МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ордена Трудового Красного Знамени федеральное государственное бюджетное образовательное учреждение высшего образования

**«Московский технический университет связи и информатики»**

Кафедра «Информатика»

**Отчет по заданию №3**

**по дисциплине**

**«Численные методы»**

Выполнил: студент гр. БЭИ2202

Васильев В. А.

Вариант 8.

Проверил: доц. каф. «Информатика»

Мацкевич А. Г.

Москва, 2023 г.

* 1. **Задание для решения задачи аппроксимации**

Для решения задачи аппроксимации методом наименьших квадратов выберем функцию y(x), заданную следующей таблицей:

Изображение выглядит как текст, Шрифт, снимок экрана, типография

Автоматически созданное описание

* 1. **Линейная аппроксимация:**

Вычислить и записать в таблицу элементы матрицы Грамма и столбец свободных членов:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Составить системы нормальных уравнений:

Для линейной функции **P1(x) = А0+А1\*x** система нормальных уравнений примет вид (линейная аппроксимация):

**6\*А0+0.6\*А1 = -0.06**

**-0.6\*А0+0.76\*А1 = 1.715**

Решив систему уравнений, получим коэффициенты **А0 = 0.234** и **А1 = -2.442**, тогда полином первой степени будет таким:

**P1(x) = 2.443x + 0.234**

* 1. **Линейная аппроксимация:**

Решим задачу аппроксимации при помощи ЯП Python.

*# Задаем исходные данные*

x = np.array([-0.6, -0.4, -0.2, 0.0, 0.2, 0.4])

y = np.array([-1.06, -1.406, -0.386, 1, 1.386, 0.406])

degrees = [1, 2, 3, 4, 5]

fig = go.Figure() *# layout\_xaxis\_range = [-2,6]*

fig.add\_trace(go.Scatter(x=x,

y=y,

mode='markers',

marker=dict(color='LightSkyBlue', size=15, line=dict(color='MediumPurple', width=3)),

name="Исходные данные"))

**for** degree **in** degrees:

coefficients = np.polyfit(x, y, degree)

poly = np.poly1d(coefficients)

**print**(coefficients)

**print**(poly)

*# Генерируем значения для построения графика*

x\_plot = np.linspace(x.min(), x.max(), 100)

y\_plot = poly(x\_plot)

*# Вычисляем среднеквадратичное отклонение*

y\_pred = poly(x)

sko = np.sqrt(np.mean((y - y\_pred) \*\* 2))

*# Строим график и выводим СКО*

fig.add\_trace(go.Scatter(x=x\_plot,

y=y\_plot,

name=f'Степень {degree}, СКО: {sko:.2f}'))

fig.update\_layout(title="Построение графика аппроксимирующих полиномов",

xaxis\_title="Значение аргумента",

yaxis\_title="Значение полинома")

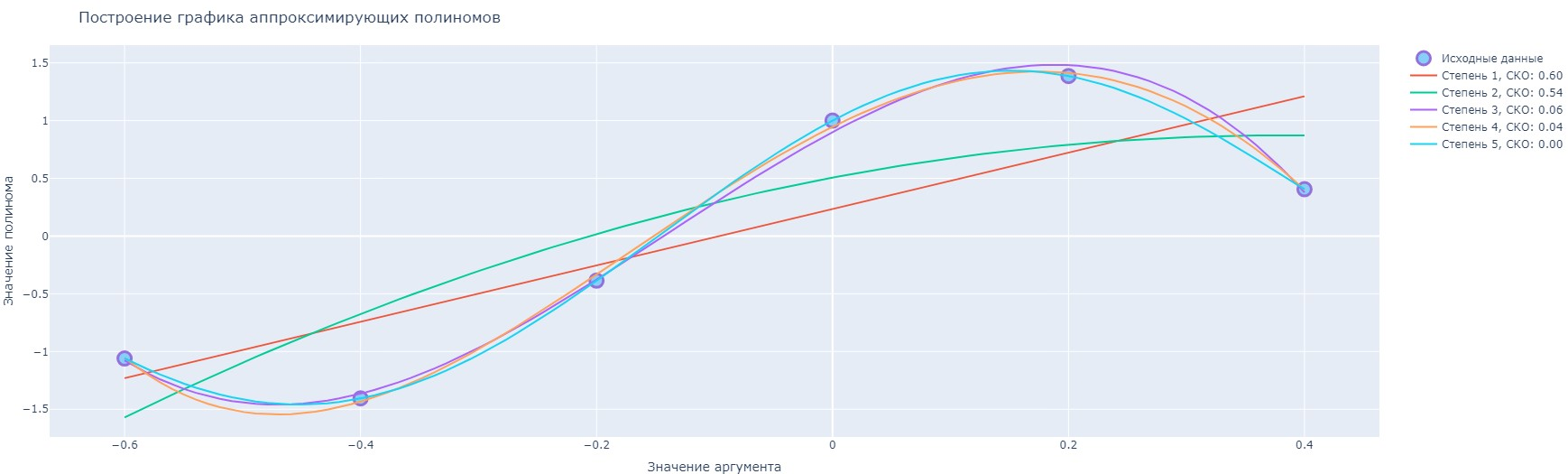


Рисунок 1 – Полученные полиномы и их СКО

Полученные в ходе аппроксимации многочлены:

;