НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ
“КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ ИМЕНИ ИГОРЯ СИКОРСКОГО”
ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ №1

Дисциплина: «Методы вычислений»

Тема: «Решение нелинейных уравнений»

Выполнила

студентка 3 курса группы ФИ-41

Лавягина Ольга Алексеевна

Проверил

Стёпочкина Ирина Валерьевна

Киев — 2017 года

1 ИСХОДНЫЕ ДАННЫЕ

В компьютерном практикуме (вариант 8) ищутся корни уравнения

$$2x^5 + 3x^2 - 2x - 6 = 0. \tag{1.1}$$

2 ДОПРОГРАММНЫЙ ЭТАП

Выпишем все коэффициенты при степенях переменной

$$a_0 = -6,\ a_1 = -2,\ a_2 = 3,\ a_3 = 0,\ a_4 = 0,\ a_5 = 2.$$

Проверим условие теоремы Гюа. Нужно проверить, существует ли k такое, что $0 < k < n$ и $a_k^2 \leq a_{k+1} \cdot a_{k-1}$, то существует хотя бы одна пара комплексно сопряжённых корней. Проверим это неравенство $a_1^2 = (-2)^2 = 4 > -6 \cdot 3 = -18$ — не подходит, $a_2^2 = 3^2 = 9 > -2 \cdot 0 = 0$ — не подходит, $a_3^2 = 0 = 3 \cdot 0 = 0$. Нашли, что для данного уравнения $k = 3$, то есть существует хотя бы одна пара комплексно сопряжённых корней.

Применим теорему про кольцо. Введём значения

$$A = \max\{|a_{n-1}|,\ |a_{n-2}|, \ldots, |a_0|\} = \max\{0, 0, 3, 2, 6\} = 6$$

и $B = \max\{|a_n|, \ldots, |a_1|\} = \max\{2, 0, 0, 3, 2\} = 3$.

Тогда

$$|x| < \frac{|a_n| + A}{|a_n|} = \frac{2+6}{2} = \frac{4}{2} = 2 = R$$

— больший радиус кольца.

С другой стороны

$$|x| > \frac{|a_0|}{|a_0| + B} = \frac{6}{6+3} = \frac{6}{9} = \frac{2}{3} = r$$

— меньший радиус кольца.

Получили, что

$$r = \frac{2}{3} < |x| < 2 = R.$$

Применим теорему про верхнюю границу. Введём величину

$$B = \max |a_i| = 6,\ m = \max i = 0,\ a_i < 0.$$

Тогда положительные корни ограничены сверху величиной

$$x^+ < 1 + \sqrt[n-m]{\frac{B}{a_n}} = 1 + \sqrt[5-0]{\frac{6}{2}} = 1 + \sqrt[5]{3} \approx 1 + 1.25 = 2.25.$$

Теорема про кольцо дала лучший результат на верхнюю границу положительных корней уравнения.

По теореме про верхнюю границу получили значение $R = 2.25$. Рассмотрим уравнение

$$P\left(\frac{1}{y}\right) = 2\left(\frac{1}{y}\right)^5 + 3\left(\frac{1}{y}\right)^2 - 2 \cdot \frac{1}{y} - 6 = 0.$$

Домножим это уравнение на y^5. Получаем

$$y^n P\left(\frac{1}{y}\right) = 2 + 3y^3 - 2y^4 - 6y^5 = 0.$$

Умножим полученное уравнение на (-1). Получим $6y^5 + 2y^4 - 3y^3 - 2 = 0$. Применим теорему про верхнюю границу. Введём величину

$$B = \max |a_i| = 3,\ m = \max i = 3,\ a_i < 0.$$

Тогда положительные корни ограничены сверху величиной

$$y^+ < 1 + \sqrt[n-m]{\frac{B}{a_n}} = 1 + \sqrt[5-3]{\frac{3}{6}} = 1 + \sqrt{\frac{1}{2}} \approx 1 + 0.71 = 2.71 = R_1.$$

Тогда

$$\frac{1}{R_1} = \frac{1}{2.71} \approx 0.74 < x^+ < 2.25 = R.$$

Теорема про верхнюю границу дала лучшее ограничение снизу для положительных корней.

