4. **Постановка задачи**

Построить графики интерполяционных многочленов, приближающих функцию Рунге*f*(*x*) =1/(1+25*х*2) на отрезке [-1,1], при разных *N*, совпадающих со степенями интерполяционных многочленов в задании №2 соответствующей «Пары» (в задании №4 *N* – число частичных отрезков). Сравнить с результатами лаб. работы №2.

Из условия лабораторной №2:

N =5, 8, 15 и 16 (вариант 6)

1. **Программа**
2. **import** numpy as np
3. **import** matplotlib.pyplot as plt
4. **import** math

7. **def** Runge(x):
8. **return** 1 **/** (1 **+** 25 **\*** x **\*** x)

11. **def** Runge\_dif(x):
12. **return** **-**50 **\*** x **/** ((1 **+** 25 **\*** x **\*\*** 2) **\*\*** 2)

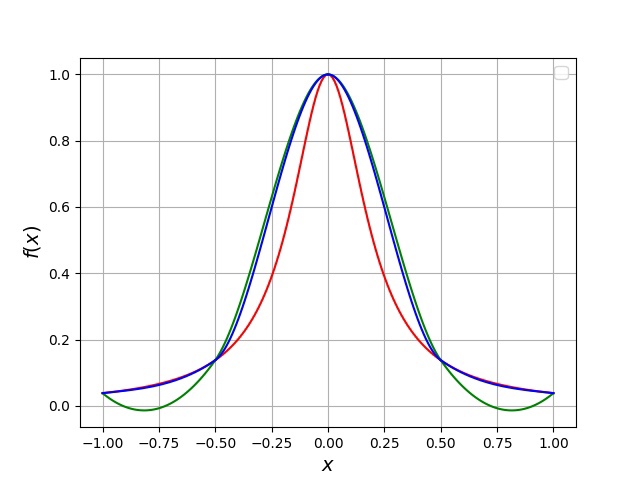
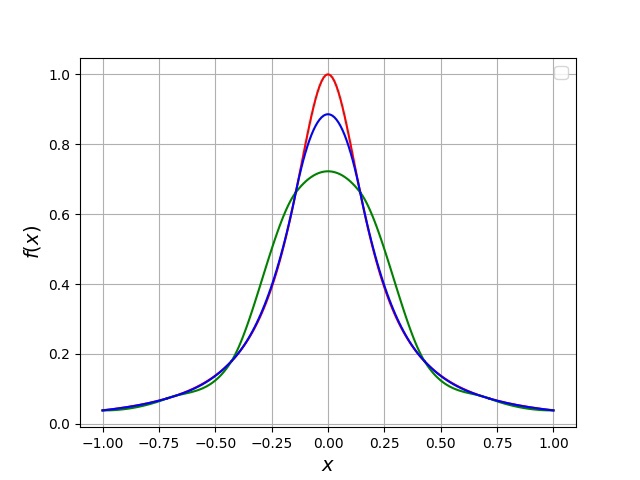
15. **def** Spline\_first(x, x1, flager):
16. h **=** x1[1] **-** x1[0]
17. **if** flager **==** 0:
18. m\_i **=** (**-**3 **\*** Runge(x1[0]) **+** 4 **\*** Runge(x1[1]) **-** Runge(x1[2])) **/** (2 **\*** h)
19. m\_i\_next **=** (Runge(x1[2]) **-** Runge(x1[0])) **/** (2 **\*** h)
20. i **=** 0
21. **elif** flager **==** 2:
22. m\_i\_next **=** (3 **\*** Runge(x1[2]) **-** 4 **\*** Runge(x1[1]) **+** Runge(x1[0])) **/** (2 **\*** h)
23. m\_i **=** (Runge(x1[2]) **-** Runge(x1[0])) **/** (2 **\*** h)
24. i **=** 1
25. **else**:
26. m\_i **=** (Runge(x1[2]) **-** Runge(x1[0])) **/** (2 **\*** h)
27. m\_i\_next **=** (Runge(x1[3]) **-** Runge(x1[1])) **/** (2 **\*** h)
28. i **=** 1
30. S3 **=** (
31. ((x1[i **+** 1] **-** x) **\*\*** 2) **\*** (2 **\*** (x **-** x1[i]) **+** h) **\*** Runge(x1[i]) **/** (h **\*\*** 3)
32. **+** ((x **-** x1[i]) **\*\*** 2) **\*** (2 **\*** (x1[i **+** 1] **-** x) **+** h) **\*** Runge(x1[i **+** 1]) **/** (h **\*\*** 3)
33. **+** ((x1[i **+** 1] **-** x) **\*\*** 2) **\*** (x **-** x1[i]) **\*** m\_i **/** (h **\*\*** 2)
34. **+** ((x **-** x1[i]) **\*\*** 2) **\*** (x **-** x1[i **+** 1]) **\*** m\_i\_next **/** (h **\*\*** 2)
35. )
37. **return** S3

