ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

tpu.ru

Углубленный курс информатики

Лекция 8

Численные методы решения нелинейных уравнений

28 марта 2022 г.

Игорь Михайлович Долганов,
к.т.н., доцент ОХИ ИШПР

ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Содержание

ТОМСКИЙ
ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ

Численные методы решения нелинейных уравнений

Постановка задачи

- Задача определения корней нелинейных уравнений встречается при расчетах тепловых балансов химико-технологических процессов, процессов однократного испарения, процессов отстаивания, при расчете точки росы, диаметров нефтепроводов и многих других задачах.
- Также задача нахождения корней нелинейных уравнений встречается при определении физико-химических свойств исследуемых систем, скажем, при вычислении констант фазового равновесия углеводородов при расчете процесса ректификации и других химико-технологических процессах, протекающих с участием нескольких фаз.

Нелинейное уравнение в общем виде можно представить следующим образом:

$$f(x) = 0 \tag{1}$$

Нелинейные уравнения подразделяются на два вида: алгебраические и трансцендентные.

- *Алгебраические* уравнения содержат только алгебраические функции (рациональные, иррациональные, целые). Данный тип нелинейных уравнений можно представить в виде полинома n-ой степени с действительными коэффициентами:

$$f(x) = a_0x^n + a_1x^{n-1} + \ldots + a_k = 0 \tag{2}$$

В качестве примера алгебраического уравнения можно привести следующее выражение:

$$2x^3 + 3x^2 + 3 = 0$$

Постановка задачи

- *Трансцендентные* уравнения – это уравнения, содержащие показательные, тригонометрические логарифмические и другие функции. К примеру:

$$3x + \cos^2 x = 0$$

- Задача определения корней уравнения (1) состоит в нахождении таких значений x_0, при которых уравнение (1) обращается в тождество:

$$f(x_0) = 0$$

  где x_0 – корень уравнения.
- Методы решения нелинейных уравнений подразделяются на прямые и итерационные.
- Прямые методы представляют собой аналитическое определение корней уравнений в виде математического выражения (формулы). Однако данные методы во многих практических случаях являются трудно применимыми из-за высокой степени сложности решаемых задач.
- Для решения подобного рода задач используются итерационные методы, т.е. методы, основанные на последовательных приближениях.

Приближенное определение корней состоит из двух этапов.

1. Отделение корней – определение малых отрезков, в каждом из которых содержится только один корень уравнения.
2. Уточнение приближенных значений корней до некоторой заранее заданной степени точности.

Графический способ отделения корней

- Предположим, что требуется отделить корни уравнения (1). Построим график функции $f(x) = 0$. Абсциссы точек пересечения графика функции с осью X будут являться приближенными значениями корней уравнения (1).
- Во многих случаях уравнение (1) представляют в более простом виде:

$$\begin{gathered} f(x) = \varphi_1(x) - \varphi_2(x) = 0 \\ \varphi_1(x) = \varphi_2(x) \end{gathered} \tag{3}$$

- Далее необходимо построить графики функций: $y_1 = \varphi_1(x)$ и $y_2 = \varphi_2(x)$. Корнями уравнения (3) будут являться абсциссы пересечения этих кривых.

Отделение корней

Графический способ отделения корней

Пусть требуется определить корни уравнения $f(x) = x \log x - 1 = 0$. Преобразуем данное уравнение к следующему виду:

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

- Далее построим графики функций $y_1 = \log x$ и $y_2 = 1/x$ (рисунок 1).
- Точка пересечения графиков функций y_1 и y_2 дает приближенное значение единственного корня $x_0 \approx 2.5$.

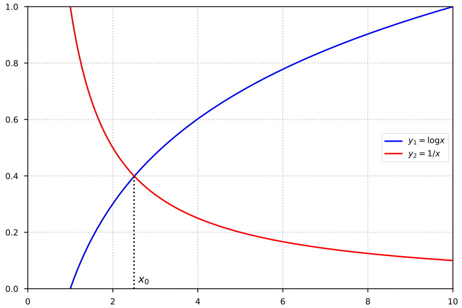


Рисунок 1 – Графический метод отделения корней

Графический способ отделения корней

Аналитический метод отделения корней

- В том случае, когда непрерывная функция $f(x)$ принимает значения разных знаков на концах отрезка $[a, b]$, иначе говоря, $f(a) \cdot f(b) < 0$, то между точками a и b существует хотя бы один действительный корень уравнения $f(x) = 0$, т.е. существует такое число $x_0 \in [a, b]$, что $f(x_0) = 0$ (рисунок 2).
- В том случае, когда на заданном отрезке $[a, b]$ определена первая производная $f'(x)$, при этом она сохраняет свой знак внутри интервала $[a, b]$ ($f'(x) > 0$ или $f'(x) < 0$), то корень x_0 будет единственным.