Применим другую замену:

$$P(-y) = 2(-y)^5 + 3(-y)^2 - 2(-y) - 6 = -2y^5 + 3y^2 + 2y - 6 = 0.$$

Умножаем это уравнение на (-1). Получаем $2y^5 - 3y^2 - 2y + 6 = 0$. Вводим величину $B = \max|a_i| = 3$, $m = \max i = 2$, $a_i < 0$. Тогда положительные корни уравнения $P(-y) = 0$ ограничены сверху величиной

$$y^+ < 1 + \sqrt[n-m]{\frac{B}{a_n}} = 1 + \sqrt[5-2]{\frac{3}{2}} = 1 + \sqrt[3]{1.5} \approx 1 + 1.14 = 2.14 = R_2.$$

Сделаем последнюю замену

$$P\left(-\frac{1}{y}\right) = 2\left(\frac{1}{y}\right)^5 - 3\left(\frac{1}{y}\right)^2 - 2 \cdot \frac{1}{y} + 6 = 0.$$

Умножим полученное уравнение на y^5. Получим $2 - 3y^3 - 2y^4 + 6y^5 = 0$. Применим для него теорему про верхнюю границу. Введём величину

$$B = \max|a_i| = 3,\ m = \max i = 3,\ a_i < 0.$$

Тогда положительные корни ограничены сверху величиной

$$y^+ < 1 + \sqrt[n-m]{\frac{B}{a_n}} = 1 + \sqrt[5-3]{\frac{3}{6}} = 1 + \sqrt{0.5} \approx 1 + 0.71 = 1.71 = R_3.$$

Для исходного уравнения отрицательные корни будут ограничены значениями

$$-R_2 = -2.14 < x^- < -\frac{1}{R_3} - \frac{1}{1.71} = -0.58.$$

Теорема про кольцо дала лучшее ограничение снизу отрицательных корней, но худшее ограничение сверху для них. Если объединить 2 результата, получим $-2 < x^- < -0.58$.

Применим теорему Лагранжа. Выписываем полином

$$P(x) = 2x^5 + 3x^2 - 2x - 6.$$

Раскладываем полином на две части $P(x) = F(x) + \Phi(x)$, где

$$F(x) = 2x^5 + 3x^2 - 2x - 6$$

содержит все подряд старшие компоненты с положительными коэффициентами и все отрицательные, $\Phi(x) = 0$ содержит все остальные.

Если существует α такое, что $F(\alpha) \geq 0$, то $x^+ < \alpha$. Подходит значение $\alpha = 1.2$. Проверяем $F(\alpha) = F(1.2) = 2 \cdot (1.2)^5 + 3(1.2)^2 - 2 \cdot 1.2 - 6 \approx 0.9 > 0$.

Получили $\frac{2}{3} < x^+ < 1.2$.

Воспользуемся методом Штурма определения количества корней на задан-

ном промежутке. Построим систему полиномов Штурма по правилу

$$P_0 = P(x),\ P_1 = P'(x),\ P_i = -P_{i-1}\, mod\, P_i,\ i = 1, \ldots, n-1.$$

```
P0 = x**4 + 3*x**3 - 21*x**2 - 43*x + 60
P1 = 4*x**3 + 9*x**2 - 42*x - 43
P2 = 195*x**2/16 + 195*x/8 - 1089/16
P3 = 1408*x/65 + 2432/65
P4 = 35721/484
```

Построим графики полиномов Штурма с помощью WolframAlpha (рис. 2.1).

