**МФТИ**

Лабораторная работа №1

“Численное решение нелинейного уравнения”

Выполнил
Студент Б03-907
Алиев Артем Эльдарович

Долгопрудный, 2021

**Задача**

Определить корни уравнений (локализовать), а затем уточнить с помощью методов:

- половинчатое деление
- МПИ
- метод Ньютона
- модифицированный метод Ньютона
- метод секущих

с точностью: $\varepsilon = 10^{-4}$

## Локализация

Требуется найти непересекающиеся отрезки $[a_i, b_i]$, каждому из которых принадлежит один и только один корень данного уравнения $x_i^*$.

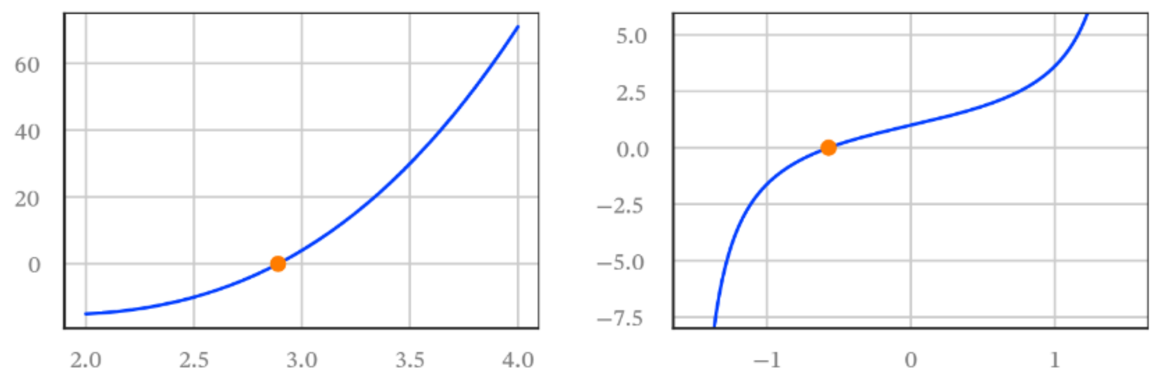
Пусть $f(x) = a_0 x^n + a_1 x^{n-1} + ... + a_{n-1} x + a_n = 0$, тогда требуется

1. Определить число этих корней $N = N_+ + N_-$;
   - Т. Декарта: число положительных корней равно числу перемен знаков в последовательност коэффициентов $a_0, a_1, ..., a_n$ или на четное число меньше.
   - Все корни многочлена (включая комплексные) лежат в кольце

   $$\frac{|a_n|}{|a_n| + B} <= |z| <= 1 + \frac{A}{|a_0|}$$

   где $A = max\{|a_1|, |a_2|, ..., |a_n|\}$, $B = max\{|a_0|, |a_1|, ..., |a_{n-1}|\}$
2. Определить отрезки, на которых лежат все действительные корни;
3. Найти $N$ непересекающихся отрезков $[a_i, b_i]$, для которых $f(a_i)f(b_i) < 0$.

## Графики функций и выбор отрезков




(a) $3x + 4x^3 - 12x^2 - 5 = 0$

(b) $2\tan(x) - \frac{x}{2} + 1 = 0$

Первая функция возрастает во всей области определения. Корень находится в сегменте $[2.5, 3.5]$. Вторая функция периодическая. Один из корней находится в сегменте $[-1, 0]$.

## Методы уточнения корней

**Уточнение корней** Для каждого корня $x_i^* \in [a_i, b_i]$ данного уравнения требуется найти $\tilde{x}_i$ такое, что $|\tilde{x}_i - x_i^*| < \varepsilon$, где $\varepsilon$ - заданная погрешность.

### Метод половинного деления

Отыскивается $\tilde{x}$ - приближение к корню $x^* \in [a, b]$ с точностью $\varepsilon$.