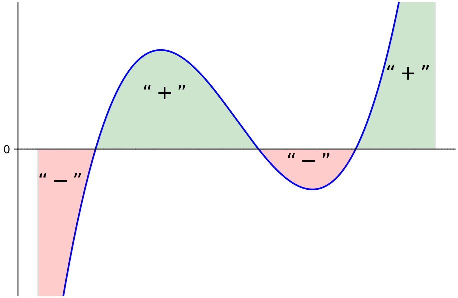


Рисунок 2 – Аналитический метод отделения корней

При аналитическом методе отделения корней, первым этапом является определение знаков функции на концах заданного отрезка a и b. Затем выделяют отрезки, на которых функция меняет знак. Выделенные отрезки содержат корень уравнения.

ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Метод деления отрезка пополам (метод бисекций)

Метод деления отрезка пополам (метод бисекций)

- Пусть дано уравнение $f(x) = 0$.
- Предположим, что известен интервал $[a, b]$, на котором расположен корень x^*, иначе говоря, $a < x^* < b$.
- В качестве начального приближения корня x_0 принимают середину отрезка $[a, b]$: $x_0 = (a + b)/2$ (рисунок 3).
- Далее необходимо исследовать значение функции: если $f(x_0) = 0$, то x_0 – искомый корень уравнения.
- Если $f(x_0) \neq 0$, то необходимо выбрать одну из половин отрезка $[a, x_0]$ или $[x_0, b]$, так, чтобы значение функции меняло знаки на ее концах.

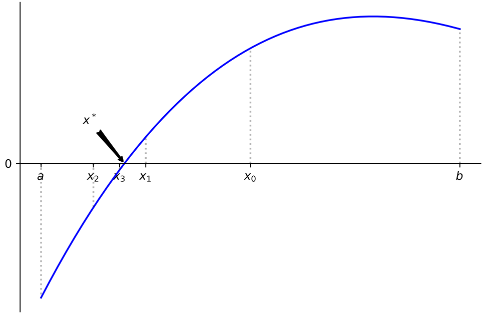
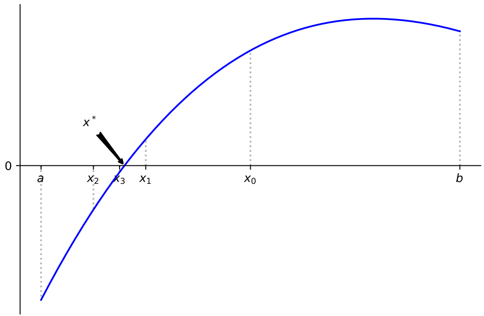


Рисунок 3 – Метод деления отрезка пополам

Метод деления отрезка пополам (метод бисекций)

- Вторую половину, на которой знак функции не меняется на противоположный отбрасывают.
- Отрезок $[a, x_0]$ делим пополам и получаем точку $x_1 = (a + x_0)/2$.
- Снова исследуем функцию $f(x)$ на концах отрезка и отбрасываем отрезок $[x_1, x_0]$.
- Находим новое приближение $x_2 = (a + x_1)/2$.
- Отбрасываем интервал $[a, x_2]$, т.к. $f(a) < 0$ и $f(x_2) < 0$.
- Снова разделим отрезок $[x_2, x_1]$ пополам и получим новое приближение $x_3 = (x_2 + x_1)/2$ и т.д.



Метод деления отрезка пополам (метод бисекций)

- Итерационный процесс будет продолжаться до тех пор, пока после n-ой итерации длина отрезка не станет меньше некоторой заранее определенной величины ε:

$$|b - a| \leqslant \varepsilon \tag{4}$$

- В этом случае искомое значение корня принимается равным полученному приближению x_n: $x^* = x_n$, т.е. решение уравнения будет найдено с точностью ε.

Пример

Рассмотрим решение следующего уравнения:

$$e^x - 6x - 3 = 0$$

на отрезке $[-3, 1]$ с точностью $\varepsilon = 0.1$.