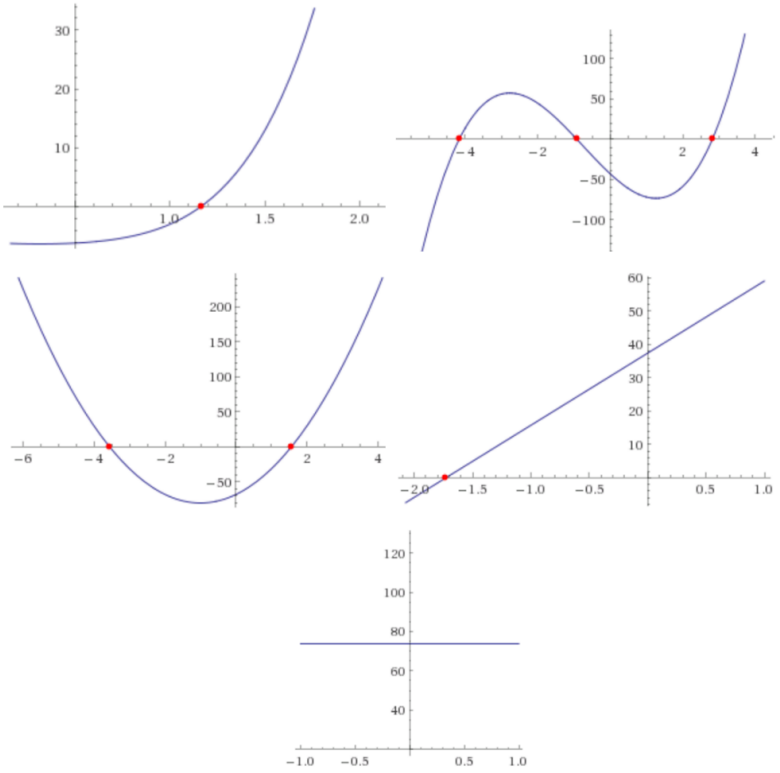

Рисунок 2.1 — Графики полиномов Штурма

На рисунке 2.1 свеху вниз слева направо отображены графики полиномов Штурма (от P_0 до P_4). Красными точками на графиках отмечены действительные корни уравнений $P_i(x) = 0,\ i = 0, \ldots 4$.

Корни каждого следующего полинома размещаются между корнями преды-

дущего полинома и образуют перевёрнутый треугольник (с основанием сверху).

Найдём количество смен знаков полиномов в крайних точках найденных интервалов

	-2	-0.58	$\frac{2}{3}$	1.2
P_0	$+$	$+$	$+$	-
P_1	$+$	-	-	-
P_2	-	-	-	-
P_3	-	$+$	$+$	$+$
P_4	$+$	$+$	$+$	$+$
Количество смен знаков	2	2	2	1

Получили, что количество действительных корней на отрезке

$$[-2, -0.58]$$

равно $2 - 2 = 0$, а на отрезке

$$\left[\frac{2}{3}, 1.2\right]$$

есть $2 - 1 = 1$ корень.

3 ЛИСТИНГ ПРОГРАММЫ

Листинг файла utils.py

```
a = 2/3
b = 1.2
accuracy = 10**(-5)

def f(x):
    return 2 * x**5 + 3 * x**2 - 2*x - 6

def output(iteration, x, accuracy):
    print 'Iteration # {}'.format(iteration)
    print 'Approximate value {}'.format(x)
    print 'Error: {}'.format(accuracy)
```

Листинг программы уточнения корней по методу бисекции

```
from utils import *

def bisection(a, b, accuracy):
    c = (a + b) / 2
    iteration = 1
    while abs(a - b) >= accuracy:
        output(iteration, c, abs(a - b))
        if f(c) == 0:
            return c
        elif f(a) * f(c) < 0:
            b = c
        elif f(b) * f(c) < 0:
            a = c
        c = (a + b) / 2
        iteration += 1
    return c

bisection(a, b, accuracy)
```

Листинг программы уточнения корней по методу хорд

```
from utils import *

def horde(a, b, accuracy):
    c = (a * f(b) - b * f(a)) / (f(b) - f(a))
    iteration = 1
    while True:
        output(iteration, c, abs(f(c)))
        if f(c) == 0:
            return c
        elif f(a) * f(c) < 0:
```

```
            b = c
        elif f(b) * f(c) < 0:
            a = c
        c_prev, c = c, (a * f(b) - b * f(a)) / (f(b) - f(a))
        if abs(c_prev - c) < accuracy or abs(f(c)) < accuracy:
            break
        iteration += 1
    return c

horde(a, b, accuracy)
```

Листинг программы уточнения корней по методу Ньютона (касательных)

```
from utils import *

def derivativeF(x):
    return 10 * x**4 + 6 * x - 2

def derivative2F(x):
    return 40 * x**3 + 6

def newton(x0, accuracy):
    iteration = 1
    while True:
        output(iteration, x0, abs(f(x0)))
        x0_prev, x0 = x0, x0 - f(x0) / derivativeF(x0)
        if abs(x0_prev - x0) < accuracy or abs(f(x0)) < accuracy:
            break
        iteration += 1
    return x0

if f(a) * derivative2F(a) > 0:
    newton(a, accuracy)
elif f(b) * derivative2F(b) > 0:
    newton(b, accuracy)
```

Листинг программы вычисления полиномов Штурма

```
from sympy import *


x, y, z, t = symbols('x y z t')
p = []

p.append(x**4 + 3*x**3 - 21*x**2 - 43*x + 60)
p.append(4*x**3 + 9*x**2 - 42*x - 43)

for i in range(3):
    q, r = div(p[i], p[i + 1])
    p.append(-r)
```

```
for i, P in enumerate(p):
    print('P{} = {}'.format(i, P))
```