40. **def** Spline\_second(x, x1):
41. h **=** x1[1] **-** x1[0]
43. m\_i **=** Runge\_dif(x1[0])
44. m\_i\_next **=** Runge\_dif(x1[1])
45. S3 **=** (
46. ((x1[1] **-** x) **\*\*** 2) **\*** (2 **\*** (x **-** x1[0]) **+** h) **\*** Runge(x1[0]) **/** (h **\*\*** 3)
47. **+** ((x **-** x1[0]) **\*\*** 2) **\*** (2 **\*** (x1[1] **-** x) **+** h) **\*** Runge(x1[1]) **/** (h **\*\*** 3)
48. **+** ((x1[1] **-** x) **\*\*** 2) **\*** (x **-** x1[0]) **\*** m\_i **/** (h **\*\*** 2)
49. **+** ((x **-** x1[0]) **\*\*** 2) **\*** (x **-** x1[1]) **\*** m\_i\_next **/** (h **\*\*** 2)
50. )
51. **return** S3

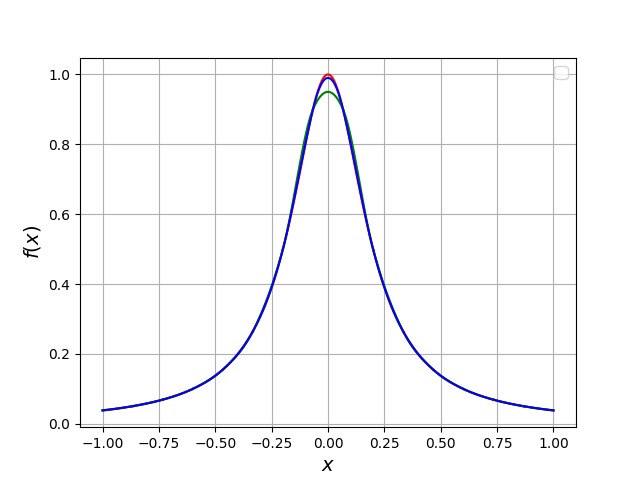
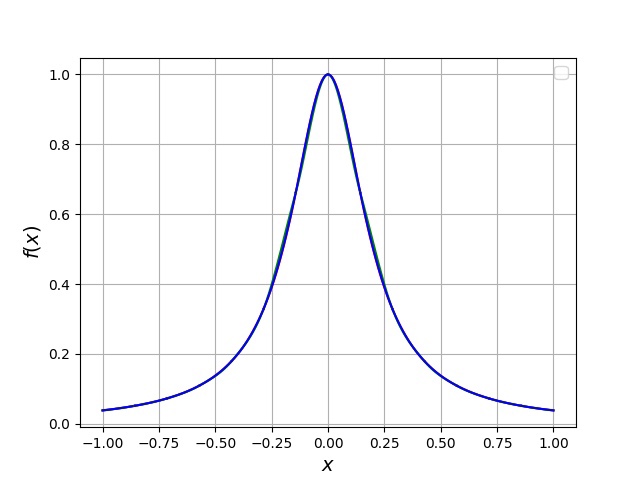
54. x **=** np.linspace(**-**1, 1, 500)
55. k **=** [5, 8, 15, 16]
57. **for** jj **in** k:
58. x1 **=** np.linspace(**-**1, 1, jj)
59. **for** kk **in** range(0, jj **-** 1):
60. x **=** np.linspace(x1[kk], x1[kk **+** 1], round(500 **/** jj))
61. plt.plot(x, np.array([Runge(ii) **for** ii **in** x]), color**=**"r")
62. **if** kk **==** 0:
63. flager **=** 0
64. buff **=** x1[:3]
65. **elif** kk **==** jj **-** 2:
66. buff **=** x1[**-**3:]
67. flager **=** 2
68. **else**:
69. buff **=** x1[kk **-** 1 : kk **+** 3]
70. flager **=** 1
71. plt.plot(x, np.array([Spline\_first(ii, buff, flager) **for** ii **in** x]), color**=**"g")
72. plt.plot(
73. x,
74. np.array([Spline\_second(ii, [x1[kk], x1[kk **+** 1]]) **for** ii **in** x]),
75. color**=**"b",
76. )
77. plt.xlabel(r"$x$", fontsize**=**14)
78. plt.ylabel(r"$f(x)$", fontsize**=**14)
79. plt.grid(True)
80. plt.legend(loc**=**"best", fontsize**=**12)
81. plt.show()
82. **Результаты**

В ходе работы приведенной выше программы были получены следующие результаты:

* Сплайн функции, рассчитанные с применением первого способа определения наклонов изображены на графике зелёными линиями, вторым способом – синими. График исходной функции изображен красным