Шаг 1. $m = 0$

Шаг 2. $a_m = a, b_m = b$

Шаг 3. $c = \frac{a_m + b_m}{2}$

Шаг 4. (а) Если $f(c)f(a_m) > 0$, то $a_{m+1} = c, b_{m+1} = b_m$ (б) Если $f(c)f(b_m) > 0$, то $a_{m+1} = a_m, b_{m+1} = c$

Шаг 5. Если $|b_{m+1} - a_{m+1}| > \varepsilon$, то $m = m + 1$, перейти к Шагу 2, иначе $\tilde{x} = \frac{a_{m+1} + b_{m+1}}{2}$

### Метод простой итерации

$$f(x) = 0 \rightarrow x = g(x) \rightarrow x_{n+1} = g(x_n)$$

Условие сходимости: $\forall x \in [a, b] : |g'(x)| < 1$ или $\forall x', x'' \in [a, b] : |g(x') - g(x'')| <= q|x' - x''|, q < 1$

### Метод Ньютона

$$f(x) = f(x_n) + f'(x_n)(x - x_n) \rightarrow x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Условие сходимости: $\frac{1}{2}\frac{M_2}{m_1}|x_0 - x^*|^2 < 1$

$$|x_{n+1} - x^*| < \frac{1}{2}\frac{M_2}{m_1}|x_n - x^*|^2 < C^{-1}(C|x_0 - x^*|)^{2^n}$$

Выбор начального приближения: $f(x_0)f''(x_0) > 0$. Практический критерий оценки достижения заданной точности:

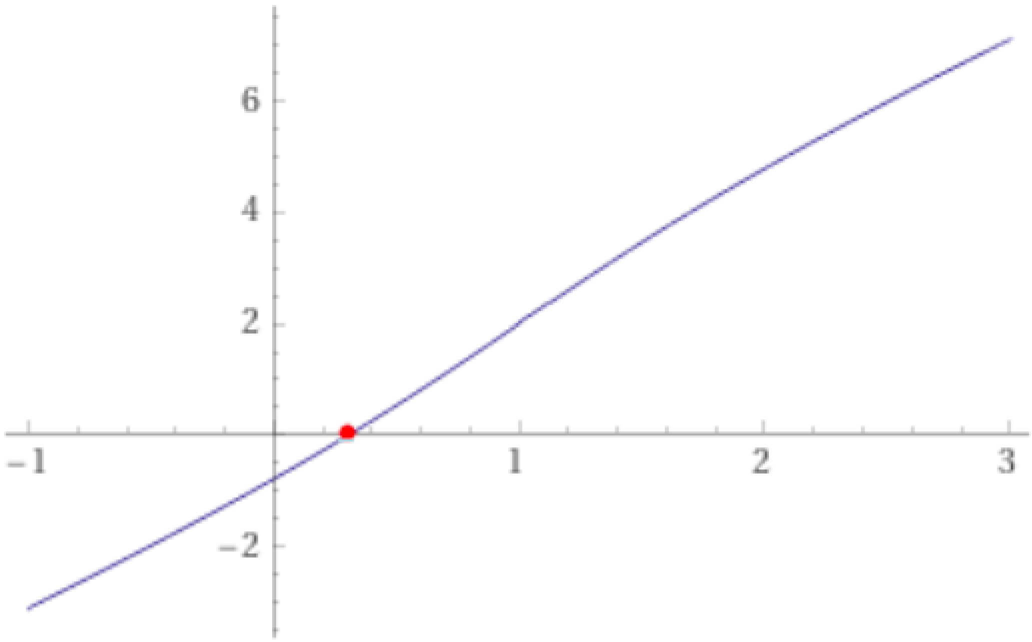
$$|x_{n+1} - x^*| < \frac{1}{2}\frac{M_2}{m_1}|x_{n+1} - x_n|^2 < \varepsilon$$

### Метод Секущих

В методе Ньютона подставим: $f'(x_n) \approx \dfrac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}$$

I. arctg(x-1) + 2x = 0




$$2x = \tan^{-1}(1-x)$$

$$2x - \frac{1}{2}\, i \log(1 - i(1-x)) + \frac{1}{2}\, i \log(1 + i(1-x)) = 0$$

## Number line


## Solution

$x \approx 0.304014$

Из графика видно, что отрезок [0; 1] можно назвать отрезком локализации.