4 РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ

Результаты метода бисекции

```
Iteration # 1
Approximate value 0.6
Error: 1.2
Iteration # 2
Approximate value 0.9
Error: 0.6
Iteration # 3
Approximate value 1.05
Error: 0.3
Iteration # 4
Approximate value 1.125
Error: 0.15
Iteration # 5
Approximate value 1.1625
Error: 0.075
Iteration # 6
Approximate value 1.18125
Error: 0.0375
Iteration # 7
Approximate value 1.171875
Error: 0.01875
Iteration # 8
Approximate value 1.1671875
Error: 0.009375
Iteration # 9
Approximate value 1.16484375
Error: 0.0046875
Iteration # 10
Approximate value 1.163671875
Error: 0.00234375
Iteration # 11
Approximate value 1.1630859375
Error: 0.001171875
Iteration # 12
Approximate value 1.16337890625
Error: 0.0005859375
Iteration # 13
Approximate value 1.16352539063
Error: 0.00029296875
Iteration # 14
Approximate value 1.16359863281
Error: 0.000146484375
Iteration # 15
Approximate value 1.16356201172
```

```
Error: 7.32421874998e-05
Iteration # 16
Approximate value 1.16354370117
Error: 3.66210937499e-05
Iteration # 17
Approximate value 1.16355285645
Error: 1.8310546875e-05
```

Результаты метода хорд с критерием завершения процесса $|f(c)| < \varepsilon$

```
Iteration # 1
Approximate value 1.04398663697
Error: 2.33794623207
Iteration # 2
Approximate value 1.15675248338
Error: 0.157065227139
Iteration # 3
Approximate value 1.16319895517
Error: 0.00837728079985
Iteration # 4
Approximate value 1.16353960357
Error: 0.000440891179815
Iteration # 5
Approximate value 1.16355752287
Error: 2.31874705268e-05
```

Результаты метода хорд с критерием завершения процесса

$$|c - c_{previous}| < \varepsilon$$

```
Iteration # 1
Approximate value 1.04398663697
Error: 2.33794623207
Iteration # 2
Approximate value 1.15675248338
Error: 0.157065227139
Iteration # 3
Approximate value 1.16319895517
Error: 0.00837728079985
Iteration # 4
Approximate value 1.16353960357
Error: 0.000440891179815
Iteration # 5
Approximate value 1.16355752287
```

```
Error: 2.31874705268e-05
```

Результаты метода Ньютона (касательных) с критерием завершения процесса $|f(x_0)| < \varepsilon$

```
Iteration # 1
Approximate value 1.2
Error: 0.89664
Iteration # 2
Approximate value 1.16542874769
Error: 0.043717708526
Iteration # 3
Approximate value 1.16356368171
Error: 0.000120381699087
```

Результаты метода Ньютона (касательных) с критерием завершения процесса $\left|x_{0_{previous}} - x_0\right| < \varepsilon$

```
Iteration # 1
Approximate value 1.2
Error: 0.89664
Iteration # 2
Approximate value 1.16542874769
Error: 0.043717708526
Iteration # 3
Approximate value 1.16356368171
Error: 0.000120381699087
```

ВЫВОДЫ

С помощью метода бисекции корень заданного уравнения был получен на 17-й итерации, с помощью метода хорд — на пятой, а с помощью метода Ньютона (касательных) —на третьей.

Для метода бисекции было обнаружено, что при увеличении и уменьшении длины отрезка количество итераций практически не меняется. Так, для отрезка $[1.1, 1.2]$ результат был получен на 14-й итерации.

Метод хорд даёт результат быстрее. При уменьшении длины отрезка количество итераций остаётся примерное такой же. Так, для отрезка $[1.1, 1.2]$ результат был получен на четвёртой итерации. Но при увеличении, например, — для отрезка $[0.5, 2]$ результат был получен на 31-й итерации, то есть количество итераций увеличивается.

Преимуществом метода Ньютона (касательных) является его быстрая сходимость. На него практически не влияет незначительное изменение длины отрезка. Для данного метода необходимо знать начальное приближение, а не границы интервала, в котором находится корень.

Недостатком методов хорд и Ньютона является то, что на заданном отрезке требуется монотонность функции и условие на вторую производную (она не должна изменять знак). В свою очередь, для метода Ньютона нужно выбрать начальное приближение так, чтобы касательная, проведённая в этой точке, не пересекла ось абсцисс вне заданного отрезка.