Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования Московский государственный технический университет имени Н.Э. Баумана

Лабораторная работа №4
«Сравнение методов решения уравнения $f(x) = 0$»
по дисциплине
«Численные методы»

Студент группы ИУ9-62 Белогуров А.А.

Преподаватель Домрачева А.Б.

Москва, 2017

Цель: Сравнительный анализ методов решения уравнения $f(x)$:

1. Метод деления отрезка пополам.
2. Метод золотого сечения.

Постановка задачи:

Дано нелинейное уравнение $f(x) = 0$ третьей степени, которое выбирается произвольно. В моем случае было выбрано следующее ур-е (1):

$$f(x) = 5x^3 - 8x^2 - 8x + 5 = 0 \tag{1}$$

С помощью выбранных методов получить приближенное решение ур-я (1) с заданной точностью ε и сравнить полученные результаты.

Функция $f(x)$ должна быть непрерывна.

Теоретические сведения:

1. Метод деления отрезка пополам

Метод деления отрезка пополам (метод бисекции) - один из методов поиска корней нелинейного уравнения, которое основывается на разбиении отрезка пополам на каждом шаге алгоритма.

Описание алгоритма: Для начала вычислений необходимо знать значения отрезка (2), на концах которого функция принимает значения разных знаков (следствие из теоремы Больцано-Коши)

$$[x_L, x_R] \tag{2}$$

Следствие из теоремы Больцано-Коши:

Пусть непрерывная функция $f(x) \in C([a,b])$, тогда если $sign(f(a)) \neq sign(f(b))$, то $\exists \in [a,b] : f(c) = 0$.

Для того, чтобы определить, принимает ли функция разные значения на концах отрезка - можно использовать два способа: через умножение (3) и через обычное сравнение (4):

$$f(x_L) \times f(x_R) < 0 \tag{3}$$

$$sign(f(x_L)) \neq sign(f(x_R)) \tag{4}$$

При использовании формулы (3) может возникнуть преждевременное переполнение, поэтому для устранения этой проблемы и для увеличения быстродействия будет использоваться формула (4).

Из непрерывности функции $f(x)$ и условия (4) следует, что на отрезке (2) существует хотя бы один корень уравнения. В нашем случае уравнение третьей степени, поэтому мы должны выделить «вилки» - отрезки, включающие точку, в котором график пересекает ось oX (решение). Таким образом надо будет выделить три таких отрезка.

Далее вычисляется значение x_M (5) в середине отрезка:

$$\frac{x_L + x_R}{2} \tag{5}$$

- Если $|b-a| \leq 2 \times \varepsilon$, то корень $x = \frac{b-a}{2}$ найден
- Иначе разобьем отрезок (2) на два равных отрезка (6), (7):

$$[x_L, x_M] \tag{6}$$

$$[x_M, x_R] \tag{7}$$

Определим отрезок, на котором функция меняет свой знак:

- Если условие (4) выполняется на левом отрезке (6), то, соответственно, корень находится на левом отрезке. Производим замену $x_R = x_M$ и возвращаемся к шагу разбиения отрезка пополам (5)
- Если условие (4) выполняется на правом отрезке (7), то, соответственно, корень находится на правом отрезке. Производим замену $x_L = x_M$ и возвращаемся к шагу разбиения отрезка пополам (5)

2. Метод золотого сечения

В основе метода лежит принцип деления отрезка пополам в пропорциях золотого сечения.

Аналогично предыдущему методу выбирается отрезок (8), внутри которого функция $f(x)$ принимает нулевые значения:

$$[a, b] \tag{8}$$

Необходимо выбрать две точки x_1 и x_2 такие, что:

$$\frac{b-a}{b-x_1} = \frac{b-a}{x_2-a} = \varphi = \frac{1+\sqrt{5}}{2} \tag{9}$$

Где φ (9) - пропорция золотого сечения. Таким образом точки x_1 (10) и x_2 (11) определяются следующим образом:

$$x_1 = b - \frac{(b-a)}{\varphi} \tag{10}$$

$$x_2 = a + \frac{(b-a)}{\varphi} \tag{11}$$

Описание алгоритма: Алгоритм почти идентичен алгоритму метода деления отрезка пополам. Отличие заключается в том, что отрезок делится не пополам, а на три части в пропорциях золотого сечения.