- Определим знак функции на концах отрезка $[-3, 1]$:

$$f(-3) = e^{-3} - 6 \cdot (-3) - 3 = 15.049787068367863$$
$$f(1) = e^1 - 6 \cdot 1 - 3 = -6.281718171540955$$

- Находим точку x_0, равную середине отрезка $[-3, 1]$:

$$x_0 = \frac{(-3+1)}{2} = -1$$

Метод деления отрезка пополам (метод бисекций)

Пример

- Определяем значение функции в точке x_0:

$$f(x_0) = e^{-1} - 6 \cdot (-1) - 3 = 3.367879441171443$$

- Функция в точке x_0 имеет положительное значение, поэтому отбрасываем половину отрезка, на концах которого функция имеет положительные знаки, т.е. отрезок $[-3, x_0]$.
- Отрезок $[x_0, 1]$ делим пополам, определяя таким образом следующее приближение x_1:

$$x_1 = \frac{(-1+1)}{2} = 0$$

- Определяем значение функции в этой точке:

$$f(x_1) = e^0 - 6 \cdot 0 - 3 = -2$$

Метод деления отрезка пополам (метод бисекций)

Пример

- $f(x_1) < 0$, следовательно, выбираем отрезок $[x_0, x_1]$ и делим его пополам:

$$x_2 = \frac{(-1+0)}{2} = -0.5$$

- $f(x_2) > 0$, следовательно, выбираем отрезок $[x_2, x_1]$ и делим его пополам:

$$x_3 = \frac{(-0.5+0)}{2} = -0.25$$

- $f(x_3) < 0$, поэтому выбираем отрезок $[x_2, x_3]$ и делим его пополам:

$$x_4 = \frac{(-0.5+(-0.25))}{2} = -0.375$$

Метод деления отрезка пополам (метод бисекций)

Пример

- $f(x_4) < 0$, поэтому выбираем отрезок $[x_2, x_4]$ и делим его пополам:

$$x_6 = \frac{(-0.5 + (-0.375))}{2} = -0.4375$$

- $f(x_5) > 0$, поэтому выбираем отрезок $[x_5, x_4]$ и проверим условие выхода по формуле (4):

$$|-0.375 - (-0.4375)| = 0.0625$$

$$0.0625 < 0.1$$

- Условие завершения расчета выполнено, поэтому любое число из интервала x_5, x_4 можно считать корнем уравнения, однако часто принимают середину этого отрезка:

$$x^* = \frac{(-0.4375 + (-0.375))}{2} = -0.40625$$

Таким образом искомое решение уравнения с точностью $\varepsilon = 0.1$ равно $x^* = -0.40625$.

Метод деления отрезка пополам (реализация)


```python
import math


def bisections(func, bounds, eps=1e-4):
    a, b = bounds
    if func(a) * func(b) > 0:
        print('На заданном интервале нет корней!')
        return

    x0 = (a + b) / 2
    while abs(a - b) >= eps:
        if func(a) * func(x0) > 0:
            a = x0
        else:
            b = x0

        x0 = (a + b) / 2

    return x0


def func(x):   #исходное уравнение
    return math.exp(x) - 6 * x - 3


x = bisections(func, (-3, 1)) #вызов функции bisections()
print(x)  #-0.386810302734375
```

ТОМСКИЙ
ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ

Метод Ньютона (касательных)

Метод Ньютона (касательных)

- В тех случаях, когда известно начальное приближение корня нелинейного уравнения $f(x) = 0$, эффективным решением является метод Ньютона (касательных), однако данный метод имеет свои ограничения, влияющие на его сходимость. Рассмотрим эти ограничения.
- Пусть для функции $f(x)$ определены первая и вторая производные на заданном интервале $[a, b]$. При этом выполнено условие наличия корня на заданном интервале: $f(a) \cdot f(b) < 0$ и производные $f'(x)$ и $f''(x)$ сохраняют свои знаки на этом отрезке.
- В этом случае, основываясь на начальном приближении $x_1 \in [a, b]$, удовлетворяющему условию $f(x) \cdot f''(x) > 0$, построим следующую итерационную последовательность:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, n = 1, 2, 3, \ldots \tag{5}$$

которая сойдется к единственному на заданном отрезке корню x_0 уравнения $f(x_0) = 0$.

