Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Алгоритми та методи обчислення

Лабораторна робота №4

**«Розв’язання нелінійних рівнянь на комп’ютері»**

Виконала:

студентка групи ІО-64

Бровченко А. В.

Перевірив:

ст.вик. Порєв В. М.

Київ

2018 р.

**Мета:** ознайомлення з методиками та вивчення різних алгоритмів розв’язання нелінійних рівнянь на комп’ютері.

**Завдання:** Закріплення знань студентів при вирішенні практичних завдань з розв’язування нелінійних рівнянь. Оволодіння методами і практичними навичками розв’язування нелінійних рівнянь на комп’ютері. Набуття умінь і навичок при програмуванні та налагодженні програм для розв’язування нелінійних рівнянь на комп'ютері.

**За варіантом:** метод половинного ділення.



**Роздруківка тексту програми:**

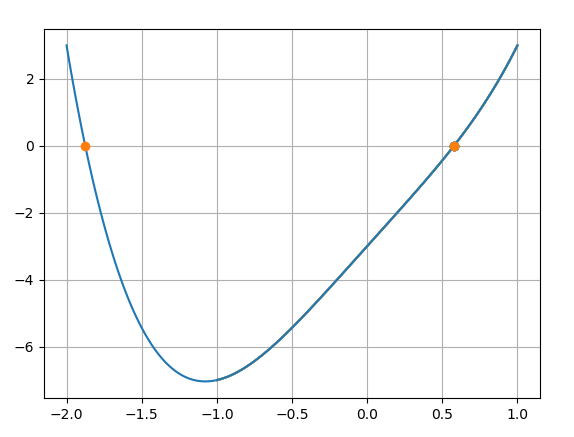
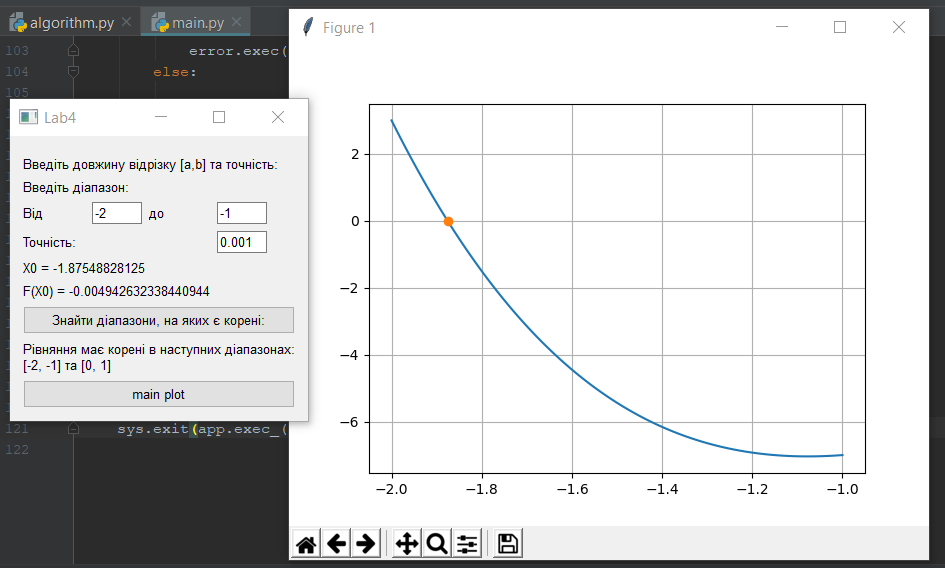
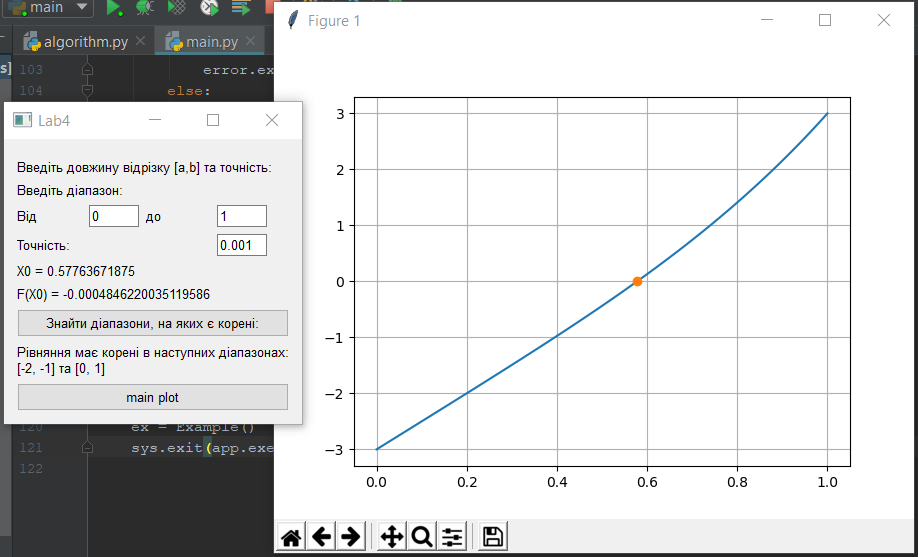
algorythm.py

import matplotlib.pyplot as plt  
  
from scipy.interpolate import lagrange  
from numpy.polynomial.polynomial import Polynomial  
import numpy as np  
  
  
def df(x):  
 return 4 \* x \*\* 3 + 5  
  
  
def function(x):  
 return x\*\*4 + 5 \* x - 3  
  
  
def find\_ranges():  
 a = -100  
 b = 100  
 step = 1  
 n = (b-a)/step  
  
 c = a  
  
 b = a+step  
 kranges = []  
 for i in range(int(n)):  
  