1) Решим с помощью бисекции:

Код:

```
import math
from math import atan

a = 0; b = 1; e = 0.0001; i = 1
def f(x):
    return atan(x-1) + 2*x
y1 = f(a); y2 = f(b)
if (y1 * y2 > 0) or (y1 * y2 == 0):
    print("no solutions")
else:
    n = 1
    x = (a+b)/2
    y3 = f(x)
    print('  ', ' a', '      b', '      (a+b)/2', 'f((a+b)/2)', 'f(a)', '    f(b)')
    while (abs(y3) > e):
        if i > 9:
            print(i, end = '. ')
        else:
            print(i, end = '.  ')
        print('%.5f' % round(a, 5),  '%.5f' % round(b, 5), end = ' ')
        x = (a+b)/2
        print('%.5f' % round(x, 5), end = ' ')
        if f(x) < 0:
            print('%.5f' % round(f(x), 5), end = '   ')
        else:
            print('%.5f' % round(f(x), 5), end = '    ')
        print('%.5f' % round(f(a), 5), '%.5f' % round(f(b), 5))
        i += 1
        y3 = f(x);
        if y1 * y3 < 0:
            b = x
        else:
            a = x
            n += 1
    print('   x = %.5f' % round(x, 5))
    print('f(x) = %.5f' % round(f(x), 5))
```

Данные:

| | a | b | (a+b)/2 | f((a+b)/2) | f(a) | f(b) |
|---|---|---|---|---|---|---|
| 1. | 0.00000 | 1.00000 | 0.50000 | 0.53635 | −0.78540 | 2.00000 |
| 2. | 0.00000 | 0.50000 | 0.25000 | −0.14350 | −0.78540 | 0.53635 |
| 3. | 0.25000 | 0.50000 | 0.37500 | 0.19140 | −0.14350 | 0.53635 |
| 4. | 0.25000 | 0.37500 | 0.31250 | 0.02271 | −0.14350 | 0.19140 |
| 5. | 0.25000 | 0.31250 | 0.28125 | −0.06070 | −0.14350 | 0.02271 |
| 6. | 0.28125 | 0.31250 | 0.29688 | −0.01907 | −0.06070 | 0.02271 |
| 7. | 0.29688 | 0.31250 | 0.30469 | 0.00180 | −0.01907 | 0.02271 |
| 8. | 0.29688 | 0.30469 | 0.30078 | −0.00864 | −0.01907 | 0.00180 |
| 9. | 0.30078 | 0.30469 | 0.30273 | −0.00342 | −0.00864 | 0.00180 |
| 10. | 0.30273 | 0.30469 | 0.30371 | −0.00081 | −0.00342 | 0.00180 |
| 11. | 0.30371 | 0.30469 | 0.30420 | 0.00050 | −0.00081 | 0.00180 |
| 12. | 0.30371 | 0.30420 | 0.30396 | −0.00016 | −0.00081 | 0.00050 |
| 13. | 0.30396 | 0.30420 | 0.30408 | 0.00017 | −0.00016 | 0.00050 |
| 14. | 0.30396 | 0.30408 | 0.30402 | 0.00001 | −0.00016 | 0.00017 |

```
   x = 0.30402
f(x) = 0.00001
```