Метод Ньютона (касательных)

- В методе Ньютона процесс поиска корня заключается в том, что в качестве нового приближения к корню принимаются значения $x_1, x_2, x_3, \ldots$, которые являются абсциссами точек пересечения касательной к графику функции $y = f(x)$.
- Геометрически метод Ньютона является заменой малой дуги кривой $y = f(x)$ на касательную к этой кривой. При этом обязательно нужно знать начальное приближение корня x_1.
- Первоначально в качестве приближения выбирается точка a из заданного интервала $[a, b]$, если выполняется условие $f(a) \cdot f''(a) > 0$ или точка b, если выполняется условие $f(b) \cdot f''(b) > 0$
- Условием окончания итерационного процесса является следующее неравенство:

$$|x_{n+1} - x_n| \leqslant \varepsilon \tag{6}$$

где ε – заранее заданная точность вычисления корня нелинейного уравнения.

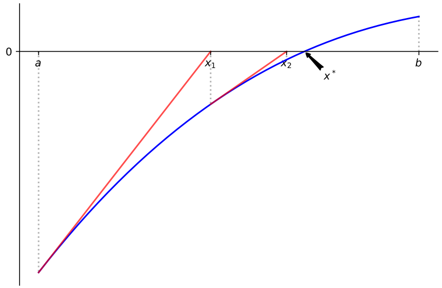


Рисунок 4 – Графическая иллюстрация метода Ньютона

Метод Ньютона (касательных)

Пример

- Рассмотрим решение нелинейного уравнения:

$$x^3 - 6x + 2 = 0$$

на интервале $[-4, -2]$ с точностью $\varepsilon = 0.1$.

- Определим первую и вторую производные для функции $f(x)$:

$$f'(x) = 3x^2 - 6$$
$$f''(x) = 6x$$

- Вычислим условия выбора начального приближения для корня уравнения:

$$f(a) \cdot f''(a) = \left[-4^3 - 6 \cdot (-4) + 2\right] \cdot [6 \cdot (-4)] = 912 > 0$$
$$f(b) \cdot f''(b) = \left[-2^3 - 6 \cdot (-2) + 2\right] \cdot [6 \cdot (-2)] = -72 < 0$$

- Так как по условию $f(x_0) \cdot f''(x_0) > 0$, выбираем в качестве начального приближения точку a, т.е. $x_0 = a$.

Метод Ньютона (касательных)

Пример

- Далее, используя итерационную последовательность (выражение (5)), вычислим новое приближение для корня:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = -4 - \frac{-4^3 - 6 \cdot (-4) + 2}{3 \cdot (-4)^2 - 6} = -3.0952$$

- Используя выражение (5), продолжим уточнение корня до заданной степени точности $\varepsilon = 0.1$:

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} = -4 - \frac{-3.0952^3 - 6 \cdot (-3.0952) + 2}{3 \cdot (-3.0952)^2 - 6} = -2.6959$$

$$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} = -4 - \frac{-2.6959^3 - 6 \cdot (-2.6959) + 2}{3 \cdot (-2.6959)^2 - 6} = -2.6062$$

- Поскольку $|x_3 - x_2| = |-2.6062 - (-2.6959)| = 0.0897 < \varepsilon$, можно говорить о том, что корень уравнения найден: $x^* = -2.6062$ с точностью $\varepsilon = 0.1$.

Метод Ньютона (реализация)


```python
def func(x):  #исходная функция
    return x ** 3 - 6 * x + 2


def fprime(x):  #первая производная
    return 3 * x ** 2 - 6


def fprime2(x):  #вторая производная
    return 6 * x


```

Метод Ньютона (реализация)


```python
def newton(func, fprime, fprime2, bounds, eps=1e-4):
    a, b = bounds
    if func(a) * fprime2(a) > 0:
        x0 = a
    elif func(b) * fprime2(b) > 0:
        x0 = b
    else:
        print('Неверно выбран начальный интревал!')
        return

    x = x0 - func(x0) / fprime(x0)
    while abs(x - x0) >= eps:
        x0 = x - func(x) / fprime(x)
        x = x0 - func(x0) / fprime(x0)

    return x


x = newton(func, fprime, fprime2, (-4, -2))
print(x) #-2.6016791319484533
```

ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Метод секущих (хорд)

Метод секущих (хорд)

- Пусть задана функция $f(x)$ действительной переменной. Задача состоит в нахождении корня уравнения
$$f(x) = 0 \tag{7}$$
на заданном отрезке $[a, b]$.
- Метод секущих можно получить из метода Ньютона (касательных) заменой $f'(x_k)$ разностным приближением:
$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \tag{8}$$
- Таким образом получим формулу для итерационного процесса:
$$x_{k+1} = x_k - \frac{f(x_k) \cdot (x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}, k = 1, 2, \ldots \tag{9}$$
- Метод секущих является *двухшаговым*, т.е. для определения нового приближения x_{k+1} необходимы две предыдущие итерации x_k и x_{k-1}. Для уравнения (7) также требуется задавать *два* начальных приближения x_0 и x_1.

