Московский Авиационный Институт(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №7 по курсу “Компьютерная Графика”

Студент: Т.А.Габдуллин

Преподаватель: Г. С. Филиппов

Группа: М8О-306Б

Оценка:

Подпись:

# Лабораторная работа №7

**Тема:**Построение плоских полиномиальных кривых.

**Задача:**Написать программу, строящую полиномиальную кривую по заданным точкам. Обеспечить возможность изменения позиции точек и, при необходимости, значений касательных векторов и натяжения

**Вариант:** Интерполяционный многочлен Лагранжа по пяти точкам.

Исходныйкод

**import sys**

**from PyQt5.QtWidgets import QDialog, QApplication, QPushButton, QVBoxLayout,\**

**QLineEdit, QLabel, QHBoxLayout, QCheckBox**

**from matplotlib.backends.backend\_qt5agg\**

**import FigureCanvasQTAgg**

**from matplotlib.backends.backend\_qt5agg\**

**import NavigationToolbar2QT**

**from scipy.interpolate import lagrange**

**import matplotlib.pyplot**

**import numpy**

**class Window(QDialog):**

**def \_\_init\_\_(self, parent=None):**

**super(Window, self).\_\_init\_\_(parent)**

**self.title = QLabel('Lagrange Polynom')**

**self.figure = matplotlib.pyplot.figure(figsize=(4, 4))**

**self.canvas = FigureCanvasQTAgg(self.figure)**

**self.toolbar = NavigationToolbar2QT(self.canvas, self)**

**self.button = QPushButton('Plot')**

**self.button.clicked.connect(self.plot)**

**self.check = QCheckBox('Tangent')**

**self.check.stateChanged.connect(self.tang)**

**self.lbl1\_0 = QLabel('Point 1')**

**self.lbl1\_1 = QLabel('X')**

**self.line1\_1 = QLineEdit()**

**self.lbl1\_2 = QLabel('Y')**

**self.line1\_2 = QLineEdit()**

**self.lbl2\_0 = QLabel('Point 2')**

**self.lbl2\_1 = QLabel('X')**

**self.line2\_1 = QLineEdit()**

**self.lbl2\_2 = QLabel('Y')**

**self.line2\_2 = QLineEdit()**

**self.lbl3\_0 = QLabel('Point 3')**

**self.lbl3\_1 = QLabel('X')**

**self.line3\_1 = QLineEdit()**

**self.lbl3\_2 = QLabel('Y')**

**self.line3\_2 = QLineEdit()**

**self.lbl4\_0 = QLabel('Point 4')**

**self.lbl4\_1 = QLabel('X')**

**self.line4\_1 = QLineEdit()**

**self.lbl4\_2 = QLabel('Y')**

**self.line4\_2 = QLineEdit()**

**self.lbl5\_0 = QLabel('Point 5')**

**self.lbl5\_1 = QLabel('X')**

**self.line5\_1 = QLineEdit()**

**self.lbl5\_2 = QLabel('Y')**

**self.line5\_2 = QLineEdit()**

**layout1 = QHBoxLayout()**

**layout2 = QHBoxLayout()**

**layout3 = QHBoxLayout()**

**layout4 = QHBoxLayout()**

**layout5 = QHBoxLayout()**

**layout6 = QHBoxLayout()**

**layout = QVBoxLayout()**

**layout.addWidget(self.title)**

**layout.addWidget(self.toolbar)**

**layout.addWidget(self.canvas)**

**layout1.addWidget(self.button)**

**layout1.addWidget(self.check)**

**layout2.addWidget(self.lbl1\_0)**