## 2) Решим методом простых итераций:

Код:

```
import math
from math import atan

e = 0.0001; i = 1
def f(x):
    return atan(x-1) + 2*x

def phi(x):
    return -atan(x-1)/2

x0 = 0.5
x1 = phi(x0)
X1 = []
X0 = []
print('  x0', '     f(x0)')
while abs(x0 - x1) > e:
    X1.append(x1)
    X0.append(x0)
    x0 = x1
    x1 = phi(x1)
    print(i, '%.5f' % round(x0, 5), '%.5f' % round(f(x0), 5))
    i += 1

print('   x0 =', '%.5f' % round(x0, 5))
print('f(x0) =', '%.5f' % round(f(x0), 5))
```

Данные:

```
   x0       f(x0)
1 0.23182 -0.19139
2 0.32752 0.06301
3 0.29601 -0.02138
4 0.30670 0.00718
5 0.30311 -0.00242
6 0.30432 0.00082
7 0.30391 -0.00027
8 0.30405 0.00009
   x0 = 0.30405
f(x0) = 0.00009
```

## 3) Решим методом Ньютона:

Код:

```
import math
from math import atan
from sympy import *

e = 0.0001; i = 1

x = Symbol('x')
f = atan(x-1) + 2*x

df = f.diff(x)

x0 = 0.5
x1 = x0 - f/df
X0 = []
F = []
dF = []

print('  x0', '     f(x0)')

while abs(x1.subs(x, x0) - x0) > e:
    X0.append(x0)
    F.append(f.subs(x, x0))
    dF.append(df.subs(x, x0))
    x0 = x1.subs(x, x0)
    x1 = x0 - f / df
    print(i, '%.5f' % round(x0, 5), '%.5f' % round(f.subs(x, x0), 5))
    i += 1

print('   x0 =', '%.5f' % round(x0, 5))
print('f(x0) =', '%.6f' % round(f.subs(x, x0), 6))
```

Данные:

```
  x0       f(x0)
1 0.30845 0.01186
2 0.30402 0.00001
   x0 = 0.30402
f(x0) = 0.000006
```

4) А также модифицированным методом Ньютона:

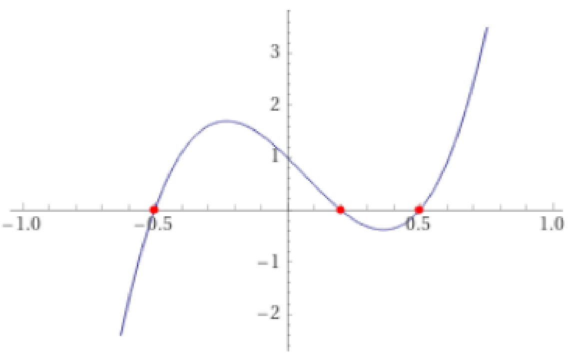
Код:

```
import math
from math import atan
from sympy import *

e = 0.0001; i = 1

x = Symbol('x')
f = atan(x-1) + 2*x

df = f.diff(x)

x0 = 0.5
x1 = x0 - f/df
X0 = []
F = []
dF = []

print('  x0', '     f(x0)')

x_tmp = x0
flag = True
while abs(x1.subs(x, x0) - x0) > e:
    if abs(x1.subs(x, x0) - x0) < e + 0.006:
        flag = False
    X0.append(x0)
    F.append(f.subs(x, x0))
    dF.append(df.subs(x, x0))
    print(i, '%.5f' % round(x0, 5), '%.5f' % round(f.subs(x, x0), 5))
    x0 = x1.subs(x, x0)
    i += 1
    if flag:
        x1 = x0 - f / df
    else:
        x1 = x0 - f / df.subs(x, x_tmp)

print('   x0 =', '%.5f' % round(x0, 5))
print('f(x0) =', '%.6f' % round(f.subs(x, x0), 6))
```

Данные:

```
  x0       f(x0)
1 0.50000 0.53635
2 0.30845 0.01186
   x0 = 0.30402
f(x0) = 0.000006
```

## II. 20x**3 - 4x**2 -5x + 1 = 0




Alternate forms

$20\,x^3 + 1 = x\,(4\,x + 5)$

$(2\,x - 1)\,(2\,x + 1)\,(5\,x - 1) = 0$

$20\left(x - \frac{1}{15}\right)^3 - \frac{79}{15}\left(x - \frac{1}{15}\right) + \frac{442}{675} = 0$

Number line


Solutions

$x = -\frac{1}{2}$

$x = \frac{1}{5}$

$x = \frac{1}{2}$

Из графика видно, что отрезки [-1; 0], [0; 0,4], [0,4; 1] можно назвать отрезками локализации.