Таким образом, в начале проверяется:

- Если $|b - a| \leq 2 \times \varepsilon$, то корень $x = \frac{b-a}{2}$ найден
- Иначе находятся значения точек x_1 (10) и x_2 (11)

Определим отрезок, на котором функция меняет свой знак:

- Если условие (4) выполняется на левом отрезке $[a, x_1]$, то, соответственно, корень находится на левом отрезке. Производим замену $b = x_1$ и возвращаемся к предыдущему шагу
- Если условие (4) выполняется на отрезке посередине $[x_1, x_2]$, то, соответственно, корень находится на отрезке посередине. Производим замену $a = x_1$, $b = x_2$ и возвращаемся к предыдущему шагу
- Если условие (4) выполняется на правом отрезке $[x_2, b]$, то, соответственно, корень находится на правом отрезке. Производим замену $a = x_2$ и возвращаемся к предыдущему шагу

Практическая реализация:

Листинг 1: Методы решения уравнения f(x)

```
import numpy
import math

phi = (1 + math.sqrt(5)) / 2


def bisection(a, b, eps, f):
  if f(a) == 0:
    return a
  if f(b) == 0:
    return b

  iteration = 0
  while (b - a) > 2 * eps:
    iteration += 1
    dx = (b - a)/2
```

```
    x = a + dx

    if numpy.sign(f(a)) != numpy.sign(f(x)):
      b = x
    else:
      a = x

  return [(a + b)/2, iteration]


def golden_section(a, b, eps, f):
  iteration = 0

  while (b - a) > 2 * eps:
    iteration += 1
    x_1 = b - (b - a) / phi
    x_2 = a + (b - a) / phi

    if numpy.sign(f(a)) != numpy.sign(f(x_1)):
      b = x_1
      continue

    if numpy.sign(f(x_1)) != numpy.sign(f(x_2)):
      a = x_1
      b = x_2
      continue

    if numpy.sign(f(x_2)) != numpy.sign(f(b)):
      a = x_2
      continue

  return [(a + b) / 2, iteration]


if __name__ == "__main__":
  f_x = numpy.poly1d([5, -8, -8, 5])
  eps = 0.001

  print("Bisection:")
  print(bisection(-1.5, 0, eps, f_x))
  print(bisection(0, 1.5, eps, f_x))
  print(bisection(1.5, 3, eps, f_x))
```

```
    print("\nGolden section:")
    print(golden_section(-1.5, 0, eps, f_x))
    print(golden_section(0, 1.5, eps, f_x))
    print(golden_section(1.5, 3, eps, f_x))
```

Результаты:

Для тестирования было выбрано уравнение (1), $\varepsilon = 0.001$.

Уравнение (1) имеет три корня, для корректного результата выберем три одинаковых «ветки»:

[-1.5, 0]	[0, 1.5]	[1.5, 3]

Ниже приведена таблица результата тестовой программы (Листинг 1) на указанных выше методах:

Отрезок	Значение	Количество итераций
Метод деления отрезка пополам		
[-1.5, 0]	-0.999756	10
[0, 1.5]	0.469482	10
[1.5, 3]	2.130615	10
Метод золотого сечения		
[-1.5, 0]	-0.999337	7
[0, 1.5]	0.468734	6
[1.5, 3]	2.130709	6
Значения, полученные с помощью сервиса WolframAlpha		
[-1.5, 0]	-1	-
[0, 1.5]	0.469338	-
[1.5, 3]	2.130662	-

Выводы:

В ходе выполнения лабораторной работы были рассмотрены 2 различных метода нахождения корня нелинейного уравнения: метод деления отрезка пополам и метод золотого сечения. Была написана реализация данных методов на языке программирования Python.

Как видно выше в таблице вычислений, по количеству итераций выигрывает метод золотого сечения, так как он в своей реализации делит отрезок на 3 части и, следовательно, приходит быстрее к результату, однако если сравнивать оба метода по точности вычислений, то ближе к правильным ответам оказался именно метод деления отрезка пополам. Стоит заметить, что при округлении значений с заданной точностью ε оба методы будут верны, поэтому в данном случае не будет корректным сравнивать эти методы по точности вычислений. Одна из причин, почему метод деления оказался ближе к результату, чем метод золотого сечения, заключается в различном способе нахождения конечного результата. Однако, если целью сравнения будет именно поиск более точного метода, то стоит рассматривать не одно уравнение, а большое их количество.