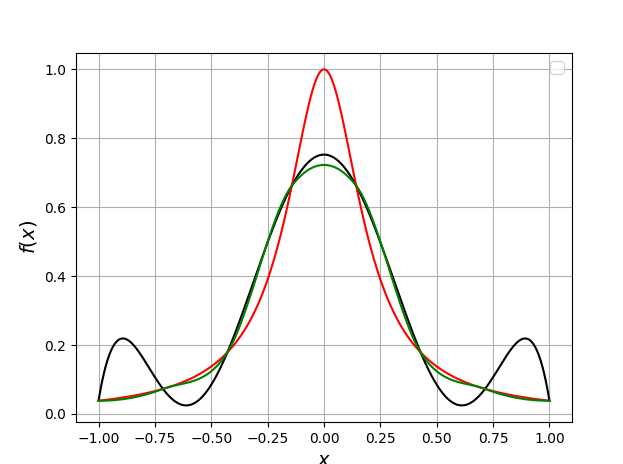
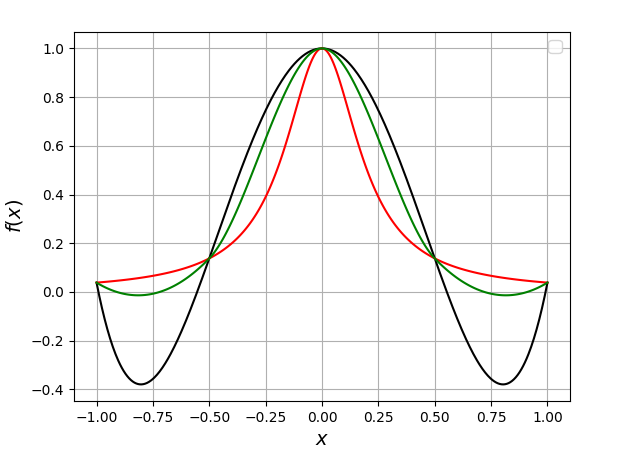
А) Б)



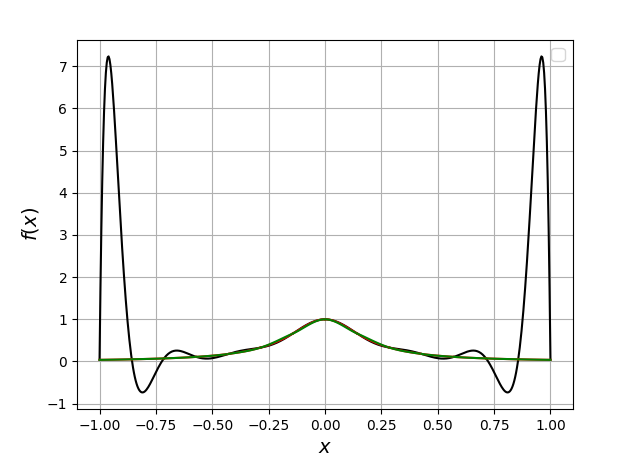
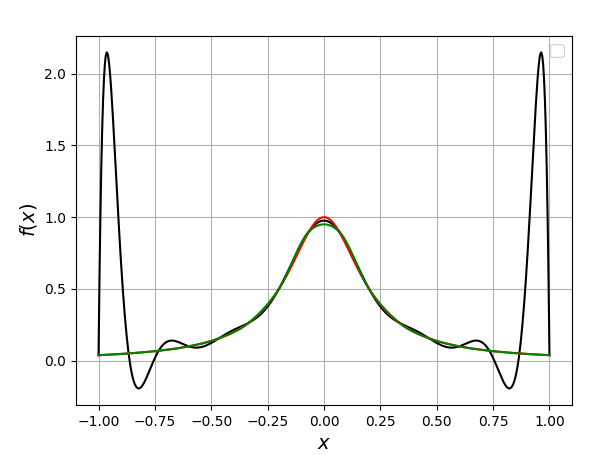
В) Г)

*Рис. 2. Графики сплайн многочленов с А) N=5, Б) N=8, В) N=15, Г) N=16*

Далее приведем сравнение сплайн многочлена (зелёным) с многочленом Ньютона (черным), полученным в лабораторной №2. Во избежание загромождения чертежа, рассмотрим только ту сплайн кривую, которая была получены первым способом



А) Б)

В) Г)

*Рис. 3. Графики сплайн многочленов и многочленов Ньютона с А) N=5, Б) N=8, В) N=15, Г) N=16*

1. **Выводы**

В данной лабораторной работе была проведена кубическая сплайн интерполяция функции Рунге. При этом наклоны функции были найдены двумя различными способами. Как видно из полученных графиков, первый способ даёт более точные результаты в окрестности середины рассмотренного отрезка, а второй способ – на концах отрезка.

Так же можно заметить, что с увеличением разбиения отрезка, увеличивается и точность интерполяции.

При сравнении сплайн интерполяции и интерполяции методом Ньютона было выяснено, что сплайн даёт значительно более точные результаты на концах отрезка, т.к. в отличие от многочлена Ньютона не имеет колебательных погрешностей вблизи начальной и конечной точек.