|  |  |  |  |
| --- | --- | --- | --- |
| Министерство образования Республики Беларусь  Учреждение образования  Белорусский Государственный Университет Информатики и Радиоэлектроники | | | |
| Факультет | Компьютерных сетей и систем | | |
| Кафедра | Информатики  Дисциплина: Конструирование те технологии электронных вычислительных средств | | |
|  |  | | |
| **Отчет ПО Лабораторной Работе**  по курсу «Системы аналитического программирования» | | | |
| Студент:  гр. 758641  Ярош Г.И. | |  | Проверил:  Герман О.В. |
| Минск, 2018 | | | |

# ИНТЕРПОЛЯЦИЯ ФУНКЦИЙ

## Цель

Изучить возможности языка Python и библиотеки SciPy для интерполяции функции. Реализовать интерполирование следующей функции:

## Теоретическая часть

Интерполяция функции представляет собой процесс восстановления функции по ограниченному набору значений. Результатом интерполяции является функция, аппроксимирующая исходную. В качестве функции интерполяции обычно используются сплайны.

В данной работе рассматривались три вида сплайнов:

1. - линейный сплайн;
2. - квадратный сплайн;
3. - кубический сплайн;

Процесс интерполяции заключается в подборе коэффициентов , , , . Это происходит путем решения системы уравнения .

## Реализация

Для построения функции, интерполирующей исходную, пакет SciPy предоставляет модуль interpolate. В нем определена функция interp1d, которая принимает массив значений *x,* значений функции *f(x)* и вид сплайна: linear для линейного сплайна, quadratic для квадратного, а qubic для кубического сплайна.

Исходный код программы приведен ниже:

*# -\*- coding: utf-8 -\*-*import numpy as np  
from matplotlib import pyplot as plt  
from scipy.interpolate import interp1d  
  
  
*# исходная функция*def f(x):  
 return np.sin(x \*\* 2) + np.cos(x \*\* 2)  
  
  
def main():  
 *# инициализация вектора иксов и подсчет значений f(x)* x = np.linspace(-4, 4, 20)  
 y = f(x)  
   
 *# получение интерполяций функции с помощью трех видов сплайна* f1 = interp1d(x, y, kind='slinear')  
 f2 = interp1d(x, y, kind='quadratic')  
 f3 = interp1d(x, y, kind='cubic')  
  
 *# генерация точек для графиков* xx = np.linspace(-4, 4, 1000)  
   
 *# график исходной функции* fig = plt.figure()  
 ax1 = fig.add\_subplot(2, 2, 1)  
 ax1.set\_title('Initial Function')  
 ax1.plot(xx, f(xx), linewidth=2)  
  
 *# график интерполяции с помошью линейного сплайна* ax2 = fig.add\_subplot(2, 2, 2)  
 ax2.set\_title('Linear Interpolation')  
 ax2.plot(xx, f(xx), '-.', color=(0, 0, 0, 0.5))  
 ax2.plot(xx, f1(xx), linewidth=1, color=(0.5, 0, 0))  
  
 *# график интерполяции с помощью квадратного сплайна* ax3 = fig.add\_subplot(2, 2, 3)  
 ax3.set\_title('Quadratic Interpolation')  
 ax3.plot(xx, f(xx), '-.', color=(0, 0, 0, 0.5))  
 ax3.plot(xx, f2(xx), linewidth=1, color=(0, 0.5, 0))  
  
 *# график интерполяции с помошью кубического сплайна* ax4 = fig.add\_subplot(2, 2, 4)  
 ax4.set\_title('Cubic Interpolation')  
 ax4.plot(xx, f(xx), '-.', color=(0, 0, 0, 0.5))  
 ax4.plot(xx, f3(xx), linewidth=1, color=(0, 0, 0.5))  
  
 *# вывод графиков на экран* plt.show()

if \_\_name\_\_ == '\_\_main\_\_':  
 main()

Полученные графики приведены на рисунке 1.