1) Решим с помощью бисекции:

Код:

Аналогичен коду прошлой функции, только теперь вводим другую **f(x)**, и соответственно новые **a** и **b** (концы отрезков локализации)

Данные:

```
   x = -0.50000
f(x) = 0.00000
```

```
   x = 0.20000
f(x) = 0.00000
```

```
    a       b       (a+b)/2 f((a+b)/2) f(a)     f(b)
1.  0.40000 1.00000 0.70000 2.40000    -0.36000 12.00000
2.  0.40000 0.70000 0.55000 0.36750    -0.36000 2.40000
3.  0.40000 0.55000 0.47500 -0.13406   -0.36000 0.36750
4.  0.47500 0.55000 0.51250 0.07910    -0.13406 0.36750
5.  0.47500 0.51250 0.49375 -0.03649   -0.13406 0.07910
6.  0.49375 0.51250 0.50313 0.01900    -0.03649 0.07910
7.  0.49375 0.50313 0.49844 -0.00931   -0.03649 0.01900
8.  0.49844 0.50313 0.50078 0.00470    -0.00931 0.01900
9.  0.49844 0.50078 0.49961 -0.00234   -0.00931 0.00470
10. 0.49961 0.50078 0.50020 0.00117    -0.00234 0.00470
11. 0.49961 0.50020 0.49990 -0.00059   -0.00234 0.00117
12. 0.49990 0.50020 0.50005 0.00029    -0.00059 0.00117
13. 0.49990 0.50005 0.49998 -0.00015   -0.00059 0.00029
14. 0.49998 0.50005 0.50001 0.00007    -0.00015 0.00029
   x = 0.50001
f(x) = 0.00007
```

2) Методом простых итераций:

НЕ УДАЛОСЬ ПОДОБРАТЬ ОБРАТНУЮ ФУНКЦИЮ

3) Метод Ньютона:

Код:

Аналогичен коду прошлой функции, только теперь вводим другую **f(x)** и разные x0 (в зависимости от отрезков локализации)

Данные:

```
  x0      f(x0)
1 -0.53846 -0.58990
2 -0.50315 -0.04440
3 -0.50002 -0.00033
   x0 = -0.50002
f(x0) = -0.000333
```

```
  x0      f(x0)
1 0.20000 0.00000
   x0 = 0.20000
f(x0) = 0.000000
```

```
  x0      f(x0)
1 0.65714 1.66251
2 0.55093 0.37569
3 0.50826 0.05135
4 0.50028 0.00168
5 0.50000 0.00000
   x0 = 0.50000
f(x0) = 0.000002
```

4) Модифицированный метод Ньютона:

Код:

Аналогичен коду прошлой функции, только теперь вводим другую **f(x)** и разные x0 (в зависимости от отрезков локализации)

Данные:

```
  x0       f(x0)
1 -0.40000 1.08000
2 -0.53846 -0.58990
3 -0.50315 -0.04440
   x0 = -0.50002
f(x0) = -0.000333
```

```
  x0       f(x0)
1 0.00000 1.00000
   x0 = 0.20000
f(x0) = 0.000000
```

```
  x0       f(x0)
1 0.40000 -0.36000
2 0.65714 1.66251
3 0.55093 0.37569
4 0.50826 0.05135
5 0.50028 0.00168
   x0 = 0.50000
f(x0) = 0.000002
```

*Ссылка на исходные коды на GitHub*