 fa = function(a)  
 fb = function(b)  
 if fa\*fb < 0:  
 kranges.append([a, b])  
 a = b  
 b = b+step  
  
 return kranges  
  
  
def bisection(a, b, e):  
  
 n = 0  
 while not abs(b - a) <= e:  
  
 c = (a + b) / 2  
  
 if function(c) == 0:  
 return c, function(c)  
  
 f1 = function(a)  
 f2 = function(c)  
 if f1\*f2<=0:  
 b = c  
 else:  
 a = c  
  
 n+=1  
  
 #print(n, abs(b-a))  
 c = (a + b) / 2  
 #print(c, function(c))  
 return c, function(c)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
  
 a = -2  
 b = 2  
 e = 0.001  
 x0, y0 = bisection(a,b,e)  
  
  
 print(find\_ranges())  
  
 x = np.linspace(-2, 2, 100)  
 y = np.array([function(i) for i in x])  
  
 dy = df(x)  
  
 plt.plot(x, y,[x0],[y0],'o',[a],[0],'|',[b],[0],'|', x, dy)  
 plt.grid(True)  
 plt.show()

main.py

import sys  
  
import matplotlib.pyplot as plt  
import numpy as np  
from PyQt5.QtGui import \*  
from PyQt5.QtWidgets import \*  
from third\_sem.AMO.Lab4.algorithm import function, bisection, find\_ranges  
  
  
# noinspection PyUnresolvedReferences,PyArgumentList  
class Example(QWidget):  
  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
  
 self.initUI()  
  
 def initUI(self):  
 grid = QGridLayout()  
 self.setLayout(grid)  
 self.move(300, 150)  
 self.setWindowTitle('Lab4')  
 self.font = QFont('Arial', 12)  
 #self.font.setPointSize(11)  
  
 self.empty = QLabel('')  
 grid.addWidget(self.empty, 0, 0, 0, 8)  
  
 self.a = QLineEdit('0')  
 self.b = QLineEdit('1')  
 self.a.setMaximumSize(50, 50)  
 self.b.setMaximumSize(50, 50)  
  
 self.l1 = QLabel("Введіть діапазон:")  
  
 self.lable\_from = QLabel('Від')  
 self.lable\_to = QLabel('до')  
  
 self.nlable = QLabel('Кількість вузлів:')  
 self.n = QLineEdit('10')  
 self.n.setMaximumSize(50, 50)  
  
 self.tlable = QLabel('Точність:')  
 self.t = QLineEdit('0.001')  
 self.t.setMaximumSize(50, 50)  
  
 self.xlable = QLabel('X0 = ?')  
 self.ylable = QLabel('F(X0) = ?')  
  
 self.find\_ranges\_but = QPushButton('Знайти діапазони, на яких є корені:')  
 self.find\_ranges\_but.clicked.connect(self.show\_ranges)  
 self.lranges = QLabel('')  
  
 button\_show\_ideal = QPushButton("main plot")  
 button\_show\_ideal.clicked.connect(self.showGraph)  
  
 grid.addWidget(self.l1, 2, 0, 1, 10)  
 grid.addWidget(self.lable\_from, 3, 0)  
 grid.addWidget(self.a, 3, 1)  
 grid.addWidget(self.lable\_to, 3, 2)  
 grid.addWidget(self.b, 3, 3)  
 grid.addWidget(self.tlable, 4, 0, 1, 3)  
 grid.addWidget(self.t, 4, 3)  
  
 grid.addWidget(self.xlable, 5, 0, 1, 3)  
 grid.addWidget(self.ylable, 6, 0, 1, 3)  
  
 grid.addWidget(self.find\_ranges\_but, 7, 0, 1, 10)  
 grid.addWidget(self.lranges, 8, 0, 1, 10)  
  
 grid.addWidget(button\_show\_ideal, 9, 0, 1, 10)  
 # grid.addWidget(button\_show\_error, 8, 0, 1, 10)  
  
 label2 = QLabel(self)  
 label2.setText("Введіть довжину відрізку [a,b] та точність:")  
 grid.addWidget(label2, 1, 0, 1, 10)  
 # label.setFont(self.font)  
 self.show()  
  
 def show\_ranges(self):  
 ranges = find\_ranges()  
 #print(ranges)  
 s = 'Рівняння має корені в наступних діапазонах:\n' \  
 '{0} та {1}'.format(ranges[0], ranges[1])  
 self.lranges.setText(s)  
  
 def getDots(self):  
 a = float(self.a.text())  
 b = float(self.b.text())  
 e = float(self.t.text())  
  
 self.x = np.linspace(a, b, 100)  
 self.y = list(map(function, self.x))  
  
 return a, b, e  
  
 def showGraph(self):  
 a, b, e = self.getDots()  
  
 if function(a) \* function(b) > 0:  
 error = QMessageBox()  
 error.setText('Invalid range')  
 error.exec()  
 else:  
  
 x0, y0 = bisection(a, b, e)  
  
 self.xlable.setText('X0 = {}'.format(x0))  
 self.ylable.setText('F(X0) = {}'.format(y0))  
  
 plt.plot(self.x, self.y)  
 plt.plot(x0, y0, 'o')  
  
 plt.grid(True)  
 plt.show()  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app = QApplication(sys.argv)  
 ex = Example()  
 sys.exit(app.exec\_())

**Скріншоти виконання програми:**



**Висновки:** У ході виконання лабораторної роботи я ознайомилась з методиками різних алгоритмів розв’язання нелінійних рівнянь на комп’ютері, реалізувала метод половинного ділення на мові Python.