Метод секущих (хорд)

- Геометрическая интерпретация метода секущих представлена на рисунке 5. Новое приближение для искомого корня x_{k+1} – это абсцисса точки пересечения секущей с осью OX.
- Другими словами, на отрезке $[x_{k-1}, x_k]$ функция $f(x)$ интерполируется полиномом первой степени и за каждое новое приближение x_{k+1} принимается корень этого полинома.
- При реализации метода секущих выбирают один из следующих критериев останова:

1. $f(x_k) < \varepsilon$ – значение функции на текущей итерации стало меньше значения ε.
2. $|x_k - x_{k-1}| < \varepsilon$ – изменение x_k после текущей итерации стало меньше значения ε.

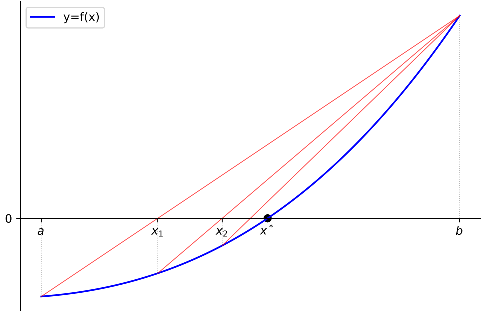


Рисунок 5 – Геометрическая интерпретация метода секущих

Метод секущих (хорд)

Пример

- Рассмотрим решение нелинейного уравнения:

$$x^3 - 18x - 83 = 0$$

  на отрезке $[3, 8]$ с заданной точностью $\varepsilon = 1 \times 10^{-4}$.

- В качестве начальных приближений возьмем концы заданного отрезка, на котором отделен корень исходного уравнения: $x_0 = 3$ и $x_1 = 8$. Вычисления будем вести до тех пор, пока не выполнится условие: $|x_{i+1} - x_i| < \varepsilon$.
  Далее, используя итерационную формулу (9), вычислим новое приближение корня:

$$x_2 = x_1 - \frac{f(x_1) \cdot (x_1 - x_0)}{f(x_1) - f(x_0)} = 8 - \frac{(8^3 - 18 \cdot 8 - 83) \cdot (8 - 3)}{(8^3 - 18 \cdot 8 - 83) - (3^3 - 18 \cdot 3 - 83)} =$$

$$= 8 - \frac{(285) \cdot 5}{(285) - (-110)} = 8 - \frac{1425}{395} = 8 - 3.60759 = 4.39241$$

Метод секущих (хорд)

Пример

- Найдем следующее приближение x_3:

$$x_3 = x_2 - \frac{f(x_2) \cdot (x_2 - x_1)}{f(x_2) - f(x_1)} = 4.39241 - \frac{(4.39241^3 - 18 \cdot 4.39241 - 83) \cdot (4.39241 - 8)}{(4.39241^3 - 18 \cdot 4.39241 - 83) - (8^3 - 18 \cdot 8 - 83)} =$$
$$= 4.39241 - \frac{278.93796}{-362.31964} = 5.16227$$

- Используя выражение (9) продолжим уточнение корня до заданного значения ε:

$$x_4 = 5.91995 \qquad x_7 = 5.70512$$
$$x_5 = 5.67875 \qquad x_8 = 5.70511$$
$$x_6 = 5.70393$$

- Поскольку $|x_8 - x_7| < \varepsilon$, корень уравнения найден, $x^* = 5.7051$.