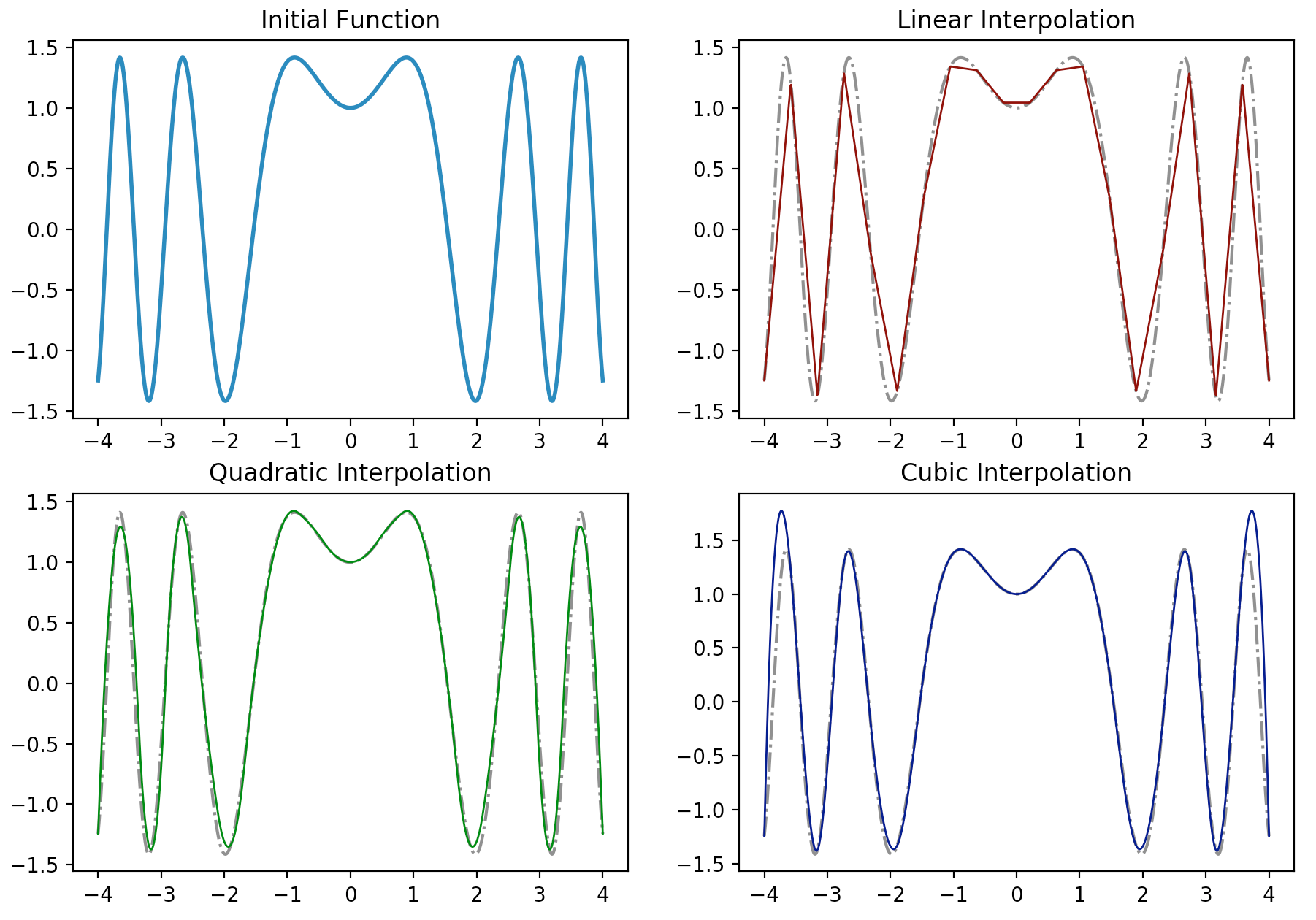


Рис. 1. Графики исходной функции и ее интерполяций.