**layout2.addWidget(self.lbl1\_1)**

**layout2.addWidget(self.line1\_1)**

**layout2.addWidget(self.lbl1\_2)**

**layout2.addWidget(self.line1\_2)**

**layout3.addWidget(self.lbl2\_0)**

**layout3.addWidget(self.lbl2\_1)**

**layout3.addWidget(self.line2\_1)**

**layout3.addWidget(self.lbl2\_2)**

**layout3.addWidget(self.line2\_2)**

**layout4.addWidget(self.lbl3\_0)**

**layout4.addWidget(self.lbl3\_1)**

**layout4.addWidget(self.line3\_1)**

**layout4.addWidget(self.lbl3\_2)**

**layout4.addWidget(self.line3\_2)**

**layout5.addWidget(self.lbl4\_0)**

**layout5.addWidget(self.lbl4\_1)**

**layout5.addWidget(self.line4\_1)**

**layout5.addWidget(self.lbl4\_2)**

**layout5.addWidget(self.line4\_2)**

**layout6.addWidget(self.lbl5\_0)**

**layout6.addWidget(self.lbl5\_1)**

**layout6.addWidget(self.line5\_1)**

**layout6.addWidget(self.lbl5\_2)**

**layout6.addWidget(self.line5\_2)**

**layout.addLayout(layout1)**

**layout.addLayout(layout2)**

**layout.addLayout(layout3)**

**layout.addLayout(layout4)**

**layout.addLayout(layout5)**

**layout.addLayout(layout6)**

**self.setLayout(layout)**

**self.x = [None] \* 5**

**self.y = [None] \* 5**

**self.f = None**

**def tang(self):**

**if self.check.isChecked():**

**f\_derivative = self.f.deriv()**

**for i in self.x:**

**lol = [i - 2, i + 2]**

**kek = [self.f(i) + f\_derivative(i) \* (j - i) for j in lol]**

**matplotlib.pyplot.plot(lol, kek,"red")**

**self.canvas.draw()**

**else:**

**self.plot()**

**def plot(self):**

**try:**

**self.x[0] = int(self.line1\_1.text())**

**self.y[0] = int(self.line1\_2.text())**

**self.x[1] = int(self.line2\_1.text())**

**self.y[1] = int(self.line2\_2.text())**

**self.x[2] = int(self.line3\_1.text())**

**self.y[2] = int(self.line3\_2.text())**

**self.x[3] = int(self.line4\_1.text())**

**self.y[3] = int(self.line4\_2.text())**

**self.x[4] = int(self.line5\_1.text())**

**self.y[4] = int(self.line5\_2.text())**

**except ValueError:**

**return**

**self.f = lagrange(self.x, self.y)**

**xnew = numpy.linspace(min(self.x) - 2, max(self.x) + 2, 500)**

**ax = self.figure.add\_subplot(111)**

**ax.set\_ylim(min(self.y) - 50, max(self.y) + 50)**

**self.figure.clear()**

**matplotlib.pyplot.plot(self.x, self.y, 'o', xnew, self.f(xnew), 'black')**

**self.canvas.draw()**

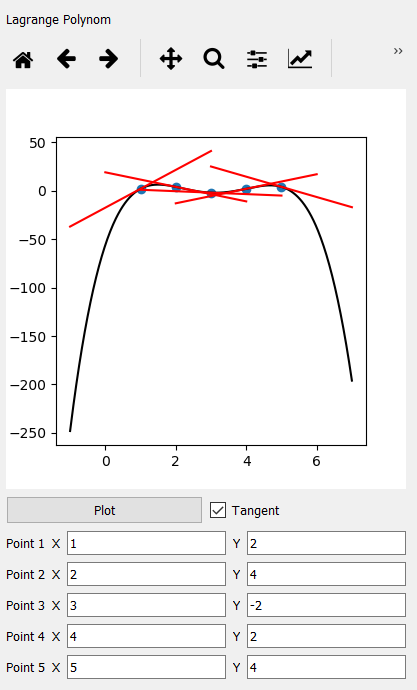
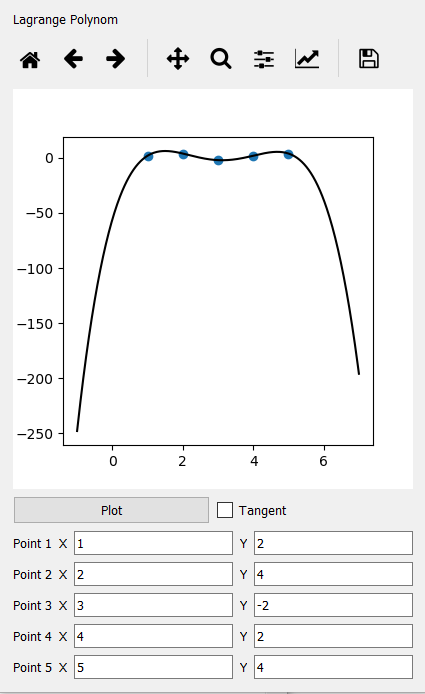
**if \_\_name\_\_ == '\_\_main\_\_':**

**app = QApplication(sys.argv)**

**main = Window()**

**main.show()**

# Скриншоты



# Выводы

Выполнив 7ую лабораторную, я научился строить полиномиальные кривые по заданным точкам. Изучил на практике построение полинома Лагранжа по 5 точкам, с отрисовкой касательных