Метод секущих (реализация)


```python
def secant(func, x0, x1, eps=1e-4):
    x_prev, x = x0, x1

    while abs(x - x_prev) >= eps:
        x, x_prev = (x - func(x) * (x - x_prev)
                     / (func(x) - func(x_prev)), x)

    return x


def func(x):
    return x ** 3 - 18 * x - 83


x = secant(func, 3, 8)
print(x) #5.705115794637447
```

ТОМСКИЙ
ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ

Метод простых итераций

Метод простых итераций

- На первом этапе необходимо заменить исходное нелинейное уравнение (1) на эквивалентное уравнение следующего вида:
$$x = \varphi(x) \tag{10}$$
- Начальное приближение корня определяется следующим образом:
$$x_0 = \varphi(a) \tag{11}$$
где a – левая граница интервала $[a, b]$, на котором предполагается наличие корня.
- Аналогичным образом можно получить $x_1 = \varphi(x_0)$. Таким образом, подставляя каждый раз новое приближение корня в выражение (10), получим последовательность итерационных вычислений:
$$x_{n+1} = \varphi(x_n), n = 1, 2, 3, \ldots \tag{12}$$
- Итерационный процесс уточнения корней будет продолжаться до тех пор, пока не станут близкими значения двух последних итераций:
$$|x_{n+1} - x_n| \leqslant \varepsilon \tag{13}$$
- Данный метод сходится к решению, только при выполнении условия сходимости:
$$\left|\varphi'(x)\right| < 1 \tag{14}$$
для любого значения $x \in [a, b]$.

Метод простых итераций

- Рассмотрим графическую интерпретацию данного метода.
- Для этого необходимо построить графики двух функций: $y = x$ и $y = \varphi(x)$ (рисунок 6).
- Корнем x_0 уравнения $x = \varphi(x)$ будет являться абсцисса точки пересечения кривой $y = \varphi(x)$ с прямой $y = x$.
- Скорость сходимости процесса уточнения корней зависит от величины $|\varphi'(x)|$.
- Чем меньше ее значение вблизи корня, тем быстрее сходится процесс.
- Выбор выражения для эквивалентной функции должен быть сделан таким образом, чтобы удовлетворялось условие (14).

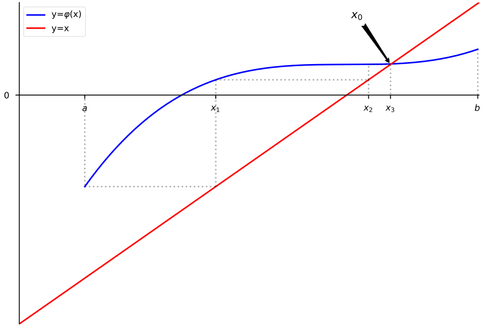


Рисунок 6 – Графическая интерпретация метода итераций

Пример

- Рассмотрим решение нелинейного уравнения:

$$e^x - 6x - 3 = 0$$

  методом итераций на отрезке $[-3, 1]$ с точностью $\varepsilon = 0.0001$.

- Запишем выражение для эквивалентной функции:

$$\varphi(x) = \frac{e^x - 3}{6}$$

- Первое приближение x_0 найдем как $\varphi(-3)$:

$$x_0 = \varphi(-3) = \frac{e^{-3} - 3}{6} = -0.4917$$

Пример

- Следующее приближение x_1 найдем как $\varphi(x_0)$:

$$x_1 = \varphi(x_0) = \frac{e^{-0.4917} - 3}{6} = -0.3981$$

- Следующее приближение x_2 найдем как $\varphi(x_1)$:

$$x_2 = \varphi(x_1) = \frac{e^{-0.3981} - 3}{6} = -0.3881$$

- Аналогичным образом находим $x_3 = -0.3869, x_4 = -0.3868, x_5 = -0.3868$

Поскольку модуль разности $|x_4 - x_5| \leqslant \varepsilon$, то принимаем, что решением является значение $x^* = x_5 = -0.3868$.

Метод простых итераций (реализация)


```python
def func(x):
    return np.exp(x) - 6 * x - 3


def phi(x):
    return (np.exp(x) - 3) / 6


def iterations(phi, a, eps=1e-4):
    i = 1
    x = phi(a)
    x0 = phi(x)

    while abs(x - x0) >= eps:
        x = phi(x0)
        x0 = phi(x)
        i += 1
        if i == 10000:
            print('Выполнено 10000 итераций, решение не нейдено!')
            return

    return x0


x = iterations(phi, -3)
print(x) #-0.386796779231439
```

ТОМСКИЙ
ПОЛИТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ

Контакты

Игорь Михайлович Долганов,
к.т.н., доцент ОХИ ИШПР

Учебный корпус №2, ауд. 136
dolganovim@tpu.ru
+7-960-978-43-07

Благодарю за внимание!